

Modeling and Formal Analysis of Meta- Ecosystems with Dynamic Structure using Graph Transformation

Boris Flotterer, Maria Maximova, Sven Schneider,
Johannes Dyck, Christian Zöllner, Holger Giese,
Christelle Hély, Cédric Gauchere

Technische Berichte Nr. 147

des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam



Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Boris Flotterer | Maria Maximova | Sven Schneider | Johannes Dyck | Christian
Zöllner | Holger Giese | Christelle Hély | Cédric Gaucherel

Modeling and Formal Analysis of Meta-Ecosystems with Dynamic Structure using Graph Transformation

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche
Nationalbibliografie; detailed bibliographic data are available on the Internet
via <http://dnb.dnb.de/>.

Universitätsverlag Potsdam 2022
<http://verlag.ub.uni-potsdam.de/>

Am Neuen Palais 10, 14469 Potsdam
Phone: +49 (0)331 977 2533 / Fax: 2292
Email: verlag@uni-potsdam.de

The series **Technische Berichte des Hasso-Plattner-Instituts für Digital
Engineering an der Universität Potsdam** is edited by the professors of the
Hasso Plattner Institute for Digital Engineering at the University of Potsdam.

ISSN (print) 1613-5652
ISSN (online) 2191-1665

The work is protected by copyright.
Layout: Tobias Pape
Print: docupoint GmbH Magdeburg

ISBN 978-3-86956-533-0

Also published online on the publication server of the University of Potsdam:
<https://doi.org/10.25932/publishup-54764>
<https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-547643>

The dynamics of ecosystems is of crucial importance. Various model-based approaches exist to understand and analyze their internal effects. In this paper, we model the space structure dynamics and ecological dynamics of meta-ecosystems using the formal technique of Graph Transformation (short GT). We build GT models to describe how a meta-ecosystem (modeled as a graph) can evolve over time (modeled by GT rules) and to analyze these GT models with respect to qualitative properties such as the existence of structural stabilities. As a case study, we build three GT models describing the space structure dynamics and ecological dynamics of three different savanna meta-ecosystems. The first GT model considers a savanna meta-ecosystem that is limited in space to two ecosystem patches, whereas the other two GT models consider two savanna meta-ecosystems that are unlimited in the number of ecosystem patches and only differ in one GT rule describing how the space structure of the meta-ecosystem grows. In the first two GT models, the space structure dynamics and ecological dynamics of the meta-ecosystem shows two main structural stabilities: the first one based on grassland-savanna-woodland transitions and the second one based on grassland-desert transitions. The transition between these two structural stabilities is driven by high-intensity fires affecting the tree components. In the third GT model, the GT rule for savanna regeneration induces desertification and therefore a collapse of the meta-ecosystem. We believe that GT models provide a complementary avenue to that of existing approaches to rigorously study ecological phenomena.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 8 |
| 2 | Material and Methods | 10 |
| 2.1 | Typed Graphs and Graph Transformation | 10 |
| 2.2 | Examples of Meta-Ecosystems Modeled as Graph Transformation Sys- tems | 13 |
| 2.3 | Labeled Transition Systems and Strongly Connected Components . . | 14 |
| 2.4 | Analysis of Graph Transformation Systems | 18 |
| 3 | Results | 20 |
| 3.1 | Analysis of GT Model \mathcal{S}_2 | 20 |
| 3.2 | Analysis of GT Model \mathcal{S}_U | 22 |
| 3.3 | Analysis of GT Model \mathcal{S}_D | 22 |
| 4 | Discussion | 24 |
| 4.1 | Acknowledgments | 29 |
| A | Supplementary Material | 34 |
| A.1 | Additional Details on Typed Graphs and Graph Transformation . . . | 34 |
| A.2 | Additional Details on the Computation Tree Logic (CTL) | 38 |
| A.3 | Proofs for Results | 39 |

1 Introduction

Over the long term, ecosystems are growing and shrinking in space according to intrinsic and extrinsic processes as well as to neighboring opportunities. To understand and predict such changing structures, ecologists usually model them as assumed empty spaces (i.e., fixed ecosystem patches) gradually filled by ecosystems ([54]). It could be more efficient to model new ecosystem patches according to present ecological processes and without assumptions about the influencing spatial structure ([15, 19]). Several theories have been proposed to handle space in ecology. To the best of our knowledge, this paper illustrates for the first time in ecology the use of discrete-state discrete-event models to capture an unlimited (i.e., unbounded) growth of meta-ecosystems.

The landscape ecology theory ([24]) focuses on spatially fixed mosaics composed of various adjacent ecosystem patches (e.g. forest stands, crop fields) and considers biotic and abiotic processes they involve ([13, 14, 53]). The meta-population theory provides a new understanding on how distant populations of a given species with connections (fluxes of individuals) to other ones influence the fate of the meta-population ([25, 51]). The meta-community theory ([34]) is a functional extension of the meta-population theory considering various communities of the same kind that are spatially connected ([29]). Finally, the meta-ecosystem theory takes into account biotic and abiotic fluxes between different and potentially distant spatial units displaying common arbitrary characteristics. In this paper, we consider a meta-ecosystem as “[...] a set of ecosystems connected by spatial flows of energy, materials, and organisms across ecosystem boundaries” ([35]). So far, none of these four theories has been able to convincingly model a growing (or shrinking) meta-ecosystem in space over time, i.e., a meta-ecosystem with new ecosystem patches added (or removed) to (from) the meta-ecosystem during the simulation ([37]).

The previously mentioned theories focus on the impacts of space in ecological (i.e., spatial) dynamics but they usually bypass modeling the space structure dynamics. In addition, most models used in ecology are based on a quantitative approach and, more specifically, on ordinary differential equations ([35, 49, 52]). The calibration and resolution of such equation systems usually show an exponential increase in complexity with the number of involved variables and the number of ecosystem patches. Due to this increase in complexity and to be able to handle more variables and ecosystem patches, it could be helpful to simplify the used models ([52]). In contrast, qualitative models may be a viable alternative grasping this ecological complexity with a flexible and still rigorous control ([8, 36, 42]) where the presence/absence of components but not their number is captured as in quantitative models.

Qualitative discrete-state discrete-event models based on Petri nets and their firing relation to model complex ecosystems integrating multiple components have been already used in ecology (e.g. [16]). However, using Petri nets often assumes a fixed (frozen) structure of the meta-ecosystem ([5, 36, 40]) whereas real meta-ecosystems are often growing and/or shrinking dynamically ([31]). To model meta-ecosystems, we therefore need to model a dynamic system on a dynamic structure (i.e., a kind of space structure growth, also called DS2 ([18])) possibly assuming and knowing how space structure will change.

The formal technique of Graph Transformation (short GT), which allows for rule-based transformation of discrete-state models, is a well-known general purpose technique that has been developed in theoretical computer science with a broad range of available tools for modeling and analysis (e.g. [12, 19, 32]). GT allows for adding and removing nodes/edges (i.e., ecosystem patches or ecological components) using GT rules (which describe ecological processes) as soon as the (meta-)ecosystem is represented as a graph with a changing topology (i.e., the neighboring relationships between ecosystem patches).

We illustrate our approach on savannas (*sensu lato*) as these ecosystems are widely represented (one fifth of the Earth land surface) and quite simple in terms of vegetation structure (i.e., tree and grass interplay) but among the ecosystems most sensitive to future global changes ([4]). Savannas result from the interplay between various components (e.g. local climate, fire, vegetation, hydrology, topography, human populations, and activities ([26])) and may be considered as meta-ecosystems ([23, 44]). Savannas are also often associated (i.e., nearby) to crop fields and woodlands in landscapes ([50]). Understanding the drivers that structure savanna meta-ecosystems is thus required to improve future management efforts ([30, 43]).

The aim of this paper is to evaluate whether the formal technique of GT is suitable for the modeling of meta-ecosystems with fixed or dynamic space structure. To achieve this, we (a) determine whether GT is suitable to model the space structure dynamics and ecological dynamics of savanna meta-ecosystems with a fixed number of ecosystem patches, (b) analyze using GT the influence of the space structure dynamics generated when the number of ecosystem patches is unlimited and varies due to the ecological processes at work (e.g. vegetation dispersal, impact of fire, vegetation regrowth), and, finally, (c) discuss the pros and cons of the usage of GT in ecology.

2 Material and Methods

In section 2.1, we introduce the formal technique of GT based on typed graphs. In our approach, we use GT for the modeling of space structure dynamics and ecological dynamics of meta-ecosystems captured by so-called GT models. Afterwards, in section 2.2, we define three concrete GT models for three different savanna meta-ecosystems. Subsequently, in section 2.3, we introduce the notion of labeled transition systems describing state spaces generated by GT models together with the central operations for merging these state spaces according to a suitable equivalence notion. Finally, in section 2.4, we discuss analysis techniques dedicated to GT models and use them to provide the reachability properties of three concrete meta-ecosystem models.

2.1 Typed Graphs and Graph Transformation

In this subsection, we provide a short informal introduction to the well-established formal technique of GT ([12]) based on typed graphs (see section A.1 for more details on formal definitions).

As discrete data structures, *graphs* can be used to represent system states. In particular, graphs are efficient at visually capturing relationships between the system elements (called *nodes*) using connections (called *edges*) between these elements. In this paper, we consider *directed graphs* where each edge has a single source node and a single target node as well as allow for the special case of edges, called *loops*, where the source and the target node are the same (see Figure 2.1b for a directed graph with a loop). Moreover, we allow that different edges may have the same source and target nodes, leading to so-called *parallel edges*. In this paper, we focus on finite graphs with a finite number of nodes and edges, to ease tool-based analysis. In our ecological case study, nodes represent ecological components as well as ecosystem patches, which describe the boundaries of ecosystems, while edges represent the interaction between ecological components on these ecosystem patches.

Definition 1 (Graph). If N is a finite set of nodes, E is a finite set of edges, $s : E \rightarrow N$ maps each edge to its source node, and $t : E \rightarrow N$ maps each edge to its target node, then $G = (N, E, s, t)$ is a *directed graph* also called *graph* for short in the following.

There exist various graph formalisms that allow for distinguishing between different kinds of nodes and edges. Here, we follow the standard approach of *typed graphs* (instead of e.g. labeled or attributed graphs). In typed graphs, each node and each edge has a certain type that is contained in a given *type graph* TG . In general, a type graph TG specifies which node and edge types are allowed in graphs that are typed



(a) The type graph TG_S for the meta-ecosystems S_2 , S_U , and S_D

(b) The initial graph G_0 for the meta-ecosystems S_2 , S_U , and S_D

Figure 2.1: Type graph and initial graph for the meta-ecosystems S_2 , S_U , and S_D . Squares in color represent ecological components of a meta-ecosystem as well as ecosystem patches. Arrows represent interactions between ecological components and ecosystem patches.

over TG . At the same time, a type graph restricts the set of all admissible graphs by preventing the existence of certain edges (based on their types) between certain nodes (based on their types).

The type graph TG_S of our ecological case study is given in Figure 2.1a. TG_S contains as nodes the node types Eco , $Grass$, and $Tree$ for modeling ecosystem patches, grass, and trees, respectively. Also, we use the node type $Rain$ for modeling rainfall on ecosystem patches during the rainy season. TG_S contains as edges the edge types $g2e$ and $t2e$ to model that a certain ecosystem patch contains grass and trees, respectively. Moreover, the edge type $r2e$ is used to model that rainfall is possible on a connected ecosystem patch. Finally, the edge types on and off are used to model if there is or if there is no rainfall on the connected ecosystem patches, respectively. Note also that TG_S prevents e.g. edges from nodes of type Eco to nodes of type $Grass$ as well as between nodes of type $Grass$ and $Rain$ altogether.

The graph G_0 , which is an example of a graph that is typed over the type graph TG_S , is given in Figure 2.1b. Intuitively, the graph G_0 represents a state of a meta-ecosystem consisting of two ecosystem patches (given by two Eco -nodes), each containing trees and grass (given by $Tree$ - and $Grass$ -nodes connected to the corresponding Eco -nodes), during the rainy season (given by the $Rain$ -node with an on -loop connected to both Eco -nodes). Note that a graph such as G_0 does not need to contain nodes or edges of each type contained in the type graph TG_S .

To transform a given graph into a new one, GT rules are used. An application of a GT rule to a graph modifies this graph using two basic operations, namely removal and addition of certain graph elements (i.e., nodes or edges). For this purpose, a GT rule contains a *left-hand-side graph* L , an *interface graph* I , and a *right-hand-side graph* R . The left-hand-side graph L is a subgraph of the graph to be modified (up to renaming of graph elements). The interface graph I , which is a subgraph of the graph L , describes the elements of L that should not be removed (i.e., the elements contained in I and L should not be removed while the elements contained in L but not in I (i.e., in $L - I$) should be removed). The right-hand-side graph R , which has I as a subgraph, describes the elements that should be added (i.e., the elements contained in R but not in I (i.e., in $R - I$) should be added). Moreover, a GT rule contains a finite

set *NAC* of *forbidden graphs* X (also called *negative application conditions* in computer science), which have L as a subgraph and are used to restrict the applicability of the GT rule (see section A.1 for more details on formal definitions). In the ecological context, GT rules represent the transitions between meta-ecosystem states and hence are directly linked to ecological processes.

Definition 2 (GT Rule). If L is a left-hand-side graph, R is a right-hand-side graph, I is an interface graph that is contained in L and R , and NAC is a finite set of forbidden graphs X containing L , then $\rho = (NAC, L, I, R)$ is a *GT rule* also called *rule* for short in the following.

For examples of GT rules see e.g. the rule *GrowGrass* in Table 2.1. Intuitively, this rule describes the ecological process that grass may grow on an ecosystem patch during the rainy season if there is no grass on that ecosystem patch yet. In our visualization, we depict the left-hand-side graph L of the rule in the square left to the double arrow, the right-hand-side graph R of the rule in the square right to the double arrow, the name of the rule over the double arrow, and the single forbidden graph from the set NAC in the red square over the left-hand-side graph L . The red triangle between the left-hand-side graph L and the forbidden graph visualizes that the extension of L to the forbidden graph disables the rule application. Moreover, we capture the node identities in L , R , and forbidden graphs using numbers in the nodes' bottom corners. Note that we do not visualize the interface graph I in the following since it is used for technical reasons only.

GT steps then apply a rule $\rho = (NAC, L, I, R)$ to a given graph G to modify G into a resulting graph H . For this purpose, the left-hand-side graph L is embedded into the graph G such that this embedding cannot be extended to an embedding of one of the forbidden graphs in NAC . Afterwards, the graph elements from $L - I$ are removed on the basis of where L has been embedded into the graph G . This removal then results in an embedding of the interface graph I into the intermediate graph D , which retains information on where graph elements have been removed. Finally, the graph elements from $R - I$ are added to the graph D resulting in the graph H . Hereby, the embedding of I into D is used to determine where graph elements should be added to D (see section A.1 for more details on formal definitions).

Definition 3 (GT Step). If $\rho = (NAC, L, I, R)$ is a rule, G is a graph, L can be embedded into G , the embedding of L in G cannot be extended to an embedding of any forbidden graph in NAC , D is a graph obtained by removing $L - I$ from G resulting in an embedding of I in D , and H is a graph obtained by adding $R - I$ to D , then $G \xrightarrow{\rho} H$ is a *GT step* modifying graph G into resulting graph H .

A GT step for an application of the rule *GrowGrass* from Table 2.1 works as follows. Whenever a subgraph containing an *Eco*-node connected to a *Rain*-node with an *on*-loop is found in a graph to be modified and there is no *Grass*-node connected to the found *Eco*-node, a *Grass*-node is added to the found subgraph and connected to the *Eco*-node.

Graph Transformation Systems (GTSs) are used to describe how and when graphs have to be modified following the concept of GT steps introduced before. A GTS

contains (a) a unique initial graph from which all sequences of GT steps start, (b) a finite set of GT rules that may be applied to perform GT steps of the GTS, and (c) an assignment of each contained GT rule to a priority given by a natural number. A GTS uses these priorities to only permit those GT steps modifying a graph G for which no other rule with higher priority can be applied to the same graph G . Such rules with higher priorities represent mandatory processes in meta-ecosystems. For examples of GTSs, see the following subsection. Note that in the ecological context, we also call GTSs *GT models*.

Definition 4 (Graph Transformation System (GTS)). If G_0 is a typed graph (called *initial graph*), P is a finite set of rules, and $\kappa : P \rightarrow \mathbf{N}$ is a mapping assigning a priority (i.e., a natural number) to each rule in P , then $\mathcal{S} = (G_0, P, \kappa)$ is a *graph transformation system (GTS)*.

2.2 Examples of Meta-Ecosystems Modeled as Graph Transformation Systems

In this subsection, we model the ecological dynamics as well as the space structure dynamics of three different savanna meta-ecosystems using GTSs. In the first GTS (also called GT model in the following), we consider a savanna meta-ecosystem that is limited in space to two ecosystem patches, whereas in the other two GTSs we consider two savanna meta-ecosystems that are unlimited in the number of ecosystem patches and only differ in one rule describing how the space structure of the meta-ecosystem grows.

For all three GT models we use the same initial graph given in Figure 2.1b that consists of two ecosystem patches with trees and grass during the rainy season. Starting with this initial graph, we then apply the corresponding rules to model the possible ecological dynamics and space structure dynamics of meta-ecosystems.

In the first GT model \mathcal{S}_2 , we model a savanna meta-ecosystem where grass and trees can grow during the rainy season and where grass and trees can disappear due to low or high intensity fire during the dry season. \mathcal{S}_2 contains six rules with the same priority (see the six rules in Table 2.1). The rules *RainOff* and *RainOn* allow to model the change between the rainy and the dry season in the meta-ecosystem. The rules *GrowGrass* and *GrowTree* allow under certain conditions to model the growth of grass and trees on ecosystem patches during the rainy season, respectively. The rules *LowFire* and *HighFire* allow to model the impact of a low and high intensity fire on affected ecosystem patches during the dry season, respectively. Note that none of the rules allows for the addition or deletion of ecosystem patches as the meta-ecosystem \mathcal{S}_2 is limited in space to two ecosystem patches only.

Definition 5 (GT Model \mathcal{S}_2). If G_0 is the initial graph from Figure 2.1b, P_2 is the set of 6 rules $\{\textit{RainOff}, \textit{RainOn}, \textit{LowFire}, \textit{HighFire}, \textit{GrowGrass}, \textit{GrowTree}\}$ (see Table 2.1), and κ_2 assigns the same priority 0 to each rule in P_2 , then $\mathcal{S}_2 = (G_0, P_2, \kappa_2)$ is a GT model.

In the second GT model \mathcal{S}_U , we model a savanna meta-ecosystem as in \mathcal{S}_2 with the additional space structure dynamics. \mathcal{S}_U contains in addition to the six rules from \mathcal{S}_2 the rules *DelEco* and *AddEcoU* (see Table 2.2). The rule *DelEco* allows to remove an ecosystem patch without trees and grass from the meta-ecosystem, whereas the rule *AddEcoU* allows ecosystem patches with grass to appear during the rainy season. To ensure that an ecosystem patch without trees and grass is removed from the meta-ecosystem before another rule can be applied, the rule *DelEco* has a higher priority than the other rules.

Definition 6 (GT Model \mathcal{S}_U). If G_0 is the initial graph from Figure 2.1b, P_U is the set of 8 rules $\{RainOff, RainOn, LowFire, HighFire, GrowGrass, GrowTree, DelEco, AddEcoU\}$ (see Table 2.1 and Table 2.2), and κ_U assigns the same priority 0 to each rule in $P_U - \{DelEco\}$ and the priority 1 to the rule *DelEco*, then $\mathcal{S}_U = (G_0, P_U, \kappa_U)$ is a GT model.

In the third GT model \mathcal{S}_D , we model a savanna meta-ecosystem as in \mathcal{S}_2 with a different kind of space structure dynamics as in \mathcal{S}_U . \mathcal{S}_D differs from the second GT model \mathcal{S}_U in only one rule describing how the space structure of the meta-ecosystem grows: The rule *AddEcoD* (see Table 2.2) is used in \mathcal{S}_D instead of the rule *AddEcoU* (explained before) to model that grass may spread to new ecosystem patches during the rainy season from ecosystem patches with grass and trees. Also, the rule *DelEco* has a higher priority than the other rules to ensure that an ecosystem patch without trees and grass is removed from the meta-ecosystem before another rule can be applied.

Definition 7 (GT Model \mathcal{S}_D). If G_0 is the initial graph from Figure 2.1b, P_D is the set of 8 rules $\{RainOff, RainOn, LowFire, HighFire, GrowGrass, GrowTree, DelEco, AddEcoD\}$ (see Table 2.1 and Table 2.2), and κ_D assigns the same priority 0 to each rule in $P_D - \{DelEco\}$ and the priority 1 to the rule *DelEco*, then $\mathcal{S}_D = (G_0, P_D, \kappa_D)$ is a GT model.

2.3 Labeled Transition Systems and Strongly Connected Components

As already introduced in section 2.1, GTs execute GT steps between successive graphs labeled with the rules used for these steps. When all possible graphs that can be reached using successive GT step computation are determined, we obtain a *labeled transition system* (LTS).

An LTS, which is a certain kind of graph as well, consists of a possibly infinite set of states S , a finite set of labels L , a distinguished initial state Z , and a step relation R where each step between two states is labeled using an element of L . In the ecological context, an LTS represents all trajectories of the modeled meta-ecosystem starting in the initial state G_0 .

| Rule | Ecological Interpretation |
|------|---|
| | The rainy season changes to the dry season (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |
| | The dry season changes to the rainy season (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |
| | A low intensity fire may destroy grass during the dry season (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |
| | A high intensity fire may destroy trees and grass during the dry season when grass is present as a combustive (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |
| | Grass may grow during the rainy season if there is no grass yet (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |
| | Trees may spread to another ecosystem patch with grass during the rainy season if there are no trees yet on that ecosystem patch (rule used in GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D). |

Table 2.1: Rules for the GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D (see also Table 2.2). For each rule, its left-hand-side graph L is depicted in the square left to the double arrow, its right-hand-side graph R is depicted in the square right to the double arrow, its name is written over the double arrow, and its possibly existing forbidden graphs are depicted in the red squares over the left-hand-side graph L . The red triangle between the left-hand-side graph L and its forbidden graph visualizes that the extension of L to the forbidden graph disables the rule application. The node identities in L , R , and forbidden graphs are captured using numbers in the nodes' bottom corners.

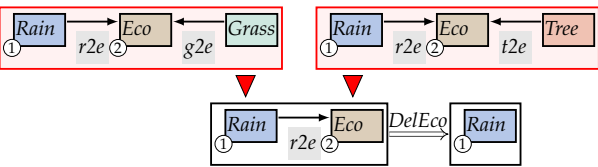
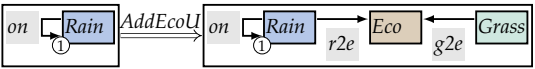
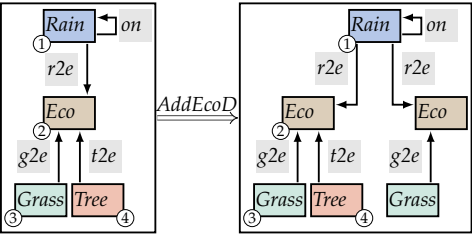
| Rule | Ecological Interpretation |
|---|---|
|  | <p>Ecosystem patches without grass and trees are removed from the meta-ecosystem (rule with higher priority used in GT models \mathcal{S}_U and \mathcal{S}_D).</p> |
|  | <p>Ecosystem patches with grass may appear during the rainy season (rule used in GT model \mathcal{S}_U).</p> |
|  | <p>Grass may spread to new ecosystem patches during the rainy season from ecosystem patches with grass and trees (rule used in GT model \mathcal{S}_D).</p> |

Table 2.2: Further rules for the GT models \mathcal{S}_U and \mathcal{S}_D (see also Table 2.1)

Definition 8 (Labeled Transition System (LTS)). If S is a set of states, L is a finite set of labels, $Z \in S$ is an initial state, $R \subseteq S \times L \times S$ is a relation for discrete labeled steps on states, then $\mathcal{L} = (S, L, Z, R)$ is a *labeled transition system (LTS)*.

In the context of GTSs, we call an LTS also a *state space* of the GTS and interpret the components of such an LTS as follows. The states are all graphs, which are reachable from the initial graph of the GTS, the labels are the rules of the GTS, the initial state Z is the initial graph of the GTS, and the step relation is obtained using GT steps.

As the next step, we want to introduce an aggregation operation that merges several LTS states when they are understood to be equivalent. Such merging of LTS states leads then to a smaller and simpler LTS. For this purpose, we first introduce the notion of *paths* through the states of an LTS. Note that the iterative application of the step relation R of an LTS results in a multi-step relation $R^* \subseteq S \times L^* \times S$. This relation thereby characterizes the paths between two states using lists of labels. That is, a tuple (s, π, s') of R^* contains the start state s , a finite list of labels π (the empty list is denoted by ϵ), and the end state s' of that path. In the ecological context, a path corresponds to a trajectory of the modeled meta-ecosystem.

Definition 9 (Paths of an LTS). Each LTS $\mathcal{L} = (S, L, Z, R)$ induces a *path relation* $R^* \subseteq S \times L^* \times S$, which is the smallest relation satisfying (a) $(s, \epsilon, s) \in R^*$ and (b) $(s, \pi, s') \in R^*$ and $(s', \ell, s'') \in R$ implies $(s, \pi \cdot \ell, s'') \in R^*$ where $\pi \cdot \ell$ means that the label ℓ is added to the end of the list π .

Two states s and s' of an LTS are understood to be equivalent when there is a path from s to s' and vice versa. In general, this equivalence results in a partitioning of the set of states of an LTS where each partition is as large as possible. Such partitions are called *strongly connected components (SCCs)*. Each state within an SCC can be reached from any other state of this SCC.

Definition 10 (Strongly Connected Component (SCC) of an LTS). If $\mathcal{L} = (S, L, Z, R)$ is an LTS and $C \subseteq S$ is a maximal subset of the states of \mathcal{L} satisfying that for each state s and s' in C there is some list of labels π such that $(s, \pi, s') \in R^*$, then C is a *strongly connected component (SCC)* of \mathcal{L} .

Based on the notion of SCCs, we now define the aggregation operation, which takes an LTS as an input and returns another LTS as an output. For each SCC of the input LTS, the aggregation operation merges all states of this SCC into a single state of the output LTS. The steps between these merged SCCs are then obtained by lifting the steps of the input LTS to the output LTS. This means that when s and s' are states contained in two (possibly the same) SCCs C and C' , respectively, then each step between s and s' results in a step between the states obtained from merging the two SCCs C and C' .

Definition 11 (SCC-Based Merged LTS). If $\mathcal{L} = (S, L, Z, R)$ is an LTS, \mathcal{C} is the set of all SCCs of \mathcal{L} , $Z' \in \mathcal{C}$ is the initial SCC containing the initial state Z of \mathcal{L} (i.e., $Z \in Z'$), and $(C, \ell, C') \in R'$ if there are states $s \in C$ and $s' \in C'$ such that $(s, \ell, s') \in R$, then $\mathcal{L}' = (C, L, Z', R')$ is the *SCC-based merged LTS* of \mathcal{L} .

In the ecological context, we are interested in the SCC-based merged state spaces (which can be understood to be directed acyclic graphs) induced by GT models. These SCC-based merged state spaces identify sets of states that are structurally stable. That is, all states in an SCC of such a state space are equal with respect to the states that can be reached from them. Hence, GT steps within an SCC describe reversible steps of the meta-ecosystem dynamics. In contrast, the step relation of an SCC-based merged state space describes how one SCC can be reached from another one using an irreversible step of the meta-ecosystem dynamics.

2.4 Analysis of Graph Transformation Systems

A wide variety of analysis techniques have been developed in the past for GTSs. Some of these analysis techniques are based on the computation of tractably small finite state spaces. Some other analysis techniques are symbolic (also called static) and allow for reasoning on the generated state space only based on the rules of the GTS and without calculating the GT steps on particular graphs. In this subsection, we discuss only those analysis techniques that we apply to the three GT models (see section 2.2) from our ecological case study.

Firstly, an exhaustive behavioral analysis of GTSs can be performed when the entire state space can be explicitly computed (depending on time and hardware restrictions). For example, the computation tree logic (CTL) ([6]) can be used to check whether a specific target state can be reached using some finite path from the initial state of the state space (see section A.2 for a formal introduction of CTL). In the savanna ecological context, such a target state could be e.g. a state that does not have any *Tree*-node. Moreover, CTL allows for recursive properties, which could require that the reachable state must not have a path back to another specific state. In the savanna ecological context, the latter state could be e.g. a state that has at least one *Tree*-node. CTL is well-suited for specifying properties of the generated state space due to its expressiveness and yet simple syntax and semantics. For our ecological case study, we used the tool GROOVE ([17]) that supports CTL by computing the entire state space first and then by evaluating the provided CTL properties.

Secondly, when the state space of the GTS is infinite or intractably large, it may be possible to adapt the GTS in a reasonable way so that it results in a finite state space. For this purpose, rules may be adapted such that they are applicable to fewer graphs while still producing the same resulting graphs as in the infinite case. For example, rules may be modified such that an upper threshold of graph elements of a certain type is never exceeded. Such an adaptation leads to a finite state space that can be inspected for specific properties. Since this adaptation cuts off all but a finite part of an infinite state space, only certain properties can then be checked. For example, a state that is reachable in the finite state space is also reachable in the infinite state space but not vice versa.

Thirdly, symbolic analysis techniques such as the principle of k-induction can be applied to GTSs that generate finite as well as infinite state spaces ([10, 46]). This particular technique attempts to establish an invariant, which requires that all

reachable states of a GTS satisfy a certain property such as e.g. that the number of graph elements of a certain type never exceeds a certain upper limit.

However, it is a well-known result in computer science that non-trivial properties on Turing-complete formalisms such as GTSs cannot be solved by fully-automatic procedures. This implies that implementations of symbolic techniques, such as k-induction, do not always succeed by proving or disproving a conjectured invariant. In cases where such available techniques for fully-automatic analysis fail, manual mathematical reasoning is the usual fallback solution. Manual mathematical reasoning then allows to verify properties of a GTS by relying on the knowledge of domain experts that understand the GTS at a more fundamental level compared to the generic fully-automatic procedures, which realize a fixed heuristics for analysis.

3 Results

In this section, we analyze the ecological dynamics as well as the space structure dynamics of the three savanna meta-ecosystems introduced in section 2.2 using the SCC-based merged state spaces of the corresponding GT models.

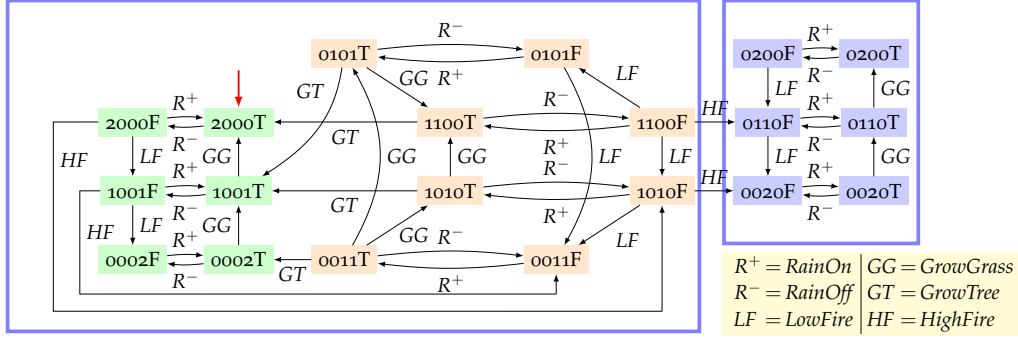
3.1 Analysis of GT Model \mathcal{S}_2

The following theorem captures the analysis results for the GT model \mathcal{S}_2 .

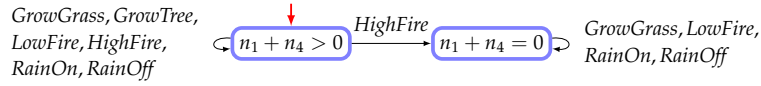
Theorem 1 (State Space Induced by \mathcal{S}_2). The GT model \mathcal{S}_2 generates a finite state space given in Figure 3.1a. The SCC-based merging of this finite state space results in the state space given in Figure 3.1b containing two SCCs.

The state space given in Figure 3.1a is computed using the general-purpose GT tool GROOVE ([17]). When inspecting each state of this state space, it turns out (by using the principle of k-induction) that all graphs reachable from the initial graph G_0 contain (1) precisely one *Rain*-node, (2) precisely one *on*-loop or precisely one *off*-loop, (3) a number of *Eco*-nodes all connected to the *Rain*-node, (4) a number of *Grass*-nodes each connected to a unique *Eco*-node, (5) a number of *Tree*-nodes each connected to a unique *Eco*-node, (6) no two different *Grass*-nodes connected to the same *Eco*-node, and (7) no two different *Tree*-nodes connected to the same *Eco*-node. Based on this observation, we can define for each graph G_i in the state space of \mathcal{S}_2 the abstraction (n_1, n_2, n_3, n_4, b) . This abstraction captures (i) the number n_1 of *Eco*-nodes in G_i with a connected *Tree*- and a connected *Grass*-node, (ii) the number n_2 of *Eco*-nodes in G_i with no connected *Tree*- and a connected *Grass*-node, (iii) the number n_3 of *Eco*-nodes in G_i with no connected *Tree*- and no connected *Grass*-node, (iv) the number n_4 of *Eco*-nodes in G_i with a connected *Tree*- and no connected *Grass*-node, and (v) the Boolean value $b = T$ if the *Rain*-node in G_i has an *on*-loop and the Boolean value $b = F$ if the *Rain*-node in G_i has an *off*-loop. In Figure 3.1a, we do not show the actual reachable graphs for presentation purposes but instead employ names of the form $n_1n_2n_3n_4b$ for the states using the five variables of the introduced abstraction. For example, the initial graph G_0 from Figure 2.1b has the name 2000T.

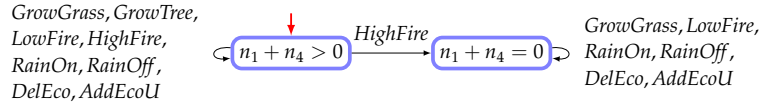
Moreover, the state space in Figure 3.1a is visually separated using blue rectangles into two SCCs that abstractly represent the two stable states of the GT model \mathcal{S}_2 . The left SCC describes \mathcal{S}_2 being in a state with at least one *Tree*-node and the right SCC describes \mathcal{S}_2 being in a state without any *Tree*-node. The fact that both visualized partitions are SCCs means that all ecological processes *within* each partition are reversible (represented by some finite sequence of GT steps inside that partition).



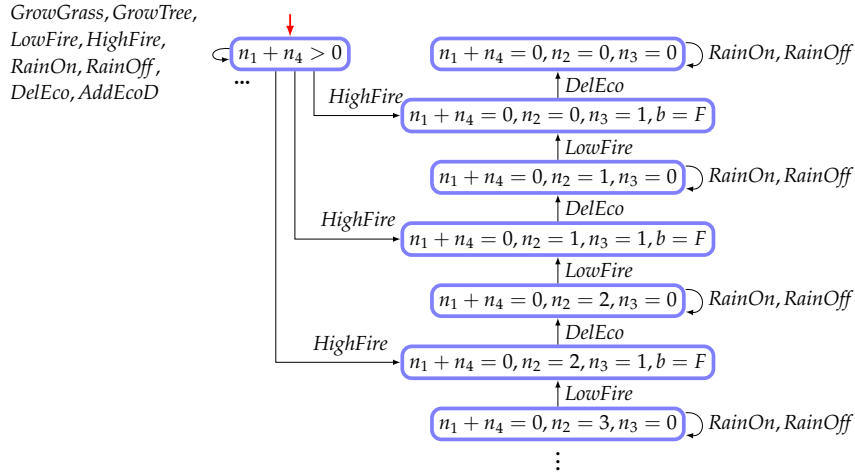
(a) The finite state space for the GT model \mathcal{S}_2 . The states are visualized using filled rectangles. The transitions between states by rule applications are visualized using labeled arrows. The six green states represent graphs with two *Tree*-nodes, the eight orange states represent graphs with one *Tree*-node, and the six blue states represent graphs with no *Tree*-nodes. The initial graph G_0 is marked with a vertical red arrow. The name of a state of the form $n_1n_2n_3n_4b$ captures the information about the structure of the underlying graph G_i using the following five variables: (i) the number n_1 of *Eco*-nodes in G_i with a connected *Tree*- and a connected *Grass*-node, (ii) the number n_2 of *Eco*-nodes in G_i with no connected *Tree*- and a connected *Grass*-node, (iii) the number n_3 of *Eco*-nodes in G_i with no connected *Tree*- and no connected *Grass*-node, (iv) the number n_4 of *Eco*-nodes in G_i with a connected *Tree*- and no connected *Grass*-node, and (v) the Boolean value $b = T$ if the *Rain*-node in G_i has an *on*-loop and the Boolean value $b = F$ if the *Rain*-node in G_i has an *off*-loop.



(b) The finite SSC-based merged state space for the GT model \mathcal{S}_2



(c) The finite SCC-based merged state space for the GT model \mathcal{S}_U



(d) A finite part of the infinite SCC-based merged state space for the GT model \mathcal{S}_D

Figure 3.1: Analysis results for the GT models \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D . The obtained SCCs are visualized using unfilled blue rectangles. The transitions between SCCs by rule applications are visualized using labeled arrows. The SCC containing the initial graph G_0 is marked with a vertical red arrow. The names of SCCs capture the information about the contained states using conditions over the variables n_1, n_2, n_3, n_4 , and b from Figure 3.1a.

The SCC-based merged state space in Figure 3.1b is obtained from the state space in Figure 3.1a by checking mutual reachability between each two states of the computed state space. We formulated the property of mutual reachability using CTL properties and checked the satisfaction of these CTL properties using the tool GROOVE. Since the number of *Tree*-nodes is captured by the expression $n_1 + n_4$ according to the abstraction above, the two SCCs are named $n_1 + n_4 > 0$ representing graphs that have at least one *Tree*-node and $n_1 + n_4 = 0$ representing graphs that have no *Tree*-nodes. Figure 3.1b shows again that the removal of the last *Tree*-node (by means of the rule *HighFire*) is an irreversible step from the left SCC to the right SCC.

3.2 Analysis of GT Model \mathcal{S}_U

The following theorem captures the analysis results for the GT model \mathcal{S}_U .

Theorem 2 (State Space Induced by \mathcal{S}_U). The GT model \mathcal{S}_U generates an infinite state space. The SCC-based merging of this infinite state space results in the state space given in Figure 3.1c containing two SCCs.

The state space of the GT model \mathcal{S}_U is infinite in contrast to that of \mathcal{S}_2 because an unlimited number of *Eco*-nodes can be added using the rule *AddEcoU*. Since the infinite state space of \mathcal{S}_U is not manageable by fully-automatic analysis techniques, as a first step for analysis, we adapt the rule *AddEcoU*. This adaptation only permits the addition of *Eco*-nodes up to a fixed threshold N resulting in a finite state space. It turns out that for thresholds such as $N = 4$ the state space can be partitioned into two SCCs as for \mathcal{S}_2 containing states with at least one *Tree*-node and states with no *Tree*-nodes.

However, the use of such thresholds may significantly change the structure of the state space in terms of resulting SCCs. Hence, as a second step for analysis, we manually analyze the infinite state space generated by \mathcal{S}_U . The result of this manual analysis is given in Theorem 2 stating that the infinite state space of \mathcal{S}_U induces the same two SCCs as suggested by the threshold-based state space generation. The proof of Theorem 2 is given in section A.3.

3.3 Analysis of GT Model \mathcal{S}_D

The following theorem captures the analysis results for the GT model \mathcal{S}_D .

Theorem 3 (State Space Induced by \mathcal{S}_D). The GT model \mathcal{S}_D generates an infinite state space. The SCC-based merging of this infinite state space results in the state space given in Figure 3.1d containing infinitely many SCCs.

The state space of the GT model \mathcal{S}_D is infinite as for \mathcal{S}_U . However, in \mathcal{S}_D , the addition of *Eco*-nodes using the rule *AddEcoD* requires the existence of *Eco*-nodes with a connected *Grass*- and *Tree*-node. Hence, as soon as all *Tree*-nodes are removed

by the rule *HighFire*, the rule *AddEcoD* is continuously disabled. As a consequence, the only remaining GT steps change the season between rainy and dry using the rules *RainOn* and *RainOff*, remove *Grass*-nodes using the rule *LowFire*, and remove *Eco*-nodes using the rule *DelEco*. Since the rules *LowFire* and *DelEco* are not reversible, there is an infinite chain of SCCs as indicated in Figure 3.1d where either the number of *Eco*-nodes or the number of *Grass*-nodes decreases from SCC to SCC.

These observations have again been obtained using the threshold-based approach as for \mathcal{S}_U . To ensure that the use of thresholds did not significantly change the structure of the induced SCC-based merged state space of \mathcal{S}_D , we employed manual analysis, which results in Theorem 3 stating that the observations above are also valid for the infinite state space of \mathcal{S}_D . The proof of Theorem 3 is given in section A.3.

4 Discussion

The main goal of this paper was to evaluate whether the formal technique of GT is suitable for the modeling and analysis of meta-ecosystems.

GT as a domain agnostic formal technique has been already applied in different fields of biology including cellular and molecular biology (e.g. [1, 20, 41]) but, to the best of our knowledge, this paper shows its applicability in the field of ecology for the first time.

In this paper, we have considered GTs in which the graphs representing states of meta-ecosystems are discrete (i.e., not featuring continuous variables) and the transitions between these states are untimed (i.e., the passage of time is not modeled), discrete (i.e., there are countably many descriptions of transitions between two states), atomic (i.e., steps cannot be interrupted half-way), asynchronous/non-deterministic (i.e., multiple transitions possibly using different rules may be possible from the same state), and nonprobabilistic (i.e., no relative likelihood is assigned to possible transitions from a single state). In contrast to the less expressive formalism of Petri nets that has been investigated previously ([16]), the use of such GTs enables the modeling of space structure dynamics in which the number of ecosystem patches in the meta-ecosystem may grow unboundedly.

To achieve our main goal, we have defined in section 2.2 three GT models for three different savanna meta-ecosystems, analyzed them with respect to their space structure dynamics and ecological dynamics in chapter 3, and we now compare the obtained results to related model-based analysis results from the literature. In particular, we have analyzed for these three GT models two different kinds of qualitative properties. Firstly, we have analyzed state reachability meaning that the modeled meta-ecosystem can reach a certain state according to the GT model at hand. This state reachability served as the basis for the validation of the GT model (i.e., it helped to determine whether the transitions in the GT model correspond to the transitions expected from the modeled meta-ecosystem) as well as for the comprehension of unpredicted dynamics governed by the rules of the GT model (i.e., the desertification effect induced by the rule for the grass dispersal (*AddEcoD*) in the third GT model \mathcal{S}_D , see below). Secondly, we have analyzed transition irreversibility (meaning that when a transition from a first state to a second state cannot be undone by any sequence of further transitions leading back to the first state) including an analysis of the transition sequences leading to a *tipping point*. The result of such an analysis of transition irreversibility is assumed to be useful for managers wishing to be notified of situations in which they want to act sparingly and efficiently on the system to timely counter certain dynamics via human intervention before unwantedly reaching the related tipping point ([7, 55]). In particular, in the considered GT models, we have analyzed the effect of two kinds of fires on the vegetation for which the

effect varies among these GT models due to the considered ecological processes for the growth of grass. Moreover, considering the distinguishing feature of unbounded space structure dynamics, we subsequently discuss the results of our evaluation with respect to the three defined GT models and assess how qualitative properties of these GT models can be derived using formal analysis techniques.

The first GT model \mathcal{S}_2 from section 3.1 describes a savanna meta-ecosystem with a constant number of two ecosystem patches (i.e., a meta-ecosystem with a rigid space structure dynamics) and induces a finite state space. For the GT model \mathcal{S}_2 , we have assumed that seeds of trees can only spread over short distances by requiring that the growth of trees on an ecosystem patch is impossible without trees on neighbor ecosystem patches. Consequently, a high intensity fire destroying the last tree in a meta-ecosystem is irreversible in \mathcal{S}_2 leading to an attractor given by a set of states representing grassland and desert states. This irreversibility (as obtained in our analysis of \mathcal{S}_2) indicates that the ecological dynamics of the modeled savanna meta-ecosystem is highly sensitive to high intensity fires. This result is in good agreement with previous studies (e.g. [49, 52]), which show the importance of the dispersal of trees for maintaining meta-ecosystems with high biodiversity.

To obtain these analysis results for the GT model \mathcal{S}_2 and to obtain hints for human intervention, we have performed an exhaustive computation of all states that are reachable from the initial state resulting in a finite state space to allow for the analysis of qualitative properties (see Figure 3.1a). State reachability properties (e.g. the expected possibility of the destruction of all trees in the meta-ecosystem) can then be analyzed either by inspecting each state and transition by hand or by applying standard logics such as CTL (which supports the checking of further properties on the entire state space) for automation. Transition irreversibility properties (given in \mathcal{S}_2 by a high intensity fire destroying the last tree in the entire meta-ecosystem) can be identified by constructing an additional finite SCC-based merged state space from the original state space where states that are mutually reachable (i.e., the transitions between them are reversible) are merged (see Figure 3.1b). The transitions between different states in the finite SCC-based merged state space then characterize the irreversible transitions of the original state space. Additionally, once such irreversible transitions are identified, transition sequences leading to these tipping points can be identified and inspected. In \mathcal{S}_2 , such an inspection results in the observation that the distance (in terms of the number of transitions) to the tipping point grows with (a) the number of ecosystem patches with trees, (b) the number of ecosystem patches with trees but without grass, and (c) the meta-ecosystem being in the rainy season. This observation indicates that (a) tree planting (to compensate for tree destruction via high intensity fires), (b) removal of grass from ecosystem patches with trees (to prevent high intensity fires where grass serves as a combustive agent), and (c) artificial watering (to prevent high intensity fires during the dry season) are viable human interventions according to the GT model. However, some of these interventions may not be feasible due to resource limitations such as economic costs or the availability of water. Note that artificial tree planting in \mathcal{S}_2 is even then successful when all trees have been destroyed because the destruction of all trees did not result in a further irreversible collapse of other ecological components. The iden-

tified tipping point and the transition sequences leading to it are in agreement with other studies (e.g. [45, 48]). However, in our approach, such transition sequences leading to the tipping point can be explicitly obtained from the state space of the GT model.

In the GT model \mathcal{S}_2 , the restriction of the space structure dynamics to a constant number of two ecosystem patches allows for a finite state space. Subsequently, we discuss the two GT models \mathcal{S}_U and \mathcal{S}_D with space structure dynamics where an unbounded number of ecosystem patches may be observed. This space structure dynamics is significant since it results in an infinite state space in both cases, which requires an adaptation of the successful analysis procedure carried out for \mathcal{S}_2 . Also note that such space structure dynamics can be modeled using GT since it does not require a fixed space structure on the reachable graphs in contrast to e.g. DS^2 ([20]) and the formalism of Petri nets ([39, 40]).

The second GT model \mathcal{S}_U from section 3.2 describes a savanna meta-ecosystem with an unbounded number of ecosystem patches (i.e., a meta-ecosystem where additional ecosystem patches can appear due to ecological processes) and induces therefore an infinite state space. The grass related ecological processes considered in \mathcal{S}_U differ from that in \mathcal{S}_2 as follows. Firstly, in \mathcal{S}_U , we have allowed that ecosystem patches without grass and without trees can disappear from the meta-ecosystem since such empty ecosystem patches are of no further interest. Secondly, as for \mathcal{S}_2 , we have assumed for \mathcal{S}_U that seeds of grass can successfully spread over close ranges during the rainy season to nearby ecosystem patches, which may result in \mathcal{S}_U in such nearby ecosystem patches appearing in the meta-ecosystem. Moreover, the spreading of trees between ecosystem patches in \mathcal{S}_U follows the same ecological processes as in \mathcal{S}_2 . Consequently, as for \mathcal{S}_2 , a high intensity fire destroying the last tree in a meta-ecosystem is irreversible in \mathcal{S}_U . Such an irreversible transition in \mathcal{S}_U leads to a more complex attractor compared to \mathcal{S}_2 containing states with any number of grassland ecosystem patches.

According to our analysis results, the GT model \mathcal{S}_U suggests the same human interventions as \mathcal{S}_2 as discussed in the following where we had to rely on the manual process of mathematical reasoning due to the infinite state space of \mathcal{S}_U . We have demonstrated that the underlying structures of the state spaces induced by \mathcal{S}_2 and \mathcal{S}_U (in terms of the SCC-based merged state spaces) coincide (compare Figure 3.1b and Figure 3.1c). This means that the number of ecosystem patches involved and the space structure dynamics of both GT models differed but both GT models exhibited the same two overall stabilities (in particular, one with trees and one without trees). Consequently, we have observed in \mathcal{S}_U the same tipping point as in \mathcal{S}_2 given by the irreversible transition enabling the destruction of the last tree by a high intensity fire during the dry season. Further analysis of the transition sequences leading to this tipping point also shows that, albeit there are states in the state space where a large number of high intensity fires is required to reach this tipping point, there is a state where a single high intensity fire is irreversible as in \mathcal{S}_2 and \mathcal{S}_U therefore suggesting the same human interventions as for \mathcal{S}_2 . We conclude that GT models with infinite space structure dynamics may be tractable using manual mathematical reasoning

resulting in suitable finite representations such as the one given by the SCC-based merged state space of \mathcal{S}_U , which demonstrates transition irreversibility.

The space structure dynamics in the GT model \mathcal{S}_U allows for a finite SCC-based merged state space. Subsequently, we discuss the GT model \mathcal{S}_D with space structure dynamics where not only the state space but also the SCC-based merged state space are infinite. For this more complex case, we evaluate by means of the GT model \mathcal{S}_D how to adapt the successful analysis procedure carried out for \mathcal{S}_U .

The third GT model \mathcal{S}_D from section 3.3 describes a savanna meta-ecosystem with an unbounded number of ecosystem patches as for \mathcal{S}_U and therefore also induces an infinite state space. In \mathcal{S}_D , we have adapted the ecological process for the spreading of grass from \mathcal{S}_U by requiring that grass may only spread to nearby ecosystem patches during the rainy season from ecosystem patches with trees. Moreover, the spreading of trees between ecosystem patches in \mathcal{S}_D follows the same ecological processes as in \mathcal{S}_2 . Consequently, as for \mathcal{S}_2 and \mathcal{S}_U , a high intensity fire destroying the last tree in a meta-ecosystem is irreversible in \mathcal{S}_D . Such an irreversible transition in \mathcal{S}_D leads to an infinite number of attractors each containing states with a certain number of grassland/desert ecosystem patches including the attractor containing a meta-ecosystem state without ecosystem patches representing a desertified savanna (see [26, 27, 50]).

According to our analysis results, the GT model \mathcal{S}_D suggests the same human interventions as \mathcal{S}_2 as discussed in the following where we had to rely as for \mathcal{S}_U on the manual process of mathematical reasoning due to the infinite state space of \mathcal{S}_D . We have demonstrated that the same transitions are irreversible in \mathcal{S}_D as in \mathcal{S}_2 and \mathcal{S}_U even though the induced SCC-based merged state space of \mathcal{S}_D is infinite and therefore fundamentally different to the finite SCC-based merged state spaces of \mathcal{S}_2 and \mathcal{S}_U (compare Figure 3.1b, Figure 3.1c, and Figure 3.1d). However, note that the state in the SCC-based merged state space of \mathcal{S}_U representing meta-ecosystem states without trees can be understood to correspond to the infinite number of states of the SCC-based merged state space of \mathcal{S}_D representing meta-ecosystem states without trees. These latter states are not merged for \mathcal{S}_D because some transitions for the spreading of grass from \mathcal{S}_U are not allowed in \mathcal{S}_D where the existence of trees is an additional requirement for the spreading of grass. Consequently, even though the number of states in the SCC-based merged state spaces differs between \mathcal{S}_D on the one hand and \mathcal{S}_2 and \mathcal{S}_U on the other hand, we have observed in \mathcal{S}_D the same tipping point as in \mathcal{S}_2 and \mathcal{S}_U given by the irreversible transition enabling the destruction of the last tree by a high intensity fire during the dry season. Further analysis reveals that the transition sequences leading to this tipping point correspond closely to those for \mathcal{S}_2 and \mathcal{S}_U therefore indicating the same human interventions. We conclude that GT models with infinite space structure dynamics may be still tractable using manual mathematical reasoning even when they induce infinite SCC-based merged state spaces.

The strong similarities between \mathcal{S}_2 , \mathcal{S}_U , and \mathcal{S}_D result from the fact that different intra-patch and inter-patch dynamics are defined independently using GT rules. This indicates that the formal technique of GT is well suited for the modeling of meta-ecosystems where even larger numbers of ecological processes are to be considered.

In comparison, other formalisms such as Petri nets would require during the manual modeling process an error-prone composition of the various transitions that may be carried out at the same state. In the future, we want to further evaluate the applicability of GT by incorporating more complex and realistic intra-patch and inter-patch dynamics observed in the past such as that of the green sahara during holocene ([9, 56]).

As described above, we have evaluated the validity of considered GT models by (a) inspecting the finite state space manually, (b) using automatic support for checking CTL properties (such as state reachability), or (c) mathematically characterizing the reachable states (for GT models with infinite state spaces). We have observed that the considered GT models show ecological dynamics similar to those modeled in other studies (e.g. [28, 31, 52]), which do not use GT and do not consider space structure dynamics, and observed through long term monitoring or paleoecological reconstructions. However, it also turned out that seemingly minor modifications such as adding, removing, or changing a single rule may drastically affect the dynamics of a GT model as can be observed by the drastically different behavior of the GT models \mathcal{S}_U and \mathcal{S}_D resulting from the replacement of the rule *AddEcoU* by the rule *AddEcoD*. This observation may be understood as a threat that a given GT model adequately captures the dynamics of the considered meta-ecosystem but it also demonstrates the feasibility of human interventions that can also be integrated into the GT model via additional rules. Nevertheless, a further systematic analysis of the sensitivity of GT models such as \mathcal{S}_U against reasonable adaptations is called for.

As mentioned before, in our approach we have considered standard GTSs where seasonal timing, durations of ecological processes, and relatively likelihoods of transitions are not captured. We have decided against the use of more advanced GTSs (e.g. [3, 21, 38, 47]) supporting such parametrizations in this paper since (a) suitable values for such parametrizations are often difficult to obtain, (b) the qualitative analysis of GT models considered in this paper can be better compared to related work involving qualitative analysis using formalisms that are not based on GT, (c) the qualitative analysis is of no less importance since it is usually simpler compared to quantitative analysis where results cannot be always obtained, and (d) the ecological dynamics of the GT models considered in this paper may serve as a starting point for the integration of time and probabilities into the GT models since these would be used to rule out certain transitions of GT models or to make these transitions less likely once certain transition sequences (e.g. an infinite uninterrupted switching between dry and rainy season) are observed in the GT model. An obvious consequence of omitting a parametrization using time and probabilities is that the properties referring to time, probabilities, or quantities (e.g. the probability that all trees have disappeared within 10 years) cannot be analyzed. With all these points in mind, once suitable parametrizations can be determined via long term monitoring of meta-ecosystems or paleoecological reconstructions, we want to explore the use of such more complex GTSs and the available analysis approaches for the field of ecology.

While not being critical for the GT model \mathcal{S}_2 with a small state space, the efficiency of the tools such as GROOVE ([17]) and AUTOGRAPH ([47]) for generating such state

spaces may become critical in the future when incorporating many ecological processes. Hence, further technical advancements may be required, which seem to have not yet been in the focus of the GT community.

Moreover, the analysis of GT models describing meta-ecosystems requires dedicated domain-specific tools tailored for the needs of ecologists. ([5]). Such tools should e.g. support the construction of the SCC-based merged state space from a given state space but also provide at least partial automation of the mathematical reasoning carried out for \mathcal{S}_U and \mathcal{S}_D with their infinite state spaces. Also, while CTL has proven to be useful in this paper for the considered GT models without any further parametrization, we also need to evaluate the usefulness of other logics such as PTCTL ([33]) or MTGL ([22]) for the specification and analysis of GT models describing meta-ecosystems that capture aspects of time or probabilities.

In conclusion, we propose to use the formal technique of GT for the modeling of (savanna) meta-ecosystems exhibiting besides ecological dynamics also space structure dynamics and their analysis with respect to qualitative properties such as transition irreversibility. In particular, the space structure dynamics cannot be adequately modeled using already existing approaches based on Petri nets (e.g. [16]), differential equations (e.g. [31, 35, 52]), or IBMs (e.g. [11]). We believe that GT models provide a complementary avenue to that of existing approaches to rigorously study ecological phenomena.

4.1 Acknowledgments

We warmly thank Franck Pommereau, Colin Thomas from the IBISC, University of Evry and Maximilian Cosme from the AMAP, University of Montpellier for valuable discussions on a preliminary version of this paper. We also thank Malo Y. Bourget from the CNRS Moulis for the proof-reading of the paper and his support during this study.

Bibliography

- [1] T. Aittokallio and B. Schwikowski. “Graph-based methods for analysing networks in cell biology”. In: *Briefings in bioinformatics* 7.3 (2006), pages 243–255.
- [2] C. Baier and J.-P. Katoen. *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, 2008. ISBN: 978-0-262-02649-9.
- [3] B. Becker and H. Giese. “On Safe Service-Oriented Real-Time Coordination for Autonomous Vehicles”. In: *11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2008), 5-7 May 2008, Orlando, Florida, USA*. IEEE Computer Society, 2008, pages 203–210. ISBN: 978-0-7695-3132-8. DOI: 10.1109/ISORC.2008.13.
- [4] W. J. Bond, G. F. Midgley, and F. I. Woodward. “The importance of low atmospheric CO₂ and fire in promoting the spread of grasslands and savannas”. In: *Global Change Biology* 9.7 (2003), pages 973–982.
- [5] C. Chaouiya, H. Kludel, and F. Pommereau. “A modular, qualitative modeling of regulatory networks using Petri nets”. In: *Modeling in Systems Biology*. Springer, 2011, pages 253–279.
- [6] E. M. Clarke and E. A. Emerson. “Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic”. In: *Logics of Programs, Workshop, Yorktown Heights, New York, May 1981*. Edited by D. Kozen. Volume 131. Lecture Notes in Computer Science. Springer, 1981, pages 52–71. ISBN: 3-540-11212-X. DOI: 10.1007/BFb0025774.
- [7] D. H. M. Cumming. “Living off ‘biodiversity’: whose land, whose resources and where?” In: *Environment and Development Economics* 4.2 (1999), pages 203–236.
- [8] J. M. Dambacher, H. W. Li, and P. A. Rossignol. “Qualitative predictions in model ecosystems”. In: *Ecological Modelling* 161.1-2 (2003), pages 79–93.
- [9] P. Demenocal, J. Ortiz, T. Guilderson, J. Adkins, M. Sarnthein, L. Baker, and M. Yarusinsky. “Abrupt onset and termination of the African Humid Period: rapid climate responses to gradual insolation forcing”. In: *Quaternary science reviews* 19.1-5 (2000), pages 347–361.
- [10] J. Dyck and H. Giese. “k-Inductive Invariant Checking for Graph Transformation Systems”. In: *Graph Transformation - 10th International Conference, ICGT 2017, Held as Part of STAF 2017, Marburg, Germany, July 18-19, 2017, Proceedings*. Edited by J. de Lara and D. Plump. Volume 10373. Lecture Notes in Computer Science. Springer, 2017, pages 142–158. ISBN: 978-3-319-61469-4. DOI: 10.1007/978-3-319-61470-0_9.
- [11] P. Edelaar, R. Jovani, and I. Gomez-Mestre. “Should I change or should I go? Phenotypic plasticity and matching habitat choice in the adaptation to environmental heterogeneity”. In: *The American Naturalist* 190.4 (2017), pages 506–520.
- [12] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. Springer, 2006.
- [13] R. T. Forman. *Land Mosaics: The Ecology of Landscapes and Regions 1995*. Springer, 2014.

- [14] R. T. T. Forman and M. Godron. "Patches and structural components for a landscape ecology". In: *BioScience* 31.10 (1981), pages 733–740.
- [15] C. Gaucherel, F. Houllier, D. Auclair, and T. Houet. "Dynamic landscape modelling: the quest for a unifying theory". In: *Living reviews in landscape Research* 8.2 (2013), pages 5–31.
- [16] C. Gaucherel and F. Pommereau. "Using discrete systems to exhaustively characterize the dynamics of an integrated ecosystem". In: *Methods in Ecology and Evolution* accepted (2019).
- [17] A. H. Ghamarian, M. de Mol, A. Rensink, E. Zambon, and M. Zimakova. "Modelling and analysis using GROOVE". In: *Int. J. Softw. Tools Technol. Transf.* 14.1 (2012), pages 15–40. DOI: 10.1007/s10009-011-0186-x.
- [18] J. L. Giavitto, H. Kludel, and F. Pommereau. "Integrated regulatory networks (IRNs): Spatially organized biochemical modules". In: *Theoretical Computer Science* 431 (2012), pages 219–234. DOI: 10.1016/j.tcs.2011.12.054.
- [19] J.-L. Giavitto and O. Michel. "Modeling the topological organization of cellular processes". In: *BioSystems* 70.2 (2003), pages 149–163.
- [20] J.-L. Giavitto, O. Michel, J. Cohen, and A. Spicher. "Computations in space and space in computations". In: *International Workshop on Unconventional Programming Paradigms*. Springer, 2004, pages 137–152.
- [21] H. Giese. "Modeling and Verification of Cooperative Self-adaptive Mechatronic Systems". In: *Reliable Systems on Unreliable Networked Platforms - 12th Monterey Workshop 2005, Laguna Beach, CA, USA, September 22-24, 2005. Revised Selected Papers*. Edited by F. Kordon and J. Sztipanovits. Volume 4322. Lecture Notes in Computer Science. Springer, 2005, pages 258–280. ISBN: 978-3-540-71155-1. DOI: 10.1007/978-3-540-71156-8_14.
- [22] H. Giese, M. Maximova, L. Sakizloglou, and S. Schneider. "Metric Temporal Graph Logic over Typed Attributed Graphs". In: *Fundamental Approaches to Software Engineering - 22nd International Conference, FASE 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*. Edited by R. Hähnle and W. M. P. van der Aalst. Volume 11424. Lecture Notes in Computer Science. Springer, 2019, pages 282–298. ISBN: 978-3-030-16721-9. DOI: 10.1007/978-3-030-16722-6_16.
- [23] I. Gounand, E. Harvey, C. J. Little, and F. Altermatt. "Meta-ecosystems 2.0: rooting the theory into the field". In: *Trends in Ecology and Evolution* 33.1 (2018), pages 36–46.
- [24] G. H. Griffiths. "Land mosaics: the Ecology of Landscapes and Regions". In: *Applied Geography* 1.18 (1998), pages 98–99.
- [25] I. Hanski. "Metapopulation Dynamics: From Concepts and Observations to Predictive Models". In: *Metapopulation Biology*. Edited by I. Hanski and M. E. Gilpin. San Diego: Academic Press, 1997, pages 69–91. ISBN: 978-0-12-323445-2. DOI: <https://doi.org/10.1016/B978-012323445-2/50007-9>.
- [26] C. Hély, S. Alleaume, R. J. Swap, H. H. Shugart, and C. O. Justice. "SAFARI-2000 characterization of fuels, fire behavior, combustion completeness, and emissions from experimental burns in infertile grass savannas in western Zambia". In: *Journal of Arid Environments* 54.2 (2003), pages 381–394.
- [27] C. Hély, P. Braconnot, J. Watrin, and W. Zheng. "Climate and vegetation: simulating the African humid period". In: *Comptes Rendus Geoscience* 341.8-9 (2009), pages 671–688.

- [28] C. Hély, L. Bremond, S. Alleaume, B. Smith, M. T. Sykes, and J. Guiot. “Sensitivity of African biomes to changes in the precipitation regime”. In: *Global Ecology and Biogeography* 15.3 (2006), pages 258–270.
- [29] R. D. Holt. “Ecology at the mesoscale: the influence of regional processes on local communities”. In: *Species diversity in ecological communities* (1993), pages 77–88.
- [30] R. B. Jackson, J. L. Banner, E. G. Jobbágy, W. T. Pockman, and D. H. Wall. “Ecosystem carbon loss with woody plant invasion of grasslands”. In: *Nature* 418.6898 (2002), page 623.
- [31] S. Kéfi, V. Guttal, W. A. Brock, S. R. Carpenter, A. M. Ellison, V. N. Livina, D. A. Seekell, M. Scheffer, E. H. van Nes, and V. Dakos. “Early warning signals of ecological transitions: methods for spatial patterns”. In: *PLoS one* 9.3 (2014), e92097.
- [32] B. König, D. Nolte, J. Padberg, and A. Rensink. “A Tutorial on Graph Transformation”. In: *Graph Transformation, Specifications, and Nets - In Memory of Hartmut Ehrig*. Edited by R. Heckel and G. Taentzer. Volume 10800. Lecture Notes in Computer Science. Springer, 2018, pages 83–104. DOI: 10.1007/978-3-319-75396-6_5.
- [33] M. Z. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. “Automatic verification of real-time systems with discrete probability distributions”. In: *Theor. Comput. Sci.* 282.1 (2002), pages 101–150. DOI: 10.1016/S0304-3975(01)00046-9.
- [34] M. A. Leibold, M. Holyoak, N. Mouquet, P. Amarasekare, J. M. Chase, M. F. Hoopes, R. D. Holt, J. B. Shurin, R. Law, D. Tilman, M. Loreau, and A. Gonzalez. “The meta-community concept: a framework for multi-scale community ecology”. In: *Ecology letters* 7.7 (2004), pages 601–613.
- [35] M. Loreau, N. Mouquet, and R. D. Holt. “Meta-ecosystems: A theoretical framework for a spatial ecosystem ecology”. In: *Ecology Letters* 6.8 (2003), pages 673–679. DOI: 10.1046/j.1461-0248.2003.00483.x.
- [36] D. Machado, R. S. Costa, M. Rocha, I. Rocha, B. Tidor, and E. C. Ferreira. “A critical review on modelling formalisms and simulation tools in computational biosystems”. In: *International Work-Conference on Artificial Neural Networks*. Springer, 2009, pages 1063–1070.
- [37] F. Massol, D. Gravel, N. Mouquet, M. W. Cadotte, T. Fukami, and M. A. Leibold. “Linking community and ecosystem dynamics through spatial ecology”. In: *Ecology letters* 14.3 (2011), pages 313–323.
- [38] M. Maximova, H. Giese, and C. Krause. “Probabilistic timed graph transformation systems”. In: *J. Log. Algebr. Meth. Program.* 101 (2018), pages 110–131. DOI: 10.1016/j.jlamp.2018.09.003.
- [39] F. Pommereau. “Algebras of coloured Petri nets”. In: *LAP LAMBERT Academic Publishing* (2010).
- [40] W. Reisig. *Petri nets: an introduction*. Volume 4. Springer Science & Business Media, 2012.
- [41] F. Rosselló and G. Valiente. “Graph transformation in molecular biology”. In: *Formal Methods in Software and Systems Modeling*. Springer, 2005, pages 116–133.
- [42] E. J. Rykiel Jr. “Artificial intelligence and expert systems in ecology and natural resource management”. In: *Ecological Modelling* 46.1-2 (1989), pages 3–8.
- [43] M. Sankaran, J. Ratnam, and N. P. Hanan. “Tree–grass coexistence in savannas revisited—insights from an examination of assumptions and mechanisms invoked in existing models”. In: *Ecology letters* 7.6 (2004), pages 480–490.

- [44] M. Sankaran et al. "Determinants of woody cover in African savannas". In: *Nature* 438.7069 (2005), page 846.
- [45] M. Scheffer, S. R. Carpenter, V. Dakos, and E. H. van Nes. "Generic Indicators of Ecological Resilience: Inferring the Chance of a Critical Transition". In: *Annual Review of Ecology, Evolution, and Systematics* 46.1 (2015), pages 145–167. ISSN: 1543-592X. DOI: 10.1146/annurev-eco\sys-112414-054242.
- [46] S. Schneider, J. Dyck, and H. Giese. "Formal Verification of Invariants for Attributed Graph Transformation Systems Based on Nested Attributed Graph Conditions". In: *Graph Transformation - 13th International Conference, ICGT 2020, Held as Part of STAF 2020, Bergen, Norway, June 25-26, 2020, Proceedings*. Edited by F. Gadducci and T. Kehrer. Volume 12150. Lecture Notes in Computer Science. Springer, 2020, pages 257–275. DOI: 10.1007/978-3-030-51372-6_15.
- [47] S. Schneider, M. Maximova, L. Sakizloglou, and H. Giese. "Formal Testing of Timed Graph Transformation Systems using Metric Temporal Graph Logic". In: *Int. J. Softw. Tools Technol. Transf.* (2020). (accepted).
- [48] A. C. Staver. "Prediction and scale in savanna ecosystems". In: *New Phytologist* 219.1 (2018), pages 52–57.
- [49] A. C. Staver, S. Archibald, and S. A. Levin. "The global extent and determinants of savanna and forest as alternative biome states". In: *Science* 334.6053 (2011), pages 230–232. DOI: 10.1126/science.1210465.
- [50] A. C. Staver and S. A. Levin. "Integrating Theoretical Climate and Fire Effects on Savanna and Forest Systems". In: *The American Naturalist* 180.2 (2012), pages 211–224. ISSN: 0003-0147. DOI: 10.1086/666648.
- [51] D. Tilman and P. Kareiva. *Spatial ecology: the role of space in population dynamics and interspecific interactions*. Volume 30. Monographs in Population Biology. Princeton University Press, 2018.
- [52] D. Touboul, A. C. Staver, J. David, A. Carla, and S. Asher. "Correction for Touboul et al., On the complex dynamics of savanna landscapes". In: *Proceedings of the National Academy of Sciences* 115.31 (2018), E7457–E7457. ISSN: 0027-8424. DOI: 10.1073/pnas.1810993115.
- [53] M. G. Turner, R. H. Gardner, and R. V. O'Neill. *Landscape ecology in theory and practice*. Volume 401. Springer, 2001.
- [54] P. H. Verburg, W. Soepboer, A. Veldkamp, R. Limpiada, V. Espaldon, and S. S. A. Mastura. "Modeling the spatial dynamics of regional land use: the CLUE-S model". In: *Environmental management* 30.3 (2002), pages 391–405.
- [55] B. Walker, C. S. Holling, S. R. Carpenter, and A. Kinzig. "Resilience, adaptability and transformability in social–ecological systems". In: *Ecology and society* 9.2 (2004).
- [56] J. Watrin, A.-M. Lézine, and C. Hély. "Plant migration and plant communities at the time of the "green Sahara"". In: *Comptes Rendus Geoscience* 341.8-9 (2009), pages 656–670.

A Supplementary Material

A.1 Additional Details on Typed Graphs and Graph Transformation

In this appendix, we give more details on the formal technique of GT based on typed graphs following the standard handbook ([12]).

First, we introduce so-called *plain graphs*, which are then used to define typed graphs. A plain graph consists of nodes that are connected by edges. Plain graphs considered here are allowed to have parallel edges (i.e., two or more edges can have the same source and target node). While we do not use parallel edges in the GT models from section 2.2, this requirement is generally assumed to ensure that pushouts (see Definition 19 below) can always be constructed.

Definition 12 (Plain Graph). If N is a finite set of nodes, E is a finite set of edges, $s : E \rightarrow N$ maps each edge to its source node, and $t : E \rightarrow N$ maps each edge to its target node, then $G = (N, E, s, t)$ is a *plain graph*.

Plain graph morphisms between plain graphs are used to detect how a plain graph G_1 can be embedded into another plain graph G_2 . This is achieved by mapping each node and edge of G_1 to one node and edge in G_2 . If both components of the plain graph morphism are injective, then G_1 is a subgraph of G_2 . Each plain graph morphism must, however, preserve the graph structure, which means that the mapping of nodes and edges must be compatible (i.e., the source and target nodes of an edge e must be mapped to the source and target nodes of the mapping of e).

Definition 13 (Plain Graph Morphism). If $G_1 = (N_1, E_1, s_1, t_1)$ and $G_2 = (N_2, E_2, s_2, t_2)$ are plain graphs, $m_N : N_1 \rightarrow N_2$ maps each node of G_1 to a node of G_2 , $m_E : E_1 \rightarrow E_2$ maps each edge of G_1 to an edge of G_2 , $s_2(m_E(e_1)) = m_N(s_1(e_1))$ and $t_2(m_E(e_1)) = m_N(t_1(e_1))$ for each edge $e_1 \in E_1$ (which means that if m_E maps an edge e_1 to an edge e_2 , then m_N must map the source/target of e_1 to the source/target of e_2), then $m = (m_N, m_E)$ is a *plain graph morphism* from G_1 to G_2 , also written $m : G_1 \rightarrow G_2$.

$$\begin{array}{ccc} E_1 & \begin{array}{c} \xrightarrow{s_1} \\ \xrightarrow{t_1} \end{array} & N_1 \\ m_E \downarrow & & \downarrow m_N \\ E_2 & \begin{array}{c} \xrightarrow{s_2} \\ \xrightarrow{t_2} \end{array} & N_2 \end{array}$$

The *composition of two plain graph morphisms* $m : G_1 \rightarrow G_2$ and $m' : G_2 \rightarrow G_3$ is defined componentwise using the composition operation for functions. It results in a plain graph morphism from G_1 to G_3 that specifies how G_1 is embedded into G_3 .

Definition 14 (Composition of Plain Graph Morphisms). If $m = (m_N, m_E) : G_1 \rightarrow G_2$ and $m' = (m'_N, m'_E) : G_2 \rightarrow G_3$ are plain graph morphisms, then $m' \circ m = (m'_N \circ m_N, m'_E \circ m_E) : G_1 \rightarrow G_3$ is the *composition* of m and m' .

$$\begin{array}{ccccc} G_1 & \xrightarrow{m} & G_2 & \xrightarrow{m'} & G_3 \\ & \searrow & & \nearrow & \\ & & & & m' \circ m \end{array}$$

The notion of *typed graphs* relies on that of plain graphs. A typed graph consists of a plain graph G that is typed over a type graph TG . The typing of G is represented by a plain graph morphism m from G to TG . Note that this typing morphism m is in general not injective since all graph elements of the same type are mapped to the same graph element of TG .

Definition 15 (Typed Graph). If G is a plain graph, TG is a plain graph called *type graph*, and $m : G \rightarrow TG$ is a plain graph morphism, then (G, m) is a *typed graph*.

Typed graph morphisms map nodes and edges as plain graph morphisms additionally ensuring that nodes and edges are only mapped to nodes and edges of the same type.

Definition 16 (Typed Graph Morphism). If $H_1 = (G_1, m_1 : G_1 \rightarrow TG)$, $H_2 = (G_2, m_2 : G_2 \rightarrow TG)$ are typed graphs using the same type graph TG , $m : G_1 \rightarrow G_2$ is a plain graph morphism, $m_2 \circ m = m_1$ (which means that m maps a node/an edge of a certain type only to a node/an edge of the same type), then m is a *typed graph morphism* from H_1 to H_2 , also written $m : H_1 \rightarrow H_2$.

$$\begin{array}{ccc} G_1 & \xrightarrow{m} & G_2 \\ & \searrow m_1 & \swarrow m_2 \\ & & TG \end{array}$$

The *composition of two typed graph morphisms* $m : H_1 \rightarrow H_2$ and $m' : H_2 \rightarrow H_3$ is defined componentwise as for plain graph morphisms.

Definition 17 (Composition of Typed Graph Morphisms). If $m = (m_N, m_E) : H_1 \rightarrow H_2$ and $m' = (m'_N, m'_E) : H_2 \rightarrow H_3$ are typed graph morphisms, then $m' \circ m = (m'_N \circ m_N, m'_E \circ m_E) : H_1 \rightarrow H_3$ is the (*typed*) *composition* of m and m' .

$$\begin{array}{ccccc} H_1 & \xrightarrow{m} & H_2 & \xrightarrow{m'} & H_3 \\ & \searrow & & \nearrow & \\ & & & & m' \circ m \end{array}$$

Note that in the following, typed graphs are also called *graphs* and typed graph morphisms are also called *morphisms* for short.

The formal technique of GT relies on the notion of *GT rules*. A GT rule contains an injective morphism $\ell : I \rightarrow L$ that describes the removal of graph elements (technically given by $L - \ell(I)$) and an injective morphism $r : I \rightarrow R$ that describes the addition of graph elements (technically given by $R - r(I)$). Moreover, GT rules contain a finite set *NAC* of so-called *negative application conditions*. Each negative application condition is given by an injective morphism $n : L \rightarrow X$ describing (in comparison to L) additional graph elements that must not be in the graph to be transformed.

Definition 18 (GT Rule). If $\ell : I \rightarrow L$ and $r : I \rightarrow R$ are injective morphisms (i.e., both of their components are injective functions) and *NAC* is a finite set of injective morphisms $n : L \rightarrow X$, then $\rho = (\text{NAC}, \ell, r)$ is a *GT rule* also called *rule* for short.

$$\begin{array}{ccccc}
 & & X & & \\
 & & \uparrow & & \\
 \cdots & & n & & \\
 & & \downarrow & & \\
 L & \xleftarrow{\ell} & I & \xrightarrow{r} & R
 \end{array}$$

The rule application in a GT step is realized by the construction of two *pushouts*. In general, a pushout construction merges two graphs H_1 and H_2 to obtain a resulting graph G' . As for the union operation on sets, some graph elements may be identified (i.e., merged), which results in the case that G' contains fewer graph elements than the sum of the graph elements in H_1 and H_2 . To determine graph elements in H_1 and H_2 that are to be identified, two morphisms $m_1 : G \rightarrow H_1$ and $m_2 : G \rightarrow H_2$ are used. Two graph elements in H_1 and H_2 are identified when they are the images of a common graph element in G via m_1 and m_2 (i.e., $m_1(x)$ and $m_2(x)$ are identified for a common graph element x in G). The resulting morphisms $m'_1 : H_1 \rightarrow G'$ and $m'_2 : H_2 \rightarrow G'$ identify then the graph elements of H_1 and H_2 in the resulting graph G' . These morphisms are needed because e.g. the identification generally implies that m'_1 and m'_2 are not both inclusions for both of their components. For better comprehensibility of the preceding explanations see the diagram in Definition 19.

Definition 19 (Pushout). If $m_1 : G \rightarrow H_1$, $m_2 : G \rightarrow H_2$, $m'_1 : H_1 \rightarrow G'$, and $m'_2 : H_2 \rightarrow G'$ are morphisms, then (m'_1, m'_2) is the *pushout* of (m_1, m_2) according to the following construction. The graph G' is the largest graph such that m'_1 and m'_2 map together to all graph elements of G' and the square (1) below commutes (i.e., $m'_1 \circ m_1 = m'_2 \circ m_2$).

$$\begin{array}{ccc}
 G & \xrightarrow{m_1} & H_1 \\
 m_2 \downarrow & (1) & \downarrow m'_1 \\
 H_2 & \xrightarrow{m'_2} & G'
 \end{array}$$

A *GT step* uses a rule $\rho = (NAC, \ell : I \rightarrow L, r : I \rightarrow R)$ to transform a graph G into a graph H . For this purpose, a morphism $m : L \rightarrow G$ (also called *match*) needs to be determined first. A GT step is possible if none of the negative application conditions $n : L \rightarrow X$ from the set NAC can be embedded into the graph G to be transformed. Technically, this means that we attempt to match the elements of X into the graph G using some morphism $k : X \rightarrow G$ such that k maps graph elements in the same way as m with respect to the morphism n . To allow for the removal of graph elements according to the morphism ℓ of the rule, two morphisms $d : I \rightarrow D$ and $\ell' : D \rightarrow G$ are constructed such that (m, ℓ') is the pushout of (ℓ, d) . Intuitively, the graph D is constructed by preserving all graph elements of G that are not mapped to by m and by adding all graph elements that should be preserved by the rule application (i.e., the graph elements from $m(\ell(I))$). Implicitly, the removal of graph elements is subject to two conditions. Firstly, the *dangling condition* ensures that a node cannot be removed by the rule application when it has an adjacent edge that is not also removed by this rule application. Secondly, the *identification condition* ensures that if two graph elements in L are mapped to the same graph element in G (using a non-injective match m), then both of these elements in L should be removed or not removed by the rule application. To allow for the addition of graph elements according to the morphism r of the rule, two morphisms $\bar{m} : R \rightarrow H$ and $r' : D \rightarrow H$ are constructed such that (\bar{m}, r') is the pushout of (r, d) . Intuitively, the graph H is constructed by preserving all graph elements of D and by adding all graph elements that are not mapped to by r (i.e., the graph elements from $R - r(I)$). For better comprehensibility of the preceding explanations see the diagram in Definition 20.

Definition 20 (GT Step). If G and H are graphs, $\rho = (NAC, \ell : I \rightarrow L, r : I \rightarrow R)$ is a rule, $m : L \rightarrow G$, $d : I \rightarrow D$, $\bar{m} : R \rightarrow H$, $\ell' : D \rightarrow G$, and $r' : D \rightarrow H$ are morphisms, $\forall n : L \rightarrow X \in NAC. \nexists k : X \rightarrow G. k \circ n = m$ (which means that there is no negative application condition given by some injective morphism $n : L \rightarrow X$ contained in the set NAC of the rule ρ that can be extended using a morphism k to a morphism $k \circ n$ in a compatible way with the morphism m), (m, ℓ') is the pushout of (ℓ, d) , and (\bar{m}, r') is the pushout of (r, d) , then there is a *GT step* from G to H using rule ρ , written $G \xrightarrow{\rho} H$.

$$\begin{array}{ccccc}
X & \xleftarrow{n} & L & \xleftarrow{\ell} & I & \xrightarrow{r} & R \\
& \searrow \nexists k & \downarrow m & & \downarrow d & & \downarrow \bar{m} \\
& & G & \xleftarrow{\ell'} & D & \xrightarrow{r'} & H
\end{array}$$

A *graph transformation system* (GTS) contains a unique initial graph G_0 from which all sequences of GT steps start. Moreover, every GT step uses only one of the rules contained in the rule set P . To simplify the used rules, we assign to them priorities ensuring that a GT step modifying a graph G using a certain rule is part of the induced LTS of a GTS (i.e., of the GTS semantics) given below if there is no other rule with a strictly higher priority also applicable to G .

Definition 21 (Graph Transformation System (GTS)). If G_0 is a graph, P is a finite set of rules, and $\kappa : P \rightarrow \mathbf{N}$ is a mapping that assigns a priority (i.e., a natural number) to each rule in P , then $\mathcal{S} = (G_0, P, \kappa)$ is a *graph transformation system* (GTS).

Finally, to connect the formal technique of GT presented here to the notion of LTSs used in section 2.3, we define the notion of *induced LTS of a given GTS*. Such an induced LTS is the foundation for our analysis using SCCs.

Definition 22 (Induced LTS of a GTS). If $\mathcal{S} = (G_0, P, \kappa)$ is a GTS, S is the set of all graphs reachable by GT steps, L is the set of rules P , Z is the initial graph G_0 , and R is the labeled step relation that contains all GT steps $G \xrightarrow{\rho} H$ where $\rho \in P$ and where no GT step $G \xrightarrow{\rho'} H'$ with $\rho' \in P$ satisfies $\kappa(\rho') > \kappa(\rho)$, then $\mathcal{L} = (S, L, Z, R)$ is the *LTS induced by the GTS \mathcal{S}* .

A.2 Additional Details on the Computation Tree Logic (CTL)

For presenting CTL, we follow the introductory book ([2], section 6.2), which covers various linear and branching temporal logics together with algorithms for model checking. CTL is a propositional temporal logic in the sense that every state (i.e., every graph in our case) of a state space is labeled with the subset of the set of so-called atomic propositions Q , which are understood to be satisfied in these states. Based on these atomic propositions, further CTL state formulas are constructed using propositional operators such as *conjunction* and *negation* (hence, we automatically also allow for other propositional operators such as *disjunction* and *implication* as they can be derived from *conjunction* and *negation*). Moreover, the path operators *next* (written X) and *until* (written U) can be used to state that the next state in a selected path satisfies a certain condition or that all following states in a path satisfy a condition until some other condition is eventually satisfied, respectively. Here we also allow for other path operators such as e.g. *eventually* (written F) and *globally* (written G) as they can be derived from the path operators *next* and *until*. For more derived path operators see ([2], section 6.2). Lastly, the branching operators *for all* (written A) and *exists* (written E) can be used to switch from a state property to a path property by requiring that all paths or some path exiting the current state satisfy/satisfies a certain path condition, respectively. For example, the condition $(EX a) \rightarrow A(b U c)$ states for atomic propositions $Q = \{a, b, c\}$ and some current state s that if a is satisfied in some state directly reachable from s , then b is satisfied until eventually c is satisfied on every path starting in s .

Definition 23 (Syntax of CTL). If Q is a finite set of atomic propositions then ψ is a *computation tree logic (CTL) formula* over Q , written $\psi \in \mathcal{S}_Q^{\text{CTL}}$, if one of the following items applies.

- $\psi \in Q$.
- $\psi = \neg\psi'$ and $\{\psi'\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.

- $\psi = \psi_1 \wedge \psi_2$ and $\{\psi_1, \psi_2\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.
- $\psi = \text{AX}\psi'$ and $\{\psi'\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.
- $\psi = \text{EX}\psi'$ and $\{\psi'\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.
- $\psi = \text{A}(\psi_1 \text{ U } \psi_2)$ and $\{\psi_1, \psi_2\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.
- $\psi = \text{E}(\psi_1 \text{ U } \psi_2)$ and $\{\psi_1, \psi_2\} \subseteq \mathcal{S}_Q^{\text{CTL}}$.

For defining the semantics of CTL, we now introduce all paths (including the infinite paths) of an LTS as sequences of states where the labels of the steps are ignored.

Definition 24 (Possibly Infinite Paths of an LTS). If $\mathcal{L} = (S, L, Z, R)$ is an LTS, then R^{ω^*} contains all finite and infinite paths w over the states S such that for each pair (s, s') of adjacent states in w there is some label l such that $(s, l, s') \in R$. Moreover, w_i is the i th state of the path w .

Finally, we provide the CTL semantics (as described above) for the case of LTSs, which are equipped with a state-labeling operation ξ .

Definition 25 (Semantics of CTL). If $\mathcal{L} = (S, L, Z, R)$ is an LTS, $\xi \subseteq S \times Q$ is a relation between states S of \mathcal{L} and atomic propositions Q , s is a state from S , and $\psi \in \mathcal{S}_Q^{\text{CTL}}$ is a CTL formula over Q , then s satisfies ψ for \mathcal{L} and ξ , written $(\mathcal{L}, \xi, s) \models \psi$, if one of the following items applies.

- $\psi \in Q$ and $(s, \psi) \in \xi$.
- $\psi = \neg\psi'$ and $(\mathcal{L}, \xi, s) \not\models \psi'$.
- $\psi = \psi_1 \wedge \psi_2$, $(\mathcal{L}, \xi, s) \models \psi_1$, and $(\mathcal{L}, \xi, s) \models \psi_2$.
- $\psi = \text{AX}\psi'$ and for each step $(s, l, s') \in R$ it holds that $(\mathcal{L}, \xi, s') \models \psi'$.
- $\psi = \text{EX}\psi'$ and for some step $(s, l, s') \in R$ it holds that $(\mathcal{L}, \xi, s') \models \psi'$.
- $\psi = \text{A}(\psi_1 \text{ U } \psi_2)$ and for each path $w \in R^{\omega^*}$ with $w_0 = s$ there is some n such that $(\mathcal{L}, \xi, w_n) \models \psi_2$ and for each $j < n$ it holds that $(\mathcal{L}, \xi, w_j) \models \psi_1$.
- $\psi = \text{E}(\psi_1 \text{ U } \psi_2)$ and there is some path $w \in R^{\omega^*}$ with $w_0 = s$ such that there is some n such that $(\mathcal{L}, \xi, w_n) \models \psi_2$ and for each $j < n$ it holds that $(\mathcal{L}, \xi, w_j) \models \psi_1$.

A.3 Proofs for Results

We now present the proofs for Theorem 2 and Theorem 3. Theorem 1 is justified by application of the tool GROOVE.

of Theorem 2. Let $\mathcal{L} = (S, L, Z, R)$ be the state space (i.e., an LTS) induced by the GT model $\mathcal{S}_U = (G_0, P_U, \kappa_U)$. We make use of the invariant that ensures that each state G of \mathcal{L} (given by a graph) can be represented by a tuple (n_1, n_2, n_3, n_4, b) where (i) n_1 is the number of *Eco*-nodes in G with a connected *Tree*- and a connected *Grass*-node, (ii) n_2 is the number of *Eco*-nodes in G with no connected *Tree*- and a connected *Grass*-node, (iii) n_3 is the number of *Eco*-nodes in G with no connected *Tree*- and no connected *Grass*-node, (iv) n_4 is the number of *Eco*-nodes in G with a connected *Tree*- and no connected *Grass*-node, and (v) $b = T$ if the *Rain*-node in G has an *on*-loop and $b = F$ if the *Rain*-node in G has an *off*-loop.

We prove the following items to focus on the central statements indicated in the theorem.

1. All two graphs from $C_1 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 > 0\}$ are mutually reachable.
2. All two graphs from $C_2 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0\}$ are mutually reachable.
3. $\exists G_1 \in C_1. \exists G_2 \in C_2. (G_1, HighFire, G_2) \in R$, which means that there is a step using the rule *HighFire* from C_1 to C_2 .
4. There are no other steps between C_1 and C_2 , which also means that C_1 and C_2 are maximal and, hence, SCCs.

We now consider the four items from above.

1. We need to show that all two graphs from $C_1 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 > 0\}$ are mutually reachable.
 Fix a list of labels π_G and $G = (n_1, n_2, n_3, n_4, b)$ with $(G_0, \pi_G, G) \in R^*$ and $n_1 + n_4 > 0$.
 Fix a list of labels $\pi_{G'}$ and $G' = (n'_1, n'_2, n'_3, n'_4, b')$ with $(G_0, \pi_{G'}, G') \in R^*$ and $n'_1 + n'_4 > 0$.
 We need to show that there is a path from G to G' .
 We show this by proving the existence of two paths $(G, \pi_1, G_0) \in R^*$ and $(G_0, \pi_2, G') \in R^*$.

- We need to show that there is a path from G to G_0 :
 We construct the list of labels π_1 needed to transform $G = (n_1, n_2, n_3, n_4, b)$ into $G_0 = (2, 0, 0, 0, T)$ as follows.¹

$$\pi_1 = DelEco^{n_3}. \tag{A.1}$$

$$(if\ b = T\ then\ RainOff\ else\ \epsilon). \tag{A.2}$$

$$(LowFire \cdot DelEco)^{n_2}. \tag{A.3}$$

$$RainOn. \tag{A.4}$$

¹The list of labels containing precisely n copies of the rule ρ is denoted by ρ^n . Similarly, the list of labels containing precisely n copies of the list of labels π is denoted by π^n .

$$\text{GrowGrass}^{n_4}. \quad (\text{A.5})$$

$$\text{AddEcoU}. \quad (\text{A.6})$$

$$\text{GrowTree}. \quad (\text{A.7})$$

$$\text{RainOff}. \quad (\text{A.8})$$

$$(\text{HighFire} \cdot \text{DelEco})^{n_1+n_4+1-2}. \quad (\text{A.9})$$

$$\text{RainOn} \quad (\text{A.10})$$

We now discuss the individual steps one by one.

Using (A.1): All n_3 *Eco*-nodes without *Tree*- and *Grass*-nodes are removed using the rule *DelEco* (note that n_3 can be at most one for all reachable states because the rule *DelEco* has a higher priority than all other rules). The resulting state is $(n_1, n_2, 0, n_4, b)$. Using (A.2): The rainy season changes to the dry season using the rule *RainOff*, if S_U is currently in the rainy season. The resulting state is $(n_1, n_2, 0, n_4, F)$. Using (A.3): The n_2 *Eco*-nodes with only a *Grass*-node are removed using the rules *LowFire* and *DelEco*. Note that the rule *DelEco* must be applied directly after the rule *LowFire* due to its higher priority. The resulting state is $(n_1, 0, 0, n_4, F)$. Using (A.4): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(n_1, 0, 0, n_4, T)$. Using (A.5): On each of the n_4 *Eco*-nodes with a *Tree*-node but no *Grass*-node, a *Grass*-node is added using the rule *GrowGrass*. The resulting state is $(n_1 + n_4, 0, 0, 0, T)$. Using (A.6): For the case that there are not at least two *Eco*-nodes with *Tree*- and *Grass*-nodes now, another *Eco*-node with a *Grass*-node (by construction) is added using the rule *AddEcoU*. The resulting state is $(n_1 + n_4, 0, 0, 1, T)$. Using (A.7): On this new *Eco*-node, a *Tree*-node is added using the rule *GrowTree*. The resulting state is $(n_1 + n_4 + 1, 0, 0, 0, T)$. Using (A.8): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(n_1 + n_4 + 1, 0, 0, 0, F)$. Using (A.9): All but two *Eco*-nodes with *Tree*- and *Grass*-nodes are removed using the rules *HighFire* and *DelEco*. Note that the rule *DelEco* must be applied directly after the rule *HighFire* due to its higher priority. The resulting state is $(2, 0, 0, 0, F)$. Using (A.10): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(2, 0, 0, 0, T)$.

The graph representing the resulting state is, as required, equal to G_0 .

- We need to show that there is a path from G_0 to G' :
This path trivially exists using $\pi_2 = \pi_{G'}$ from above.

2. We need to show that all two graphs from $C_2 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0\}$ are mutually reachable.

Fix a list of labels π_G and $G = (n_1, n_2, n_3, n_4, b)$ with $(G_0, \pi_G, G) \in R^*$ and $n_1 + n_4 = 0$.

Fix a list of labels $\pi_{G'}$ and $G' = (n'_1, n'_2, n'_3, n'_4, b')$ with $(G_0, \pi_{G'}, G') \in R^*$ and $n'_1 + n'_4 = 0$.

We need to show that there is a path from G to G' .

We show this by proving the existence of two paths $(G, \pi_1, \bar{G}) \in R^*$ and $(\bar{G}, \pi_2, G') \in R^*$ where $\bar{G} = (0, 0, 0, 0, F)$.

- We need to show that there is a path from G to \bar{G} :

We construct the list of labels π_1 needed to transform $G = (0, n_2, n_3, 0, b)$ into $\bar{G} = (0, 0, 0, 0, F)$ as follows.

$$\pi_1 = \text{DelEco}^{n_3}. \quad (\text{A.11})$$

$$(\text{if } b = T \text{ then RainOff else } \epsilon). \quad (\text{A.12})$$

$$(\text{LowFire} \cdot \text{DelEco})^{n_2} \quad (\text{A.13})$$

We now discuss the individual steps one by one.

Using (A.11): All n_3 *Eco*-nodes without *Tree*- and *Grass*-nodes are removed using the rule *DelEco* (note that n_3 can be at most one for all reachable states because the rule *DelEco* has a higher priority than all other rules). The resulting state is $(0, n_2, 0, 0, b)$. Using (A.12): The rainy season changes to the dry season using the rule *RainOff*, if S_U is currently in the rainy season. The resulting state is $(0, n_2, 0, 0, F)$. Using (A.13): The n_2 *Eco*-nodes with only a *Grass*-node are removed using the rules *LowFire* and *DelEco*. Note that the rule *DelEco* must be applied directly after the rule *LowFire* due to its higher priority. The resulting state is $(0, 0, 0, 0, F)$.

The graph representing the resulting state is, as required, equal to \bar{G} .

- We need to show that there is a path from \bar{G} to G' :

We construct the list of labels π_2 needed to transform $\bar{G} = (0, 0, 0, 0, F)$ into $G' = (0, n'_2, n'_3, 0, b')$ as follows.

$$\pi_2 = \text{RainOn}. \quad (\text{A.14})$$

$$\text{AddEcoU}^{n'_2 + n'_3}. \quad (\text{A.15})$$

$$\text{RainOff}. \quad (\text{A.16})$$

$$\text{LowFire}^{n'_3}. \quad (\text{A.17})$$

$$(\text{if } b' = T \text{ then RainOn else } \epsilon) \quad (\text{A.18})$$

We now discuss the individual steps one by one.

Using (A.14): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(0, 0, 0, 0, T)$. Using (A.15): $n'_2 + n'_3$ *Eco*-nodes, each with a *Grass*-node (by construction), are added using the rule *AddEcoU*. The resulting state is $(0, n'_2 + n'_3, 0, 0, T)$. Using (A.16): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(0, n'_2 + n'_3, 0, 0, F)$. Using (A.17): The n'_3 *Grass*-nodes are removed using the rule *LowFire* (note that n'_3 can be at most one for all reachable states as before). The resulting state is $(0, n'_2, n'_3, 0, F)$. Using (A.18): The dry season changes to the rainy season using the rule *RainOn* if required. Note that b' cannot be equal to T when $n'_3 = 1$ because $n'_3 = 1$ can only be reached using the rule *LowFire*, which can only be applied during the

dry season. Moreover, as the rule *DelEco* must be applied directly after the rule *LowFire*, \mathcal{S}_U has still to be in the dry season. The resulting state is $(0, n'_2, n'_3, 0, b')$.

The graph representing the resulting state is, as required, equal to G' .

3. We need to show that $\exists G_1 \in C_1. \exists G_2 \in C_2. (G_1, HighFire, G_2) \in R$, which means that there is a step using the rule *HighFire* from C_1 to C_2 .
Let $G_1 = (1, 0, 0, 0, F)$ and $G_2 = (0, 0, 1, 0, F)$. Obviously, $(G_1, HighFire, G_2) \in R$. It is sufficient to show now that G_1 is in C_1 . Note that G_2 is in C_2 by appending the rule *HighFire* to the list of labels used for showing that G_1 is in C_1 .
We construct the list of labels π needed to transform $G_0 = (2, 0, 0, 0, T)$ into $G_1 = (1, 0, 0, 0, F)$ as follows.

$$\pi = RainOff. \tag{A.19}$$

$$HighFire. \tag{A.20}$$

$$DelEco \tag{A.21}$$

We now discuss the individual steps one by one.

Using (A.19): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(2, 0, 0, 0, F)$. Using (A.20): One *Tree*- and one *Grass*-node are removed from one *Eco*-node using the rule *HighFire*. The resulting state is $(1, 0, 1, 0, F)$. Using (A.21): The *Eco*-node without a *Tree*- and a *Grass*-node is removed using the rule *DelEco*. The resulting state is $(1, 0, 0, 0, F)$.

The graph representing the resulting state is, as required, equal to G_1 .

4. We need to show that there are no other steps between C_1 and C_2 , which also means that C_1 and C_2 are maximal and, hence, SCCs.
 - Assume that there are $\rho \in P, G_1 \in C_1$, and $G_2 \in C_2$ such that $(G_2, \rho, G_1) \in R$. But there is no such rule ρ in the GT model \mathcal{S}_U as follows. Only the rule *GrowTree* can increase the number of *Tree*-nodes but this rule requires the existence of a *Tree*-node in G_2 . Hence, there is no such step $(G_2, \rho, G_1) \in R$ as required.
 - Assume that there are $\rho \in P, G_1 \in C_1$, and $G_2 \in C_2$ such that $(G_1, \rho, G_2) \in R$ and $\rho \neq HighFire$. But the rule *HighFire* is the only rule that can decrease the number of *Tree*-nodes as required for such a step (G_1, ρ, G_2) .

□

of Theorem 3. Let $\mathcal{L} = (S, L, Z, R)$ be the state space (i.e., an LTS) induced by the GT model $\mathcal{S}_D = (G_0, P_D, \kappa_D)$. We make use of the invariant as in the proof of Theorem 2 before.

We prove the following items to focus on the central statements indicated in the theorem.

1. All two graphs from $C_1 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 > 0\}$ are mutually reachable.

2. For each $n \in \mathbf{N}$ it holds that all two graphs from $C_{2,n,0} = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0 \wedge n_2 = n \wedge n_3 = 0\}$ are mutually reachable.
3. For each $n \in \mathbf{N}$ it holds that all two graphs from $C_{2,n,1} = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0 \wedge n_2 = n \wedge n_3 = 1 \wedge b = F\}$ are mutually reachable.
4. $\forall n \in \mathbf{N}. \exists G_1 \in C_1. \exists G_2 \in C_{2,n,1}. (G_1, HighFire, G_2) \in R$, which means that there is a step using the rule *HighFire* from C_1 to $C_{2,n,1}$.
5. $\forall n \in \mathbf{N}. \exists G_1 \in C_{2,n,1}. \exists G_2 \in C_{2,n,0}. (G_1, DelEco, G_2) \in R$, which means that there is a step using the rule *DelEco* from $C_{2,n,1}$ to $C_{2,n,0}$.
6. $\forall n \in \mathbf{N} - \{0\}. \exists G_1 \in C_{2,n,0}. \exists G_2 \in C_{2,n-1,1}. (G_1, LowFire, G_2) \in R$, which means that there is a step using the rule *LowFire* from $C_{2,n,0}$ to $C_{2,n-1,1}$.
7. There are no other steps between the mentioned SCC candidates, which means that they are maximal and, hence, SCCs.

We now consider the seven items from above.

1. We need to show that all two graphs from $C_1 = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 > 0\}$ are mutually reachable.
 Fix a list of labels π_G and $G = (n_1, n_2, n_3, n_4, b)$ with $(G_0, \pi_G, G) \in R^*$ and $n_1 + n_4 > 0$.
 Fix a list of labels $\pi_{G'}$ and $G' = (n'_1, n'_2, n'_3, n'_4, b')$ with $(G_0, \pi_{G'}, G') \in R^*$ and $n'_1 + n'_4 > 0$.
 We need to show that there is a path from G to G' .
 We show this by proving the existence of two paths $(G, \pi_1, G_0) \in R^*$ and $(G_0, \pi_2, G') \in R^*$.
 - We need to show that there is a path from G to G_0 :
 We construct the list of labels π_1 needed to transform $G = (n_1, n_2, n_3, n_4, b)$ into $G_0 = (2, 0, 0, 0, T)$ as follows (which is similar to the corresponding part in the proof of Theorem 2 above).

$$\pi_1 = DelEco^{n_3}. \tag{A.22}$$

$$(if\ b = T\ then\ RainOff\ else\ \epsilon). \tag{A.23}$$

$$(LowFire \cdot DelEco)^{n_2}. \tag{A.24}$$

$$RainOn. \tag{A.25}$$

$$GrowGrass^{n_4}. \tag{A.26}$$

$$AddEcoD. \tag{A.27}$$

$$GrowTree. \tag{A.28}$$

$$RainOff. \tag{A.29}$$

$$(HighFire \cdot DelEco)^{n_1+n_4+1-2}. \tag{A.30}$$

$$RainOn \tag{A.31}$$

We now discuss the individual steps one by one.

Using (A.22): All n_3 *Eco*-nodes without *Tree*- and *Grass*-nodes are removed using the rule *DelEco* (note that n_3 can be at most one for all reachable states because the rule *DelEco* has a higher priority than all other rules). The resulting state is $(n_1, n_2, 0, n_4, b)$. Using (A.23): The rainy season changes to the dry season using the rule *RainOff*, if S_D is currently in the rainy season. The resulting state is $(n_1, n_2, 0, n_4, F)$. Using (A.24): The n_2 *Eco*-nodes with only a *Grass*-node are removed using the rules *LowFire* and *DelEco*. Note that the rule *DelEco* must be applied directly after the rule *LowFire* due to its higher priority. The resulting state is $(n_1, 0, 0, n_4, F)$. Using (A.25): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(n_1, 0, 0, n_4, T)$. Using (A.26): On each of the n_4 *Eco*-nodes with a *Tree*-node but no *Grass*-node, a *Grass*-node is added using the rule *GrowGrass*. The resulting state is $(n_1 + n_4, 0, 0, 0, T)$. Using (A.27): For the case that there are not at least two *Eco*-nodes with *Tree*- and *Grass*-nodes now, another *Eco*-node with a *Grass*-node (by construction) is added using the rule *AddEcoD*. Note that the required *Eco*-node with a *Tree*- and a *Grass*-node exists for the application of the rule *AddEcoD*. The resulting state is $(n_1 + n_4, 0, 0, 1, T)$. Using (A.28): On this new *Eco*-node, a *Tree*-node is added using the rule *GrowTree*. The resulting state is $(n_1 + n_4 + 1, 0, 0, 0, T)$. Using (A.29): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(n_1 + n_4 + 1, 0, 0, 0, F)$. Using (A.30): All but two *Eco*-nodes with *Tree*- and *Grass*-nodes are removed using the rules *HighFire* and *DelEco*. Note that the rule *DelEco* must be applied directly after the rule *HighFire* due to its higher priority. The resulting state is $(2, 0, 0, 0, F)$. Using (A.31): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(2, 0, 0, 0, T)$. The graph representing the resulting state is, as required, equal to G_0 .

- We need to show that there is a path from G_0 to G' :
This path trivially exists using $\pi_2 = \pi_{G'}$ from above.

2. We need to show that for each $n \in \mathbf{N}$ it holds that all two graphs from $C_{2,n,0} = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0 \wedge n_2 = n \wedge n_3 = 0\}$ are mutually reachable.
Only the two graphs $G_1 = (0, n, 0, 0, F)$ and $G_2 = (0, n, 0, 0, T)$ may be in $C_{2,n,0}$. The rules *RainOn* and *RainOff* can be applied to reach those graphs from each other.
3. We need to show that for each $n \in \mathbf{N}$ it holds that all two graphs from $C_{2,n,1} = \{G \mid \exists \pi, n_1, n_2, n_3, n_4, b. (G_0, \pi, G) \in R^* \wedge G = (n_1, n_2, n_3, n_4, b) \wedge n_1 + n_4 = 0 \wedge n_2 = n \wedge n_3 = 1 \wedge b = F\}$ are mutually reachable.
 $C_{2,n,1}$ may only contain the graph $G = (0, n, 1, 0, F)$. Hence, the empty list of labels ϵ can be used for the mutual reachability.

4. We need to show that $\forall n \in \mathbf{N}. \exists G_1 \in C_1. \exists G_2 \in C_{2,n,1}. (G_1, HighFire, G_2) \in R$, which means that there is a step using the rule *HighFire* from C_1 to $C_{2,n,1}$.

Fix some $n \in \mathbf{N}$.

Let $G_1 = (1, n, 0, 0, F)$ and $G_2 = (0, n, 1, 0, F)$. Obviously, $(G_1, HighFire, G_2) \in R$. It is sufficient to show now that G_1 is in C_1 . Note that G_2 is in $C_{2,n,1}$ by appending the rule *HighFire* to the list of labels used for showing that G_1 is in C_1 .

We construct the list of labels π needed to transform $G_0 = (2, 0, 0, 0, T)$ into $G_1 = (1, n, 0, 0, F)$ as follows.

$$\pi = RainOff. \tag{A.32}$$

$$HighFire. \tag{A.33}$$

$$DelEco. \tag{A.34}$$

$$RainOn. \tag{A.35}$$

$$AddEcoD^n. \tag{A.36}$$

$$RainOff \tag{A.37}$$

We now discuss the individual steps one by one.

Using (A.32): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(2, 0, 0, 0, F)$. Using (A.33): One *Tree*- and one *Grass*-node are removed from one *Eco*-node using the rule *HighFire*. The resulting state is $(1, 0, 1, 0, F)$. Using (A.34): The *Eco*-node without a *Tree*- and a *Grass*-node is removed using the rule *DelEco*. The resulting state is $(1, 0, 0, 0, F)$. Using (A.35): The dry season changes to the rainy season using the rule *RainOn*. The resulting state is $(1, 0, 0, 0, T)$. Using (A.36): n *Eco*-nodes, each with a *Grass*-node (by construction), are added using the rule *AddEcoD*, which can be applied because there is still one *Eco*-node with a *Grass*- and a *Tree*-node. The resulting state is $(1, n, 0, 0, T)$. Using (A.37): The rainy season changes to the dry season using the rule *RainOff*. The resulting state is $(1, n, 0, 0, F)$.

The graph representing the resulting state is, as required, equal to G_1 .

5. We need to show that $\forall n \in \mathbf{N}. \exists G_1 \in C_{2,n,1}. \exists G_2 \in C_{2,n,0}. (G_1, DelEco, G_2) \in R$, which means that there is a step using the rule *DelEco* from $C_{2,n,1}$ to $C_{2,n,0}$.

Fix some $n \in \mathbf{N}$.

Let $G_1 = (0, n, 1, 0, F)$ and $G_2 = (0, n, 0, 0, F)$. Obviously, $(G_1, DelEco, G_2) \in R$. It is sufficient to show now that G_1 is in $C_{2,n,1}$. This has been shown in the previous item. Note that G_2 is in $C_{2,n,0}$ by appending the rule *DelEco* to the list of labels used for showing that G_1 is in $C_{2,n,1}$.

6. We need to show that $\forall n \in \mathbf{N} - \{0\}. \exists G_1 \in C_{2,n,0}. \exists G_2 \in C_{2,n-1,1}. (G_1, LowFire, G_2) \in R$, which means that there is a step using the rule *LowFire* from $C_{2,n,0}$ to $C_{2,n-1,1}$.

Fix some $n \in \mathbf{N}$ with $n > 0$.

Let $G_1 = (0, n, 0, 0, F)$ and $G_2 = (0, n - 1, 1, 0, F)$. Obviously, $(G_1, LowFire, G_2) \in$

R. It is sufficient to show now that G_1 is in $C_{2,n,0}$. This has been shown in the previous item. Note that G_2 is in $C_{2,n-1,1}$ by appending the rule *LowFire* to the list of labels used for showing that G_1 is in $C_{2,n,0}$.

7. We need to show that there are no other steps between the mentioned SCC candidates, which means that they are maximal and, hence, SCCs.
 - Assume that there are $\rho \in P$, $G_1 \in C_1$, and $G_2 \in C_{2,n,0} \cup C_{2,n,1}$ such that $(G_2, \rho, G_1) \in R$. But there is no such rule ρ in the GT model \mathcal{S}_D as follows. Only the rule *GrowTree* can increase the number of *Tree*-nodes but this rule requires the existence of a *Tree*-node in G_2 . Hence, there is no such step $(G_2, \rho, G_1) \in R$ as required.
 - Assume that there are $\rho \in P$, $G_1 \in C_1$, and $G_2 \in C_{2,n,0} \cup C_{2,n,1}$ such that $(G_1, \rho, G_2) \in R$ and $\rho \neq \textit{HighFire}$. But the rule *HighFire* is the only rule that can decrease the number of *Tree*-nodes as required for such a step (G_1, ρ, G_2) .
 - Assume that there are $\rho \in P$, $G_1 \in C_{2,n,0}$, and $G_2 \in C_{2,m,0}$ such that $(G_1, \rho, G_2) \in R$ and $n \neq m$. There is no rule that changes the number of *Eco*-nodes with a connected *Grass*-node. Each removal of *Eco*-nodes requires two steps using the rules *LowFire* and *DelEco*. Each addition of *Eco*-nodes requires an existing *Eco*-node with a connected *Tree*- and a connected *Grass*-node using the rule *AddEcoD*.
 - Assume that there are $\rho \in P$, $G_1 \in C_{2,n,0}$, and $G_2 \in C_{2,m,1}$ such that $(G_1, \rho, G_2) \in R$, $n \neq m$, and $\rho \neq \textit{LowFire}$. The only two rules that lead to the existence of an *Eco*-node with no *Tree*- and no *Grass*-node are the rules *LowFire* and *HighFire*. The rule *LowFire* is excluded here by assumption. The rule *HighFire* cannot be applied because there is no *Tree*-node in G_1 .
 - Assume that there are $\rho \in P$, $G_1 \in C_{2,n,1}$, and $G_2 \in C_{2,m,1} \cup C_{2,m,0}$ such that $(G_1, \rho, G_2) \in R$ and $n \neq m$. ρ must be the rule *DelEco* due to the priority but the application of that rule retains the number of *Eco*-nodes with only a *Grass*-node implying $n = m$.

□

Current Technical Reports of the Hasso-Plattner-Institute

| Vol. | ISBN | Titel | Authors/Editors |
|------|-------------------|---|--|
| 146 | 978-3-86956-532-3 | Probabilistic metric temporal graph logic | Sven Schneider, Maria Maximova, Holger Giese |
| 145 | 978-3-86956-528-6 | Learning from failure: a history-based, lightweight test prioritization technique connecting software changes to test failures | Falco Dürsch, Patrick Rein, Toni Mattis, Robert Hirschfeld |
| 144 | 978-3-86956-526-2 | Die HPI Schul-Cloud – Von der Vision zur digitalen Infrastruktur für deutsche Schulen | Christoph Meinel, Catrina John, Tobias Wollowski, HPI Schul-Cloud Team |
| 143 | 978-3-86956-531-6 | Invariant Analysis for Multi-Agent Graph Transformation Systems using k-Induction | Sven Schneider, Maria Maximova, Holger Giese |
| 142 | 978-3-86956-524-8 | Quantum computing from a software developers perspective | Marcel Garus, Rohan Sawahn, Jonas Wanke, Clemens Tiedt, Clara Granzow, Tim Kuffner, Jannis Rosenbaum, Linus Hagemann, Tom Wollnik, Lorenz Woth, Felix Auringer, Tobias Kantusch, Felix Roth, Konrad Hanff, Niklas Schilli, Leonard Seibold, Marc Fabian Lindner, Selina Raschack |
| 141 | 978-3-86956-521-7 | Tool support for collaborative creation of interactive storytelling media | Paula Klinke, Silvan Verhoeven, Felix Roth, Linus Hagemann, Tarik Alnawa, Jens Lincke, Patrick Rein, Robert Hirschfeld |
| 140 | 978-3-86956-517-0 | Probabilistic metric temporal graph logic | Sven Schneider, Maria Maximova, Holger Giese |
| 139 | 978-3-86956-514-9 | Deep learning for computer vision in the art domain : proceedings of the master seminar on practical introduction to deep learning for computer vision, HPI WS 20/21 | Christian Bartz, Ralf Krestel |

ISBN 978-3-86956-533-0
ISSN 1613-5652