Emergence and complex systems: The contribution of dynamic graph theory

Jacques Gignoux, Guillaume Chérel, Ian D. Davies, Shayne R Flint, Eric Lateltin

Emergence and complex systems: the contribution of dynamic graph theory

Jacques Gignoux[1*], Guillaume Chérel[2], Ian D. Davies[3], Shayne R. Flint[4], Eric Lateltin[1]

January 31, 2017

1. Institute of Ecology and Environmental Sciences, CNRS UMR 7618, 4 place Jussieu, aile 44-45 CC 237, 75005 Paris, France
2. Institut des Systèmes Complexes Paris Île-de-France, 113 rue Nationale, 75013 Paris, France
3. Fenner School of Environment and Society, The Australian National University, Canberra ACT 2601, Australia
4. Research School of Computer Science, The Australian National University, Canberra ACT 0200, Australia

* corresponding author: jacques.gignoux@upmc.fr

Abstract

Emergence and complex systems have been the topic of many papers and are still disputed concepts in many fields. This lack of consensus hinders the use of these concepts in practice, particularly in modelling. All definitions of emergence imply the existence of a hierarchical system: a system that can be observed, measured and analysed at both macroscopic and microscopic levels. We argue that such systems are well described by mathematical graphs and, using graph theory, we propose an ontology (*i.e.* a set of consistent, formal concept definitions) of dynamic hierarchical systems capable of displaying emergence. Using graph theory enables formal definitions of system macro-state, micro-state and dynamic structural changes. From these definitions, we identify four major families of emergence that match existing definitions from the literature. All but one depend on the relation between the observer and the system, and remind us that a major feature of most supposedly complex systems is our inability to describe them in full. The fourth definition is related to causality, in particular, to the ability of the system itself to create sources of change, independent from other external or internal sources. Feedback loops play a key role in this process. We propose that their presence is a necessary condition for a hierarchical system to be qualified as complex.

Keywords: hierarchy, ontology, feedback loop, causality, computational irreducibility

Funding: This work was supported by the French Agence Nationale de la Recherche, grant number ANR-07-CIS7-001.

Word count: abstract 215 words; main text: 9981 words

Introduction

Interest in the concepts of *complex system* and *emergence* has been pursued over many years in many fields. In such circumstances, it is inevitable that conflicting definitions will arise. When attempting a definition, it is important to be wary of including concepts that are either themselves poorly defined or are merely correlated with the concept at hand (Jax, 2007). With these caveats in mind, we propose a formal definition of a system from which we can formalise the circumstances under which emergence may arise.

In doing this, we build an *ontology* of useful and rigorous concepts related to emergence. An ontology (in computer science) is a set of formal definitions of concepts and their relationships, that can lead to automatic processing for the construction of formal grammars and software (Guarino, 1995). It is our hope that this methodology will underpin a broadly applicable clarification of these concepts (*e.g.* an application to ecology in Gignoux et al., 2011). Following Jax (2007), we begin by using mathematical notation to define the concept of a 'system'. Our notation provides for, but does not impose, the possibility of emergent properties, based on the commonalities between most definitions of emergence. In so doing, we extract generic properties of 'systems with emergence'.

Despite fundamental differences, all definitions of emergence share a common assumption: emergence arises only in systems that can be described at both *macroscopic* and *microscopic* levels (de Haan, 2006; De Wolf & Holvoet, 2005; Bedau, 2003). A system with such properties is usually called a *hierarchical system* (Allen & Hoekstra, 1992; Ahl & Allen, 1996; O'Neill et al., 1986). In contrast, a *non-hierarchical system*, also called *atomic system,* is one which cannot be divided into sub-systems; it is atomic in the sense that we have no knowledge of a microscopic representation (see below Definition 24).

How can we formally define a hierarchical system in a generic way? In one of its most commonly accepted definitions (Carnot, 1824), a system is *'the part of the world under consideration for a particular purpose'*. Implicit in this definition is the existence of an observer, someone or thing for which a part of the world is extracted for consideration to some end. The ecosystem, as initially defined by Tansley (1935), falls within the scope of this definition, just as do, for example, thermodynamic systems and systems of social organisation. In the field of systems thinking, Jordan (1981) finds that nothing more specific can be said in defining the term 'system' (the fundamental concept in the author's discipline) other than that *'a system is composed of identifiable entities and their relationships'*. This definition is just as applicable to concrete objects as it is to *virtual* or *conceptual* objects. For emergence to occur, the system must be characterised as *hierarchical*, in the sense that we can provide both a macroscopic and a microscopic description. We will therefore define a hierarchical system as *an object composed of components in interaction.* This is close to some definitions of a complex system, but we make no assumption about emergence as this is precisely what we wish to explore.

A system comprising components and their interactions is well described by a mathematical graph (Diestel, 2000; Gross & Yellen, 1999). A graph is a set of nodes connected by edges. We propose to represent a hierarchical system as a mathematical graph: the 'interacting components' that produce the 'microscopic state' of the system are the nodes, the edges represent interactions, while the system as a whole is represented by the graph. Although the hierarchical relation between the graph and its components is not explicit at this stage, this representation allows us to consider both a *macroscopic* view of the system – the graph as a whole – and a *microscopic* view – the list of all its components *and their interactions.*

1 Formal definitions for a hierarchical system: an ontology

We first provide a minimal set of mathematical definitions to describe a system without any a priori knowledge of emergence.

1.1 The system

We postulate that a hierarchical system can be represented as a graph. We call the *world* $\mathscr{W}$, that set of objects from which an observer draws a subset to build a system *for some purpose*. Components of the system are defined as objects $c \in \mathscr{W}$, and interactions as relations between any two components of $\mathscr{W}$.

Proposition 1. *A* hierarchical system S *is defined as the graph:*

$$S := (C, R, \gamma)$$

where C *is the set of* components *(nodes) of the system:*

$$C := \{c_u\}_{u \leq n_c < \infty}\,,\; c_u \in \mathscr{W}$$

R *is the set of* relations *(edges) between components of the system:*

$$R := \{r_v\}_{v \leq n_r < \infty}\,,\; r_v \in \mathscr{W}^2$$

and γ *is the* incidence function*, which assigns a relation to a pair of components:*

$$\begin{aligned} \gamma : R &\to C \times C \\ r_v &\to (c_i, c_j)_{i \leq n_c,\, j \leq n_c} \end{aligned}$$

n_c *is the number of components and* n_r *the number of relations of the system;* $\mathscr{W}^2$ *is the set of applications from* $\mathscr{W}$ *to* $\mathscr{W}$*.*
We make no assumption as to the type of graph used to represent S*. It can be directed, undirected, a multigraph or any other kind of graph, hence the need for an explicit incidence function.*
Where it may be ambiguous, we subscript sets C*,* R *and function* γ *by the graph to which they belong.*

Figure 1 gives examples of systems represented as graphs.
For later simplification, its is convenient to define:

Definition 1. Components c_u and relations r_v are called *elements* of the system S. We denote them by $e_w \in E$, with $E = C \cup R$ and the element index w verifies $w \leq n_c + n_r = n_e$. The set of all possible elements is $\mathscr{E} = \mathscr{W} \cup \mathscr{W}^2$.
In the UML language (Object Management Group, 2015), components and relations are *specialisations* of elements (cf. Appendix).

Definition 2. The *connection set* of a system component, $\zeta(c_u)$, is the set of relations connected to this component:

$$\zeta(c_u) = \left\{ r_v \in R \,|\, \gamma(r_v) = (c_u, c_i)_{i \leq n_c} \text{ or } \gamma(r_v) = (c_i, c_u)_{i \leq n_c} \right\}$$

So far, the hierarchical relation between the system and its components has not been made explicit. To provide for the possibility of building successive levels of nested systems, as in hierarchy theory (Allen & Hoekstra, 1992), we introduce a graph operator (a function on a graph returning a graph) to transform the system into a tree. This tree is built by (1) creating a new node that represents the whole system, denoted by $\text{top}(S)$, and (2) linking this node to all the components of S:

Definition 3. For a system S, we define its *hierarchical view* $H(S)$ as a *directed* graph resulting from the

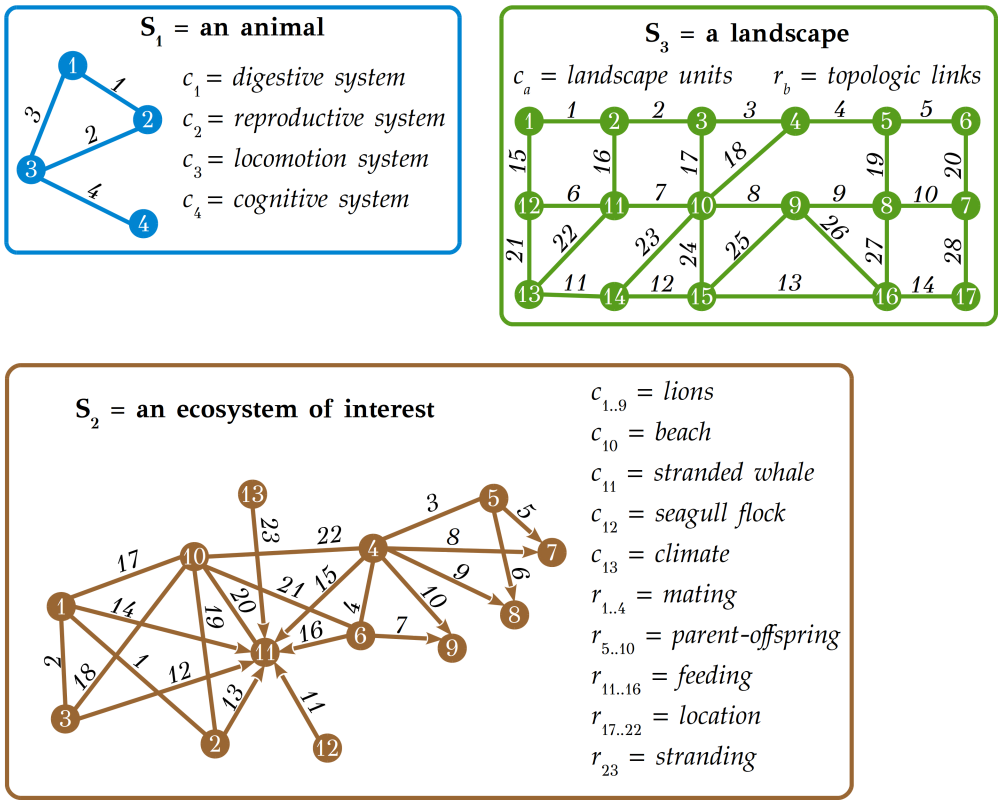


Figure 1: Three different examples of systems represented as graphs using Proposition 1. Circles denote system components c_u and lines between them denote relations r_v.

application of the graph operator H given by:

$$
\begin{aligned}
H : & \mathscr{G}(\mathscr{W}) \rightarrow \mathscr{G}(\mathscr{W} \cup \{\text{top}(S)\}) \\
& H(S) := (C_H, R_H, \gamma_H) \\
& C_H := C_S \cup \{\text{top}(S)\} \\
& R_H := \{r'_u\}_{u \leq n_c},\ r'_u \in \{\text{top}(S)\} \times C_S \\
& \gamma_H(r'_u) := (\text{top}(S), c_u)
\end{aligned}
$$

where $\mathscr{G}(\mathscr{W})$ is the set of all possible graphs constructed from objects belonging to $\mathscr{W}$; $\text{top}(S)$ is a new node that represents the entire system and is called the top node; to represent the hierarchical relation between the system top node $\text{top}(S)$ and its component nodes c_u, the new r'_u edges are directed edges (the direction meaning *system* 'is composed of' *component*; in UML language (Object Management Group, 2015), r'_u are aggregation instances). Figure 2 illustrates the implicit hierarchy present in any graph.

Since any subgraph of a system is also a system, the hierarchical view of successively nested systems is by construct a *rooted tree* and a natural representation of a hierarchy

The concepts defined in this section and their relationships are shown in Figure 1 of the appendix.

1.2 The description of the system by an observer

We have yet to provide a means to *describe* elements of our system such that two observers can unambiguously agree on their meaning. Such a description relies on measurement. Measurement itself depends on previous theoretical knowledge, instrument manufacture and on our senses to read and interpret them. All this constitutes a filter between the real world and the knowledge we have of it: measurement is *subjective*. In order to communicate our experience of real-world objects, we must specify a description that can be shared between different observers. Science can then proceed because observers then speak the same language. To this end, we

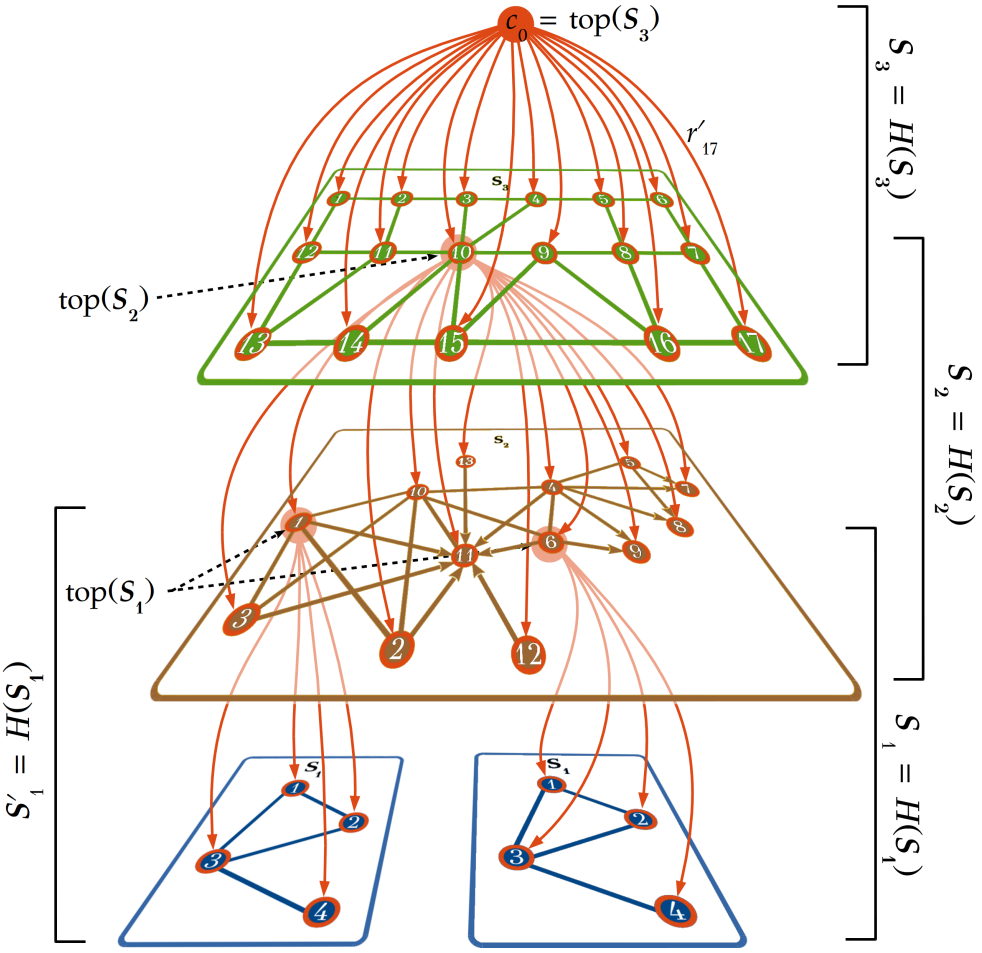


Figure 2: Illustration of the hierarchical view of systems S_1 (animal), S_2 (ecosystem), and S_3 (landscape) from Figure 1 according to Definition 3. The hierarchical views, labelled S'_1, S''_1, S'_2 and S'_3, are shown in orange. The top operator enables nesting systems into a hierarchy, by making it possible for a component in the upper-level system to represent a full lower-level system.

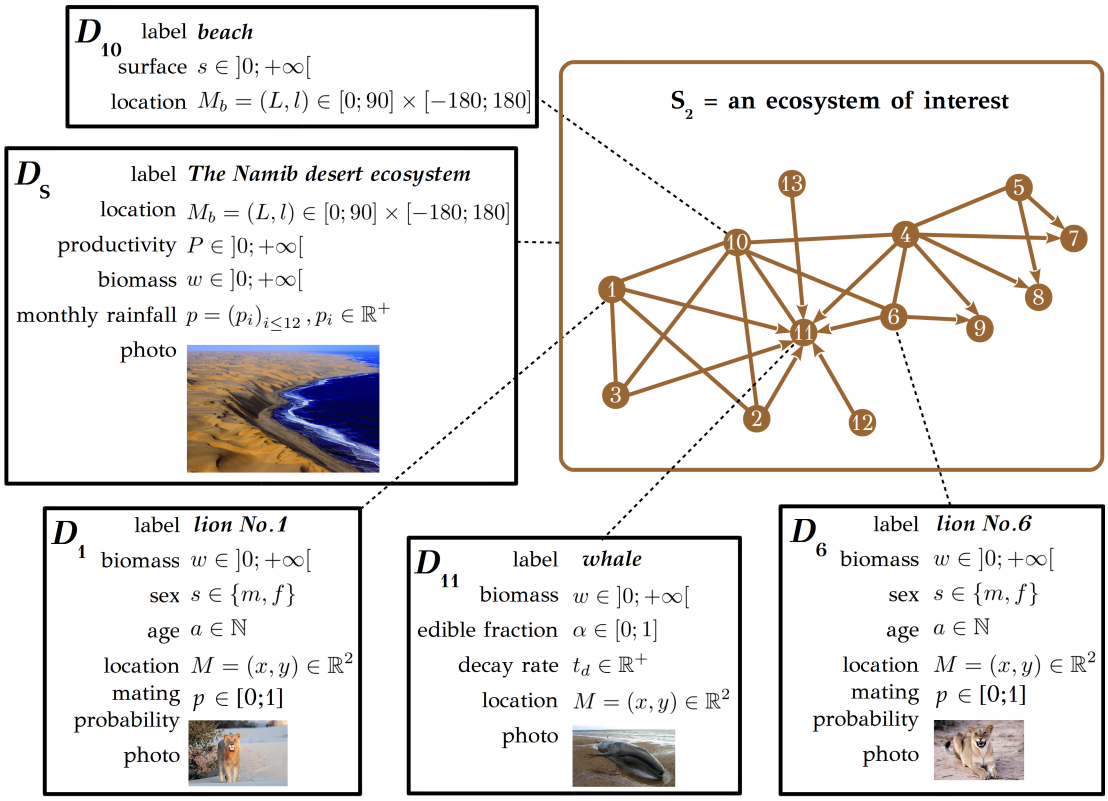


Figure 3: Descriptions of system components using Definitions 4-5. Example based on system S_2 of Figure 1. Descriptions, each made of a list of descriptors, are shown for four components and the system itself. For example, assuming that 'lion No.1' is the 1st component of the system, its description is $D_1(c_1) = (d_1 = \text{id},\ d_2 = w,\ d_3 = s,\ d_4 = a,\ d_5 = M,\ d_6 = p,\ d_7 = \square)$, where id is the label and $\square$ represents a bitmap (for the photo). The description of the whole system (The Namib desert ecosystem) is $D_S = ((\text{id}), (L,l), P, w, (p_1 \ldots p_{12}))$. The structural description (Definition 6) of the system is $D_S = (13, (1,..,13), 23, (1,..,23), ((1,1,2), (2,1,3), .., (23,13,11)))$. Pictures were downloaded from low-resolution thumbnails on the internet.

propose to attach *descriptors* (labels, indices, categories, quantities, texts, images, sounds etc., with appropriate units or other technical metadata) to system elements. For notational simplicity, we assume that all descriptors can be represented numerically.

Proposition 2. *An* observer $\mathcal{O}$ *is an agent (person, device) able to produce a* description D *of a hierarchical system, component, or relation. Descriptions constitute the only way for observers to communicate about a system.*

Definition 4. Description.

1. The *description* $D(e_w)$ of a system element $e_w \in \mathscr{E}$ is a n-tuple of descriptors d_x defined by their use and interest for the observer (cf. Definition 5):

$$D(e_w) := (d_x)_{x \leq n_D}$$

where the index w is the element index in system S. n_D is the *dimension* of the description.

2. Similarly, we define the description of the system $D(S)$, with the convention that $D(S) = D(\text{top}(S))$.

Definition 5. A *descriptor* $(d, \mathcal{R})$ of a system element e_w is the association of:

1. d: a n-tuple of numeric variables:

$$d(e_w) := (z_y)_{y \leq n_d}, z_y \in \mathbb{R}$$

n_d is called the *dimension* of the descriptor.

2. $\mathcal{R}$: an interpretation rule, which associates meaning to d for an observer $\mathcal{O}$. Examples of such rules include measurement units, particular algebraic rules for combining values of different descriptors, coding rules, and any other metadata applying to d.

We assume the rule $\mathcal{R}$ is not necessarily expressed in mathematical terms. For this reason, in what follows we denote the descriptor by d only. The particular meaning of each descriptor within a description is apparent in our notations by considering that the description cannot be written simply as a single concatenated list of numeric variables.

See Figure 3 for an illustration of definitions 4 and 5. The description of an element depends on the observer, who has a particular purpose to guide her/his choice. As a result, many descriptions of the same element or system may exist, representing the observer's interpretation of the system.

We now return to Proposition 1, to distinguish between the real world and its representation. Following Tansley (1935), a system is a 'mental isolate' of the *real* world. As system elements belong to the *real* world, it means Proposition 1 is just a *description* sensu Definition 4 of the system. Proposition 1 describes the *structure* of a hierarchical system in mathematical terms. By structure, we mean how components and relations are linked to form an organised system, without any assumption as to what components and relations really are. We acknowledge this through the following re-interpretation of Proposition 1:

Definition 6. The *structural* description of a hierarchical system is

$$D\left(S\right) = \left(d_1, d_2, d_3, d_4, d_5\right)$$

where

$$\begin{aligned}
d_1 &= n_c \\
d_2 &= \left(\mathrm{id}_C\left(c_u\right)\right)_{u \leq n_c} \\
d_3 &= n_r \\
d_4 &= \left(\mathrm{id}_R\left(r_v\right)\right)_{v \leq n_r} \\
d_5 &= \left(\mathrm{id}_R\left(r_v\right), \mathrm{id}_C\left(c_i\right), \mathrm{id}_C\left(c_j\right)\right)_{v \leq n_r,\, \gamma(r_v) = (c_i, c_j)}
\end{aligned}$$

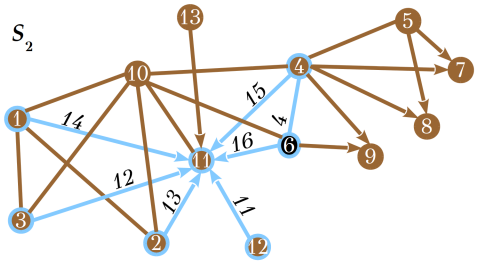
and where $\mathrm{id}_A\left(\ldots\right)$ is the identifier function over set A: it associates a number to a component or relation that is unique over their respective sets C and R. d_1-d_5 are the *standard structural descriptors* of the system.

Although making the difference between the system and its mathematical expression explicit may seem pedantic, it will be useful when considering: (1) dynamic systems; (2) experiments on real-world systems; and (3), systems which elements are partly unknown.

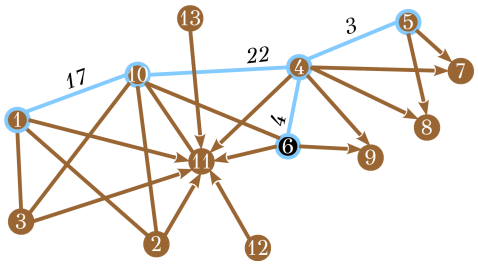
The concepts defined in this section and their relationships are shown in Figure 2 of the appendix.

1.3 The state of a system and its measurement context

If obtaining values for descriptor variables (z_y) (Definition 5) is the act of measurement, then the set of measurements of all descriptors of an element is its *state* at the time of measurement (system dynamics are explored in Section 1.4). In computer science terms, the act of measurement as defined here is an *instantiation*: the descriptor being a class or type definition and the state, an instance of this class with a physical existence somewhere (in computer memory or on a sheet of paper as the case may be). It must be assumed that measurements are *reliable:* if they differ, the element is assumed to have changed. For this to hold, measurements must satisfy some robustness principle, such as ergodicity (Boltzmann, 1871) or the law of great numbers (Bernouilli, 1713), which state that the mean of repeated measurements in the same conditions on the same object will converge asymptotically to the same value. The whole theory of sampling statistics (*e.g.* Cochran, 1977) was developed to solve the problem of how to get a 'representative' measure of a real object or system.



$$l_{2_{1,6}}(c_6) = (C_6 = \{c_6, c_1, c_3, c_2, c_4, c_{12}, c_{11}\},$$
$$R_6 = \{r_4, r_{11}, r_{12}, r_{13}, r_{14}, r_{15}, r_{16}\},$$
$$\gamma_6(r_4) = (c_4, c_6)$$
$$\gamma_6(r_{11}) = (c_{12}, c_{11})$$
$$\ldots)$$



$$l_{6_{1,6}}(c_6) = (C_6 = \{c_6, c_4, c_5, c_{10}, c_1\},$$
$$R_6 = \{r_3, r_4, r_{22}, r_{17}\},$$
$$\gamma_6(r_3) = (c_4, c_5)$$
$$\gamma_6(r_4) = (c_4, c_6)$$
$$\ldots)$$

Figure 4: Two different local contexts (Definition 7) $l_{2_{1,6}}$ (for descriptor 2 (biomass w), observer 1, component 6) and $l_{6_{1,6}}$ (for descriptor 6 (mating probability p), observer 1, component 6), computed for focal component c_6 (Lion 6) of example system S_2 (Figure 1). Indexing according to Figure 5. Brown: the full system graph as on Figure 3; black: the focal component c_6; light blue: the relations and components involved in the local contexts. Local context $l_{2_{1,6}}$ groups all the components and relations needed to compute an increase of biomass for lion c_6 (component and relation meanings are listed in Figure 1) through feeding: c_{11} is the food source, i.e. a stranded whale; c_1-c_4 are other lions, one of them (c_4) belonging to the same pride as c_6; c_{12} is a flock of seagulls attracted by the carcasse. It is easy to imagine that the share of the food available to the focal lion c_6 can be the result of a relatively complex computation / measurement protocol involving all these components and their relations. The newly computed biomass w for lion c_6 is its partial state (Definition 9) $\sigma_{2_{1,6}}$ relative to $l_{2_{1,6}}$ for descriptor $d_{2_{1,6}}$. $l_{2_{1,\ldots}}$ is thus the local context relevant to compute biomass increase for lions. Local context $l_{6_{1,6}}$ groups all the components and relations needed to compute probability of mating between lions: (female) lion c_6 is member of a pride comprising one male $c_4$and another female c_5; but while feeding on the beach c_{10} it may be approached by male lion c_1, member of a different pride, and interesting behavioural interactions may result in birth of lions, *i.e.* changes in graph structure (Section 1.4). $l_{6_{1,\ldots}}$ is thus the local context relevant to compute probability of Lion 6 mating, i.e. a potential structural change in the system S_2.

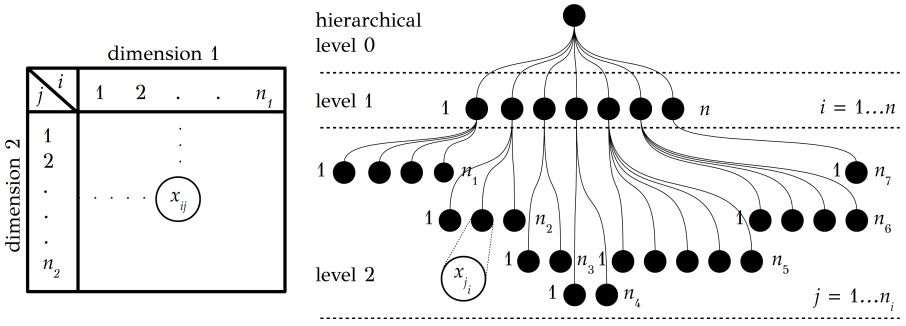


Figure 5: Indexing rules. Left, dimensional indexing: each index represents a dimension of a data structure, hence the notation x_{ij}. Right, hierarchical indexing: each lower level index in a hierarchy tree depends on the previous level index, hence the notation x_{j_i}.

Although the descriptor is attached to an element (Definition 5), hereafter called the *focal* element, its instantiation may require additional knowledge such as the location of the focal element within the graph representing the system. For example: many plant competition models are based on measures such as the number of neighbour plants in a circle of fixed radius centred on the focal plant; the amount of food gained by an animal may depend on the size of its social group or its position in the social hierarchy; water run-off on a given landscape unit may depend, not just on its elevation, but on its relative position within the landscape (slope, valley, ridge top). All these focal element descriptors depend on its relations with other elements as specified by the system incidence function (Proposition 1). The sub-set of elements required to compute the state of a descriptor of a focal element we call a *local context.*

Definition 7. The *local context* l of a *focal* element e_w is the result of applying a graph operator L_w on system S, that returns a *connected* (Diestel, 2000) subgraphS' of S containing e_w:

$$
\begin{aligned}
& l\left(e_w\right) = S' := L_w\left(S\right) \\
& L_w\left(S\right) := \left(C_w, R_w, \gamma_w\right) \\
& C_w \subset C_S \\
& R_w \subset R_S \\
& \gamma_w\left[R_w\right] \subset \gamma_S\left[R_S\right] \\
& \nexists c_i \in C_w \,|\, \zeta\left(c_i\right) = \emptyset
\end{aligned}
$$

where $f\left[A\right]$ is the image of set A by function f, and with the additional constraints:

1. if e_w is a component $e_w = c_u \in C_S$:

$$c_u \in C_u$$

2. if e_w is a relation $e_w = r_v \in R_S$:

$$
\begin{aligned}
& r_v \in R_v \\
& \gamma_v\left(r_v\right) = \left(c_1, c_2\right) \Rightarrow c_1 \in C_v, c_2 \in C_v
\end{aligned}
$$

There may be a unique subgraph for each descriptor. Figure 4 provides an example of two local contexts for Lion 6 in the Namib desert ecosystem of Figure 3: one is used to calculate the biomass of Lion 6 and the second to calculate its mating probability.

Emergence is about comparing microscopic and macroscopic states of the system. The microscopic state involves computing the states of all system elements and combining them in some way into a system-level state. From Definitions 5 and 7, it should be straightforward to compute the state of a single system element. However, much confusion may arise if we consider that: (1) elements may have different descriptions, *i.e.* different sets

of descriptors, with different sets of variables; (2) different observers may define different descriptions, and even a single observer may wish to work with different descriptions of the same element; (3) the method used to compute a local context may depend both on the particular element considered and on the particular descriptor. To denote the dependency on all these families of objects (observers, elements, descriptor variables), we will use rigorous indexing rules (Figure 5): D_{mw} indicates that the description of an element e_w depends on the element's individuality within the system (index w) and on the observer or the observer's choice (index m) (Figure 4). Since the descriptors are nested within descriptions (Definition 4), they will be denoted by $d_{x_{mw}}$, and since descriptor variables are nested within descriptors, they will be denoted by $z_{y_{x_{mw}}}$. Dimensions of descriptions and descriptors now become $n_{D_{mw}}$ and $n_{d_{x_{mw}}}$. The local context being used to measure or compute the state of an element and for a particular descriptor, must match the descriptor, and hence is denoted by $l_{x_{mw}}$. Finally, it makes sense to group the local contexts which will be used to compute the description of a single element:

Definition 8. A *context* q is a n-tuple of local contexts applying to the same focal element:

$$q\left(e_w\right) = \left(l_i\left(e_w\right)\right)_{i\in\mathbb{N}}$$

In the Figure 4 example, we may associate the two local contexts defined for computing biomass and mating probability of Lion 6 into the same context: $q\left(c_6\right) = \left(l_{2_{1,6}}\left(c_6\right), l_{6_{1,6}}\left(c_6\right)\right)$.

With this settled, we can now compute element and system states:

Definition 9. The *partial state* $\sigma_{x_{mw}}$ of an element e_w, *relative to a local context* $l_{x_{mw}}$, *for descriptor* $d_{x_{mw}}$ associates an element and a local context with a descriptor instance:

$$\begin{array}{rccc} \sigma_{x_{mw}}: & \mathscr{E}\times\mathscr{G}\left(\mathscr{W}\right) & \rightarrow & \mathbb{R}^{n_{d_{x_{mw}}}} \\ & \left(e_w, l_{x_{mw}}\left(e_w\right)\right) & \rightarrow & d_{x_{mw}} = \left(z_{y_{x_{mw}}}\right)_{y\leq n_{d_{x_{mw}}}} \end{array}$$

where $w \leq n_e$ is the element index, m the observer index, $x \leq n_{D_{mw}}$ the descriptor index in description D_{mw}, $y \leq n_{d_{x_{mw}}}$ the variable index within descriptor $d_{x_{mw}}$.

Definition 10. The *local state* σ_{mw} of an element e_w *relative to a context* q_{mw}, for description D_{mw} is defined as:

$$\begin{array}{rccc} \sigma_{mw}: & \mathscr{E}\times\left(\mathscr{G}\left(\mathscr{W}\right)\right)^{n_{D_{mw}}} & \rightarrow & \mathbb{R}^{n_{mw}} \\ & \left(e_w, q_{mw}\right) & \rightarrow & D_{mw}\left(e_w\right) = \left(\sigma_{x_{mw}}\right)_{x_{mw}\leq n_{D_{mw}}} = \left(\left(z_y\right)_{y\leq n_{d_x}}\right)_{x\leq n_{D_{mw}}} \end{array}$$

where $n_{mw} = \sum_{x_{mw}=1}^{n_{D_{mw}}} n_{d_{x_{mw}}}$ is the total number of descriptor variables in description D_{mw}; $q_{mw} = \left(\left(l_x\right)_{x\leq n_{D_{mw}}}\right)_{mw}$ is the context grouping all the local contexts $l_{x_{mw}}$ used to compute partial states $\sigma_{x_{mw}}$ of descriptors $d_{x_{mw}}$ of description D_{mw}.

Among all possible subgraphs required to compute a local context, two extremes exist: those that require no other elements and those that require all other elements to be considered.

Definition 11. The *minimal local context* l_0 of a focal element e_w is the local context of this element in isolation:

1. for a component:

$$l_0\left(c_u\right) = \left(\left\{c_u\right\}, \emptyset, 0\right)$$

where $\emptyset$ is the empty set and 0 the null function

   (a) for a relation:

$$l_0\left(r_v\right) = \left(\left\{c_1, c_2\right\}, \left\{r_v\right\}, \gamma\left(r_v\right) = \left(c_1, c_2\right)\right)$$

As a consequence, the minimal context q_0 is a trivial n-tuple repeating the minimal local context $n_{D_{mw}}$ times.

Definition 12. The *maximal local context* l_S of a focal element e_w is the whole system S:

$$\forall e_w \in S,\, l_S\left(e_w\right) = S$$

By construction (Definition 3), when choosing a system we isolate it from the rest of the world. Therefore, no local context, other than the minimal local context is relevant at the system level:

Definition 13. The *system local context* is the minimal local context of $\text{top}\left(S\right)$:

$$l\left(\text{top}\left(S\right)\right) := l_0\left(\text{top}\left(S\right)\right) = \left(\left\{\text{top}\left(S\right)\right\}, \emptyset, 0\right)$$

Definition 14. The *disconnected state* σ_w of an element e_w is its local state relative to the minimal context q_0:

$$\sigma_w\left(e_w\right) = D_{0w}\left(e_w\right) = \sigma_{0w}\left(e_w, q_0\right)$$

We can now define the system *microscopic* and *macroscopic* states:

Definition 15. The *macroscopic state*, or *macro-state*, of a system $\Omega\left(S\right)$ is the disconnected state of the component representing the whole system in its hierarchical view (Definition 3):

$$\Omega\left(S\right) = \sigma_0\left(\text{top}\left(S\right)\right)$$

assuming the component index 0 is attributed to $\text{top}\left(S\right)$. Even if the observer has not defined any description for the system as a whole, the structural description (Definition 6) always exists. Therefore, any system has at least one macro-state.

Definition 16. The *microscopic state*, or *micro-state*, of a system is defined as the n-tuple:

$$\omega_m\left(S\right) = \left(\sigma_{mw}\left(e_w\right)\right)_{w \leq n_e}$$

Contrary to the macro-state, the micro-state is not a single description taking some measured values, but a n-tuple of descriptions of all the elements of the system, measured or computed for a particular n-tuple of local contexts that depends on observer m. There may be many micro-states for the same system, depending on the observer's choices.

Definition 17. The *disconnected micro-state* ω_0 of the system S is the micro-state computed using the minimal context q_0 for all elements:

$$\omega_0\left(S\right) = \left(\sigma_w\left(e_w\right)\right)_{w \leq n_e}$$

With these definitions, the emergence problem can be restated simply as: is there a function (to keep it simple; it may be an algorithm or some more elaborate transformation) that links the macro-state and the micro-state(s) of a system? Emergence will depend on the existence and nature of this function.

The concepts defined in this section and their relationships are shown in Figure 3 of the appendix.

1.4 System dynamics

So far, only *static* systems have been considered. We now turn to dynamic systems: systems that change over time. We follow the definition of a dynamic graph by Harary & Gupta (1997).

Definition 18. A *dynamic hierarchical system* is defined as:

$$S\left(t + \Delta t\right) = g\left(S\left(t\right), \Delta t\right)$$

where: t represents time; Δt a small amount of time over which a change can be observed; and g, some function.

Two kinds of change may affect the system: changes in structure and changes in state.

1. Changes in *structure* involve creation and deletion of components and relations. We define a structural change as a triplet of sets:

   Definition 19. A *structural change* in the system between times t and $t + \Delta t$ is defined as

   $$\Delta S\left(t, \Delta t\right) := \left(\Delta S^{+}\left(t, \Delta t\right), \Delta S^{-}\left(t, \Delta t\right), \Delta S^{\gamma}\left(t, \Delta t\right)\right)$$

   where

   - Δ denotes any, possibly large, change;
   - $\Delta S^{+}\left(t, \Delta t\right)$ is the set of elements added to the system (*creation set*);
   - $\Delta S^{-}\left(t, \Delta t\right)$ is the set of elements removed from the system (*deletion set*);
   - $\Delta S^{\gamma}\left(t, \Delta t\right)$ is the set of incidences of new relations over components (*new branching set*) at time t over Δt.

   We further define, from Proposition 1, and omitting $(t, \Delta t)$ for notational simplicity, ΔC^{+}, ΔC^{-},ΔR^{+} and ΔR^{-}, as, respectively, the set of components added to and removed from, and the set of relations added to and removed from the system between time t and $t + \Delta t$. By construction $\Delta S^{+} = \Delta C^{+} \cup \Delta R^{+}$ and $\Delta S^{-} = \Delta C^{-} \cup \Delta R^{-}$. With these sets defined, the new structure of the system is computed from

   $$\begin{cases} C\left(t + \Delta t\right) = C\left(t\right) \cup \Delta C^{+} \setminus \Delta C^{-} \\ R\left(t + \Delta t\right) = R\left(t\right) \cup \Delta R^{+} \setminus \Delta R^{-} \\ \gamma\left[R\left(t + \Delta t\right)\right] = \gamma\left[R\left(t\right) \setminus \Delta R^{-}\right] \cup \Delta S^{\gamma} \end{cases}$$

   where $\setminus$ stands for set substraction.

   Since graph elements are discrete objects, a structural change can be reduced to a set of *elementary structural changes* of at most four types. By elementary, we mean *atomic*, *i.e.* a change that cannot be smaller and occurs instantaneously. We use δ to denote such changes.

   Definition 20. An *elementary structural change* in system $\delta S = (\delta S^{+}, \delta S^{-}, \delta S^{\gamma})$, is one of the four following changes

   (a) **deletion of a single relation** r_v – The deletion set contains only a single relation $\delta S^{-} = \delta R^{-} = \{r_v\} \subset R\left(t\right)$:

   $$\delta S_r^{-} = \left(\emptyset, \{r_v\}, \emptyset\right)$$

   (b) **creation of a new relation** r_v **between two existing components** c_i **and** c_j – The creation set contains only a single relation to add to the system $\delta S^{+} = \delta R^{+} = \{r_v\}$. In addition, the components to be connected by this relation must be specified and the incidence function modified accordingly:

   $$\delta S_r^{+} = \left(\{r_v\}, \emptyset, \{(c_i, c_j)\}\right)$$

   with $c_i \in C\left(t\right)$ and $c_j \in C\left(t\right)$.

   (c) **deletion of an existing component** c_u – The deletion set contains the component plus its connection set: $\delta C^{-} = \{c_u\} \subset C\left(t\right)$ and $\delta R^{-} = \zeta\left(c_u\right)$:

   $$\delta S_c^{-} = \left(\emptyset, \{c_u\} \cup \zeta\left(c_u\right), \emptyset\right)$$

   (d) **creation of a new component** c_u – The creation set contains the new component plus a new connection set relating it to other already existing components of the graph (possibly none): $\delta C^{+} =$

$\{c_u\}$, with $c_u \notin C(t)$, and $\delta R^+ = \zeta(c_u)$. ΔR^+ might be empty if the new component is disconnected from the rest of the graph. Information must be provided on how the new component is to be connected to other components. The new branching set is $\delta S^\gamma = \gamma[\zeta(c_u)]$. The structural change becomes:

$$\delta S_c^+ = (\{c_u\} \cup \zeta(c_u), \emptyset, \gamma[\zeta(c_u)])$$

Adding and deleting components and relations atomically in a graph is, therefore, not as straightforward as it might seem. The operation of removing a relation or adding a disconnected component acts on only one object. In all other cases (2, 3 and 4 above), additional information is required.

2. Changes in state affect values of descriptors of components, relations, or the system itself:

Definition 21. A *state change* $\Delta\sigma_{mw}(t, \Delta t)$ for an element $e_w(t)$ relative to the context $q_{mw}(t)$, is defined as a change in numeric values of its descriptor variables $z_{y_{x_{mw}}}$ between two successive local states σ_{mw} obtained at times t and $t + \Delta t$ (Definition 10) :

$$\Delta\sigma_{mw}(e_w(t), q_{mw})(t, \Delta t) := \left(\left(z_{y_{x_{mw}}}(t + \Delta t) - z_{y_{x_{mw}}}(t)\right)_{y_{x_{mw}} \leq n_{d_{x_{mw}}}}\right)_{x_{mw} \leq n_{D_{mw}}}$$

A similar definition applies to the system as a whole by using its description.

Definition 22. We define $d\sigma_{mw}$, $d\omega_m$, and $d\Omega$ as, respectively: an *infinitesimal* change in local, micro- and macro-state. Infinitesimal means the smallest possibly observable, measurable, or computable change in state as dt approaches zero.

Definition 23. We define an *elementary change in system* dS as either an atomic structural change δS, or an infinitesimal state change ($d\omega_m$ or $d\Omega$). In both cases, dS is the smallest possible change a system can undergo.

Changes in structure are discrete by nature (section 1.4), because components and relations are discrete system elements. Hence discrete-time and event-driven models may better manage structural changes (Zeigler et al., 2000). Changes in descriptor values may be discrete or continuous. It follows that models describing the dynamics of a hierarchical system must allow for both continuous and discrete change. Coupling differential equations with a discrete-event logic is a difficult task that has been explored mainly by simulation specialists (*e.g.* Duboz et al., 2003).

The concepts defined in this section and their relationships are shown in Figure 4 of the appendix.

1.5 Causality

It is not our aim here to enter a philosophical debate addressing the reductionnist *vs.* holistic dispute, or the mechanistic *vs.* empirical modelling issue. In this framework, a cause is what produces a change in system state. In general, dynamics, as an observed succession of changes, does not inform about the underlying causes. System dynamics may be described in many ways, *e.g.* by a set of differential equations linking system variables, stochastic markovian transitions, deterministic rules, etc. – there is usually some debate about whether these methods represent real causes or are just phenomenological descriptions of dynamic changes. However, a recurrence equation such as that of Definition 18 implies a form of causality, as it assumes that the state of the system at time $t + dt$ can be computed from its state at time t. This satisfies a *necessary* condition for causality: a cause must, by definition, precede the effect. We propose here to assume that this condition is also *sufficient* under the condition that dt represents the smallest amount of time at which we are able to see a difference in system states. In this case, nothing can be said about what happened between t and $t + dt$, and rather than saying that either (1) some complicated speculative process occuring within this time period caused the change, or (2) what we observe is only correlation, we *assume* that the state at time t caused the state at time $t + dt$. This is an application of the parsimony principle: in the absence of better information on the system functioning, *at a chosen time grain* the recurrence equation of Definition 18 describes some form of causality that we call *apparent*:

Algorithm 1 Pseudocode for the propagation of changes in an atomic system (Definition 20).

```
generateChange(S(t),t,dt) {
  return(dS(t,dt))
}
changeSystem(S,dS,t,dt) {
  S(t+dt) := some_computation(S(t),dS(t,dt))
}
propagateChanges(S,t_0) {
  t := t_0
  while (stopping_condition(t,S) is false) {
    ... compute dt ...
    dS := generateChange(S,t,dt)
    changeSystem(S,dS,t,dt)
    t := t+dt
  }
}
```

Proposition 3. Apparent causality*: If S is a dynamic system sensu Definition 18 and if dt is an infinitesimal atomic duration (called* time grain*) then we postulate that $S(t)$ is the* apparent cause *of $S(t+dt)$* at time grain *dt.*

We consider that *apparent* cause is the best approximation of real cause given the information we have on the system.

Often, the dynamics of a system is represented by a set of coupled recurrence equations (*e.g.* differential equations) applying to a set of variables. In our framework, such a set of variables is what we have called the description (Definition 4) of a component, relation or system. In other words, this common use of recurrence equations does not apply to our hierarchical system, but to a simpler system or object:

Definition 24. *Atomic system.* A system S is *atomic* if it cannot be subdivided into elements. Such a system is non-hierarchical.

An atomic system can have a description (Definition 4).

The recurrence equation applies to an atomic system as a whole. As a consequence, changes in system state result from each other: if we link changes through the causal relation implied by the recurrence equation, we will obtain a simple *chain of change events.* This is sometimes called *linear causality*[1], as the common reductionnist view is to identify changes with causes, specially as $dt \to 0$. For a better understanding and later comparison, we provide the trivial pseudocode (Algorithm 1) for the case of the atomic system.

In a hierarchical system, this simple scheme no longer applies, as system elements may be subject to their own dynamic recurrence equation. Let us first consider a graph which only undergoes state changes (Definition 21) but no structural changes (Definition 19) and, for simplicity, that these changes are driven by differential equations (among other possible methods) to describe a dynamic. Can we rewrite the recurrence equation of Definition 18 in order to account for graph structure? It may be possible to re-organise it into a system of equations applying to components and relations. But this ignores graph structure and treats the graph as an unstructured list of elements. It is difficult to imagine a graph component affecting another without there being a relation between them – otherwise, by definition, the graph is not a correct representation of the observer's system. To be consistent with our initial assumption (Proposition 1), that a graph is the correct way to represent a hierarchical system, we postulate that the graph architecture must be used in dynamics computations in a meaningful way. We propose the following to impose this consistency:

Proposition 4. Locality of causes*: in a* dynamic hierarchical system*,* elementary *dynamic changes are produced and successively spread following system relations and components*, i.e. *locally.*

[1] Caution: linear here does not refer to its usual mathematical meaning.

Algorithm 2 Pseudocode for propagation of elementary changes over the system graph.

```
generateChange(e_c, σ_ck(t), l_k(t), t, dt) {
  return(de_c(t, dt))
}
neighbours(c_a) {
  return({r_j} | γ(r_j) = (c_i, c_a)_{i≤n_c})
}
neighbours(r_b) {
  return({c_a} | γ(r_b) = (c_i, c_a))
}
// recursive
updateElement(e_c, k) {
  newContext := some_computation(l_k(e_c))
  newState := some_other_computation(newContext)
  if (newContext ≠ l_k(e_c) or newState ≠ σ_ck(e_c)) {
    l_k(e_c) := newContext
    σ_ck(e_c) := newState
    for (e_d in neighbours(e_c)) {
      updateElement(e_d, k)
    }
  }
}
changeElement(e_c, k, de_c, t, dt) {
  l_k(e_c(t + dt)) := some_computation(l_k(e_c(t)), de_c(t, dt))
  σ_ck(t + dt) := some_other_computation(l_k(e_c(t + dt)))
  for (e_d in neighbours(e_c)) {
    updateElement(e_d, k)
  }
}
propagateChanges(S, t_0, k) {
  t := t_0
  while (stopping_condition(t, S) is false) {
    ... compute dt ...
    for (c := 1 to n_e) {
      de_c := generateChange(e_c, σ_ck(t), l_k(t), t, dt)
      changeElement(e_c, k, de_c, t, dt)
    }
    t := t + dt
  }
}
```

This means that the dynamic recurrence equation of Definition 18 has a causal meaning only if applied at graph element level; the system-level recurrence equation will be the result of these local dynamic equations. The algorithm needed to compute changes and propagate them over the graph becomes more elaborate, involving recursion (Algorithm 2).

At each granular time step, elementary change (Definition 23) may arise due to element local context (Definition 7) and state (Definition 10). Focusing on only one of these changes, once applied, it will in turn at the next granular time step trigger its elements' neighbours (outgoing relations for components, end component for relations) to update their own local context and state accordingly; if this results in a change of local context and state, the process is further propagated to the graph network until there are no more changes in local context and state. This sequence describes the spread of changes from only one element. However, many changes could arise simultaneously from many different elements, generating overlapping spreading networks of changes, generating possible conflicts and interactions between them. The algorithm does not specify how changes are generated: it merely assumes changes may be generated using the information available to the observer of a given system element, *i.e.* its local context and state. With an appropriate method to generate

changes, the system might start evolving simply due to its initial micro-state.

In Algorithm 2, it is theoretically possible to find the changes that have been caused by other past changes by tracking the succession of local context and state updates that caused the `generateChange()` function to generate it. Elementary changes can then be chained, possibly forming a network of changes *apparently* caused by each other (Proposition 3). This is a long way from the simplicity of linear causality. For the analysis of system dynamics, it is helpful to define the following graph:

Definition 25. A *dynamic change network* of a dynamic hierarchical system $K(S)$ is a directed graph describing the causal pathways between elementary changes of a realised system dynamics $(S(t))_{t<\infty}$ under the assumption of *apparent causality*. Its components and relations are defined as follows:

1. To each node $v_i \in C_K$ is attached a description with four descriptors :

$$D_K(v_i) = (d_{i1}, d_{i2}, d_{i3}, d_{i4})$$

where

$$\begin{aligned} d_{i1} &= \mathrm{id}_{C_K}(v_i), v_i \in C_K \\ d_{i2} &= \mathrm{id}_{C_S}(e_w), e_w \in E_S \\ d_{i3} &= t \\ d_{i4} &= dS \\ d_{i5} &= dt \end{aligned}$$

d_{i1} is the unique identifier of node v_i in graph $K(S)$, $E_S = C_S \cup R_S$ is the set of elements of S , d_{i2} the unique identifier of element e_w in system S, d_{i3} is the time at which the change occured, d_{i4} is the elementary change in S, and d_{i5} is the time grain.

2. The *directed* edges of $K(S)$ reflect the direction of apparent causality (Proposition 3), $a \to b$ meaning that a caused b. They have no descriptor variable except their unique identifier. They link nodes that differ at successive times separated by the time grain dt:

$$\begin{array}{rccc} \gamma_K : & R_K & \to & C_K \times C_K \\ & r_v & \to & (v_i, v_j) \mid (d_{i3} = t) \text{ and } (d_{j3} = t + dt) \end{array}$$

The dynamic change network of a dynamic system provides an image of the succession of changes that occur during a particular dynamic (Figure 6). Since time increases on every edge, there is no possibility to loop back in such a network, which is thus an *oriented graph* (Diestel, 2000). However, since the causal pathways may involve loops (system elements affected more than once over the time course of the dynamics) and conflicts (more than one change occurring on one element at the same time), it may be useful to make them visible by 'projecting' the dynamic change network back on the initial system:

Definition 26. A *causal network* of a dynamic system $Q(S)$ is a graph operator mapping a system dynamic change network $K(S)$ on the system S, where there is one node c'_w for every element e_w of S :

$$C_Q = \{c'_w\}_{w \leq n_e}$$

and the *directed* edges are build by 'merging' all edges of $K(S)$ where start and end nodes have the same d_{i2}

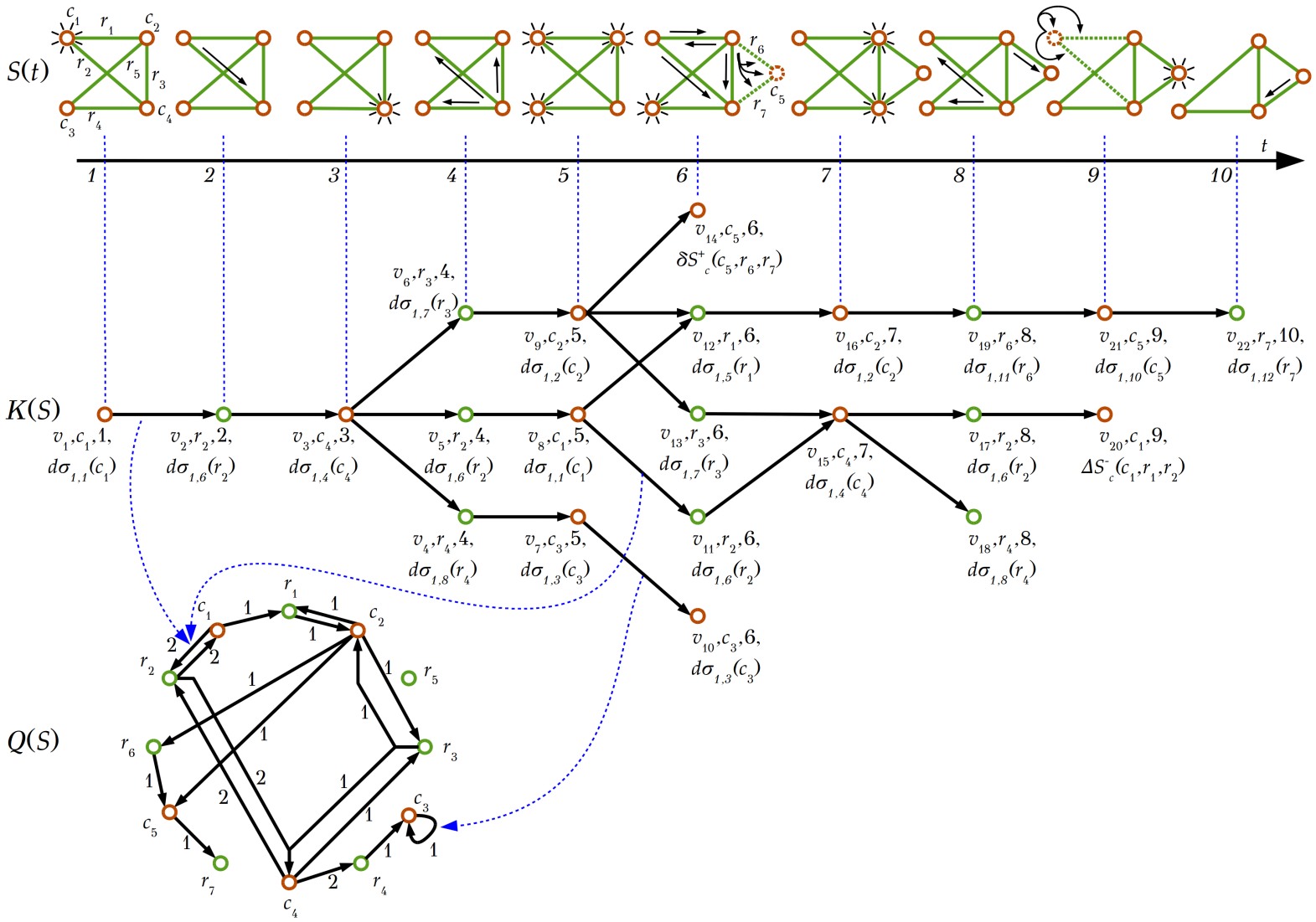


Figure 6: Example of a dynamic change network $K(S)$ and its causal network $Q(S)$ associated to a simple system S. These networks reflect the cascade of changes occuring after a single initial change in one component of the system. *Top*: successive states of S along a dynamics (from time $t = 1$ to 10); rays around a component: change on that component; straight arrow: change on a relation; curved arrows: structural changes, created or deleted elements appearing as dotted lines. *Middle*: the dynamic change network $K(S)$, recording all changes on S; $K(S)$ node: an element of S subject to a change at a given time step; $K(S)$ link: the causal link between the changes according to the locality of cause principle (Proposition 4); the four entries for each node represent the first four descriptors of Definition 25 (the fifth descriptor, time step, is assumed constant, hence not shown). *Bottom:* $K(S)$ is folded over itself to yield the causal network $Q(S)$, where a unique node is kept for every element of S, and where links of $K(S)$ are merged into single links with a multiplicity (Definition 26), shown as a number on every link of $Q(S)$. The dotted arrows between $K(S)$ and $Q(S)$ show how links of $Q(S)$ are built in two particular cases (for the $c_3 \to c_3$ link and the $c_1 \to r_2$ link, which occured twice in $K(S)$, resulting in a multiplicity of 2 in $Q(S)$).

descriptor, *i.e.* refer to the same element of S. We use the merging function h to build the edges:

$$\begin{array}{lccc} h: & R_K^{\mu} & \rightarrow & R_Q \\ & \mathscr{M} = \{r_v \,|\, \gamma_K(r_v) = (v_i, v_j)\} & \rightarrow & r'_v \,|\, \gamma_Q(r'_v) = \left(c'_{w_1}, c'_{w_2}\right) \\ \text{with} & d_{i2}(v_i) = \mathrm{id}_{E_S}\left(c'_{w_1}\right) & & \\ \text{and} & d_{j2}(v_j) = \mathrm{id}_{E_S}\left(c'_{w_2}\right) & & \end{array}$$

We associate to each edge r'_v of $Q(S)$ a descriptor called the *multiplicity* μ, which is the number of edges of $K(S)$ used to construct it: $\mu = |\mathscr{M}|$, where $|A|$ denotes the cardinal of set A. This will indicate which causal pathways are most frequently used over a particular dynamics. By construction, $\sum_{i=1}^{|R_Q|} \mu_i = |R_K|$.

Causal networks contain causal patterns other than simple linear patterns (Figure 7). Due to these patterns, the system dynamics may become extremely difficult to track, predict, or understand from a mechanistic point of view. This issue, characteristic of hierarchical systems, has long been identified as a major feature associated with emergence (e.g. Bedau, 2003; de Haan, 2006; Searle, 1992; Kim, 1992).

The concepts defined in this section and their relationships are shown in Figure 5 of the appendix.

Conclusion: the hierarchical system ontology

While the definitions proposed so far are not necessarily original, we have: (1) made them precise; (2) organised them in a logical progression; and (3), used only the indispensable backbone of the hierarchical system in order to see where and how emergence arises within this set of concepts. Our hope is that this ontology will help to clarify inter-disciplinary discussion of hierarchical systems.

2 Where does emergence lurk?

There is no single, agreed upon, definition of emergence (*e.g.* Bar-Yam, 2004; Bedau, 2003, 2008; Chen et al., 2009; Cotsaftis, 2009a; Crutchfield, 1994; Kim, 1999; Muller, 2003; Searle, 1992), but syntheses have been proposed (Bedau & Humphreys, 2008; de Haan, 2006; Bonabeau & Dessalles, 1997). Three major types of emergence were proposed by de Haan (2006): (1) observer-induced emergence, but without consequences for the system itself, that he called *discovery*; (2) *mechanistic* emergence, with real consequences for the system independent of the observer; and (3) *reflective* emergence, where the observer is actually part of the system and may affect its dynamics (something that Muller (2003) calls *strong* emergence). Most authors develop similar classifications, e.g. (Assad & Packard, 1992) distinguish four types of emergence on a scale of increasing 'objectivity'; terms of 'weak' and 'strong' emergence are often used, although with different definitions (*e.g.* Bedau, 1997; Muller, 2003; Searle, 1992).

From our ontology, we identified four types of emergence that match precise properties of the hierarchical system defined as a graph (Proposition 1). We do not claim these four types are exhaustive, but they address the most often encountered meanings of emergence. One we do not address is the reflective emergence *sensu* de Haan (2006) or strong emergence *sensu* Muller (2003), i.e. the case where the observer is part of the system. In our ontology, the position of the observer relative to the system is irrelevant. We formulate precise definitions of emergence in terms of the system properties they imply. The first three definitions depend on the presence of the observer (Proposition 2) and the last applies only to dynamic systems (Definition 18).

2.1 'Trivial emergence' due to ignoring system *integration*

The definition of complex systems and emergence are often linked. Typically, a system is said to be complex if it displays 'properties that emerge from interactions between its components'. This just means, in our formalism, that a *set* of components is different from a *graph*. The difference lies in the relations and the incidence function (Proposition 1).

One should avoid reducing a hierarchical system merely to the set of its components, but this seems to be common practice (Cotsaftis, 2009b). The popular statement that 'the whole is more than the sum of its parts' is just the (not very surprising) observation that relations are important. Many multi-agent systems focus on the *control problem*: they consider that emergence arises when the interactions between agents are not controlled by a central or external force (systems that Cotsaftis (2009a) calls 'complicated'). This is equivalent, in our case, to systems where the incidence function γ_S is not known a priori, but is generated by system dynamics from individual component behaviours. To distinguish those hierarchical systems that are simple enough to understand without being concerned with relations, from those where this information is important, we define two new terms:

Definition 27. A *flat system* is a system in which the only micro-state is the disconnected micro-state:

$$\forall m \in \{1...M\}, \omega_m(S) = \omega_0(S)$$

Neither local contexts nor interactions play any role in the micro-state computation: the system is 'flat', *i.e.* displays no structure. It is nothing more than the set or the disconnected graph of its components: $S = \left(\{c_u\}_{u \leq n_c}, \emptyset, 0\right) \equiv \{c_u\}_{u \leq n_c}$.

On the example of Figure 4, the flat micro-state of S_2 would ignore the local contexts $l_{2_{1,6}}$ and $l_{6_{1,6}}$. In the case of $l_{2_{1,6}}$, it means that access of Lion 6 to food would not depend on the presence of other lions. This is synonymous to an absence of competition for food and comparing $\omega_0(S_2)$ and $\omega_1(S_2)$ would be a measure of this competition.

Definition 28. An *integrated system* is a system which verifies

$$\exists m \in \{1...M\} \mid \omega_m(S) \neq \omega_0(S)$$

Since we can compute the disconnected micro-state (Definition 17) of any system, we can measure the importance of relations, which we call *integration* $\mathscr{I}$ of the system, by comparing its micro-state to its flat micro-state using some measure of distance:

Definition 29. System *integration* $\mathscr{I}_m(S)$ is defined as the distance between $\omega_m(S)$ and $\omega_0(S)$:

$$\mathscr{I}_m(S) = \sum_{w=1}^{n_e} \sum_{x_{mw}=1}^{n_{D_{mw}}} \sum_{y_{x_{mw}}=1}^{n_{d_{x_{mw}}}} \left\| z_{y_{x_{mw}}} - z_{y_{x_{0w}}} \right\|$$

where $\|\ldots\|$ denotes some distance defined on $\mathbb{R}^+$. For the flat system, $\mathscr{I}_0(S) = 0$.

We can then state:

Definition 30. A system is said to display *trivial emergence* if

$$\exists m \in \{1...M\} \mid \mathscr{I}_m(S) > 0$$

When the terms 'complex system' and 'emergence' are used in this usual, 'naive' sense, we suggest using definitions 28 & 30 instead, as those both refer to some observer surprise at novelty which is not really new: it's just that an important part of a system has been ignored. Here, trivial emergence is an indication of unaccounted for system integration, *i.e.* the importance of the relations and incidence functions in computing the system micro-state. By construction, any non-flat hierarchical system has trivial emergence, which makes this concept uninformative within the present formalism.

2.2 Emergence due to the observer's choice of descriptors: discovery

Even after accounting for system integration, observers may build a model that, *to the best of their knowledge*, represents accurately their understanding of the system under study. However, after further analysis, they find

that the behaviour of their model and the real system diverge. The observer then realises that the model may be revised to reduce this discrepancy. This is what Bonabeau & Dessalles (1997) call *emergence relative to a model*: the system displays an unexpected property, that was not observed when the model was constructed. This emergence is *observer-* (model-) *dependent* and *transient*, as it will disappear when the model is revised to account for the formerly emergent property. Should we use the name 'emergence' to only differentiate between unexpected and expected results in the routine process of experimental science? We agree with Crutchfield (1994) in calling this kind of emergence *discovery*, which implies the idea of being part of a dynamic process. Contrary to emergence, discovery is an event in time. Before the discovery, the unexpected properties seem emergent simply because they are unexplained; after the discovery, with the improved model of the system, their emergent character vanishes.

Definition 31. *Discovery* is defined as an event (at step 6) in the following process:

1. an observer $\mathcal{O}$ of a system S expects the system dynamics to match some predefined *pattern, i.e.* a particular time series of states $\theta = (\Omega(t), \omega_m(t))_{t<\infty}$, where m denotes the method chosen by the observer to generate the micro-states, according to some previous knowledge;
2. $\mathcal{O}$ chooses system and element descriptions D_S and $(D_{mw})_{w\leq n_e}$assumed to appropriately describe the system in order for its dynamics to display the pattern;
3. using some measurement or computation technique, $\mathcal{O}$ generates a *realised* dynamic series of system states $\hat{\theta} = \left(\hat{\Omega}(t), \hat{\omega}_m(t)\right)_{t<\infty}$;
4. $\mathcal{O}$ uses an *error function* $\phi\left(\theta, \hat{\theta}\right)$ to compute a distance between the realised system dynamics and the expected pattern;
5. based on some tolerance threshold τ, $\mathcal{O}$ then decides that the realised dynamics does not match the expected pattern when $\phi\left(\theta, \hat{\theta}\right) > \tau \geq 0$. We call this event *the observer surprise in front of unexpected results*;
6. Using another set of descriptions $\left\{D'_S, (D'_{mw})_{w\leq n_e}\right\}$, and repeating steps 3-4, we call *discovery of a better description of the system* the fact that $\phi\left(\theta', \hat{\theta}'\right) < \phi\left(\theta, \hat{\theta}\right)$;
7. Furthermore, when $\phi\left(\theta', \hat{\theta}'\right) < \phi\left(\theta, \hat{\theta}\right)$ but $\phi\left(\theta', \hat{\theta}'\right) > \tau \geq 0$, the discovery is *incomplete* as there is a possibility to further improve system description beyond the current improvement.

If the realised dynamics matches the expected pattern at step 5, or if the observer does not try another set of descriptions at step 6, there is no possibility of discovery.

Let us now imagine that we have discovered all the descriptors needed to understand a system. It may well be that the system and its elements have completely different descriptors, that mean completely different things (in Definition 4, each element can have a different description, *i.e.* a different *set* of descriptors). Bedau (2003) defined *nominal emergence* as the case where system-level properties have no counterpart at the component level because this would not mean anything, like the concept of the speed of an object in a movie has no meaning when individual frames are taken separately. However, he also explains that nominal emergence is a 'first approximation', a very broad concept that only points to a discrepancy between macro- and micro-level descriptions of a system, without further explanation. Here, nominal emergence arises naturally from Definition 4.

Finally, we might find that the macro-state of a system is a much 'simpler' description of the system than its micro-state. Dessalles et al. (2007) formalised this case as 'emergence relative to a model' (but with a different meaning from Bonabeau & Dessalles (1997)). It is based on the observation of a *drop in complexity* (using some mathematical definition of complexity, such as algorithmic complexity) between the higher-level observations of a system behaviour (the macro-state) and the detail of the rules generating those observations (the micro-state).

For example, in J.H. Conway's game of Life (https://en.wikipedia.org/wiki/Conway's _Game_of_Life), one may observe that the patterns called *gliders* reproduce themselves every four timesteps at a certain distance and in a certain direction, which produces the illusion of movement. One can then describe a glider by giving the position of the cells for the first four time steps and say that this sequence continues as long as the glider 'lives'. This is a description of a glider that is in general shorter than listing the positions of the cells that constitute the glider for its entire lifespan. The shorter description requires a macroscopic point of view from the observer to recognise spatial patterns and relate together patterns that are several timesteps apart. The description, being shorter, represents a drop in complexity.

2.3 Emergence due to *computational irreducibility* between macro-state and micro-states

In sections 2.1-2.2, emergence could be attributed to a lack of information about the system. However, if we assume perfect knowledge, emergence may still be manifest as a gap between system macro- and micro-state. We formalise this gap through the following question: is there a function such that the macro-state can be computed from one of the micro-states?

Definition 32. The *upscaling function relative to a context* q_m is defined as:

$$\begin{array}{rccc} f_m: & \mathbb{R}^{n_{m1}} \times \ldots \times \mathbb{R}^{n_{mne}} & \rightarrow & \mathbb{R}^{n_{D_S}} \\ & \omega_m(S) & \rightarrow & \Omega(S) \end{array}$$

f_m allows two nested organizational 'scales' or levels to be related: the system (upper level/scale) and its elements (lower level/scale).

Emergence can be defined from the properties of the upscaling function:

1. f_m exists – According to Kim (1999), there is no emergence if the upscaling function exists. The macro-state simply *results* from its micro-state(s). It is nothing more than a computation based on the system elements, no matter how convoluted that computation may be.

   Other authors (Bedau, 1997; Searle, 1992) consider that the difficulty of the computation does make a difference: when the computation of the macro-state is particularly difficult (Bedau, 2003), the system is said to be *weakly emergent*. Zwirn & Delahaye (2013) formally define *computational irreducibility*[2] to specify what is meant by 'particularly difficult'. Other authors are very unclear on this topic. Informally (and using our definitions 15 & 16), computational irreducibility means that there is no way to compute $\Omega(S)$ from $\omega_m(S)$ without having to compute every state of every element in turn; in other words, f_m cannot be simplified: every detail of the system matters.

2. f_m does not exist – Assad & Packard (1992) rate this circumstance at the top of a scale of four increasing degrees of emergence and call it *maximal emergence*: the macro-state cannot be deduced from the micro-state *by nature*.

Our definitions of micro-state, macro-state and upscaling function, while mapping relatively well to existing concepts of emergence, nevertheless raise new questions.

Authors defining weak emergence usually consider only system 'components' without explicitly including relations and incidence functions in their system definition (cf. Section 2.1). Therefore, it is difficult to know to what parts of the system computational irreducibility should apply. There appear to be two options:

1. Computational irreducibility interpreted *stricto sensu*, i.e., the computation of the macro-state must include all the details of the system. We interpret this as meaning that the only relevant context is the maximal context q_S. The condition for weak emergence is then:

[2]Zwirn's definition of computational irreducibility is based on Turing machines (Zwirn & Delahaye, 2013), but he states in this article that a similar definition could be proposed based on functions. We assume here that such a definition exists, functions being easier to include in the present formalism.

Definition 33. *Weak emergence* occurs when the only existing upscaling function is based on q_S and is computationally irreducible *sensu* Zwirn & Delahaye (2013):

$$\begin{cases} \exists f_S \neq 0 \,|\, \Omega(S) = f_S(\omega_S(S)), \text{ and } f_S \text{ is irreducible} \\ \nexists m \in \mathbb{N} \,|\, \Omega(S) = f_m(\omega_m(S)) \end{cases}$$

where 0 stands for the null function.

2. Computational irreducibility interpreted more loosely:

Definition 34. *Context-dependent emergence* occurs when the upscaling function f_m is computationally irreducible *sensu* Zwirn & Delahaye (2013):

$$\exists m \in \mathbb{N} \,|\, (\Omega(S) = f_m(\omega_m(S))) \text{ and } f_m \text{ is irreducible}$$

Here we make no assumption about other contexts as context-dependent emergence applies only within this specified context m. The system may well be non-emergent using other local contexts.

With the upscaling function, we can explicitly link weak emergence (Bedau, 1997; Searle, 1992) to computational irreducibility, as claimed by Zwirn & Delahaye (2013). Our explicit representation of the system as a graph enables us to distinguish between a weak emergence independent of local context and a context-dependent weak emergence, which depends on how micro-states are built with respect to local context. Definition 34 is useful when only one non-trivial local context is available to compute the micro-state, a probably common case compared to the requirements of Definition 33 where all local contexts must be assessed.

The case where an upscaling function does not exist is difficult to interpret: it is hard to imagine this possibility without it being a case of missing descriptors, as in *discovery* (Definition 31). If this is not the case, how should we interpret this kind of emergence, usually qualified as *strong*? Could a system, with a macro-state that cannot be explained from its micro-state, exist? To diagnose strong emergence in a system would mean that there is no way to explain its macroscopic behaviour (or some part or it) from its microscopic description. It would mean that any scientific enquiry to explain the macroscopic behaviour from the parts would be doomed to fail. Conversely, any attempt at explaining the macroscopic behaviour from the parts assumes that the system does not display strong emergence. Quoting Bedau (2003):

> Strong emergence starts where scientific explanation ends.

Short of having any evidence of being in presence of strong emergence, and as long as we are willing to keep trying to explain the macro-state from the miscro-state, we are left with the assumption that strong emergence does not exist. In our formalism, this translates to *assuming*:

Proposition 5. *A hierarchical system always has at least one upscaling function:*

$$\forall S, \exists (\Omega, m, \omega_m, f_m) \,|\, \Omega(S) = f_m(\omega_m(S))$$

If the upscaling function cannot be found using the current system description, then new descriptors must be added to the description until a satisfactory upscaling function can be written.

In accordance with our present scientific approach of the world, we therefore assume that maximal emergence *sensu* Assad & Packard (1992) does not exist. We expect system-level and subsystem-level descriptions to match somewhere – even if this 'somewhere' is very difficult to find. If they do not, it means something is missing in the description of either or both levels.

2.4 Emergence due to complex causality networks

Integration, discovery and computational irreducibility, considered as forms of emergence by many authors, do not require the system to be dynamic. However, many specialists of emergence base their definitions on

causation, implying a dynamic system (Bedau, 2003; Kim, 1992). For Bedau (2003) and Searle (1992), *downward causation* (*i.e.* the system causing its elements to change) that is *not reducible to upward causation* (*i.e.* changes in the system being caused by its elements) is called *strong emergence*. However, they believe strong emergence does not exist in nature. In contrast, for Muller (2003), the distinction between weak and strong emergence depends on downward causation from the system on its components by (1) merely affecting their behaviour (weak) or (2) changing the rules of their behaviour (strong). It seems that in all cases, including Kim (1999), emergence is associated with the difficulty in predicting the system's behaviour because of complicated causal pathways. Although there is clearly no agreement on how emergence relates to causal pathways, there seems to be a tendency to admit that (1) the microscopic control of dynamics is the 'normal' causal pathway, *i.e.* the system macro-state is supervenient to its micro-state given some context, and is (weakly) emergent when the upscaling function f_m is computationally irreducible (Definitions 33-34); and (2) any macroscopic control of the dynamics is reducible to the microscopic dynamics, *i.e.* there is a causal loop from the micro-states to the macro-state and back to the micro-states.

There are problems with these definitions: (1) they tend to focus on components to the exclusion of relations; relations could also have causal powers; (2) there may be a wide diversity of causal powers among components and relations, *i.e.* they may generate structural change events, or drive smooth changes in descriptor variables, or just change passively in response to stimuli from other components or relations; and (3) they ignore the potentially very high complexity of causal networks (Section 1.5), where causal pathways may cross and interact in very elaborate ways, generating particular causal circuits (Figure 7). These ideas may be present, but only *implictly* in most articles. Making the causal network explicit (Definition 26) is our answer to these complex causality issues: what could happen may be so intricate that the notions of downward or upward causation just become obsolete because they are incapable of capturing such complexity.

Research in cybernetics (Wiener, 1948) and biology (Weiss, 1971) has for some time recognized the ability of feedback loops to disrupt the linear chain of causality by having an effect acting back upon its cause. Patten & Odum (1981) identified two different ways of constructing feedback loops within a cybernetic system. For Bateson (1972, quoted in Ulanowicz, 2009), causal circuits can react non-randomly to random events. Electronic engineering routinely uses feedback loops of various types to design complex regulatory systems (control theory: Astolfi et al., 2008). The causal loop is considered the first step towards self-organisation (De Wolf & Holvoet, 2005). Cotsaftis (2009b) considers that a system is complex when 'self-organisation filters out external action, making the system more robust to outer effects'. In ecology, Ulanowicz (1990) builds upon this result to demonstrate that *feedback loops* in trophic networks disrupt the overwhelming physical linear causation to yield a different logic, where the stability of cycles arises from a constant throughput of energy or matter. His reasoning is based on the energy and matter flows in trophic webs. Ulanowicz (2009) further demonstrates that some loops may be stable and constitute kernels of coherence in a system. In all these works, loops are able to acquire a predictable and deterministic-like behaviour from randomness, and remain partly autonomous from external forcings. The weakness of all these works, however, is that most of them cultivate an ambiguity between flows and causes: there is a semantic drift from flow loops to causal loops in their vocabulary.

Here, we solve this ambiguity by explicitly building, for any dynamic hierarchical system, a causal network (Definition 26). The causal network itself is a particular case of a flow network (Diestel, 2000), so that all the previously mentioned cybernetic and flow dynamics results apply. Our formalism gives them a much broader scope. Not only matter and energy flows, but any other type of graph dynamics, including structural changes (component and relation addition and deletion: cf. Definition 19, 20), can be considered.

What is the relation between (stable) causal loops and emergence? According to Bedau (1997), emergence has two 'hallmarks':

> (1) Emergent phenomena are somehow constituted by, and generated from, underlying processes;
>
> (2) Emergent phenomena are somehow autonomous from underlying processes.

Autonomy is the key word. According to cybernetics, stable causal loops are able to provide some autonomy to a system. Here, autonomy means the ability for the system to generate outputs independently from external

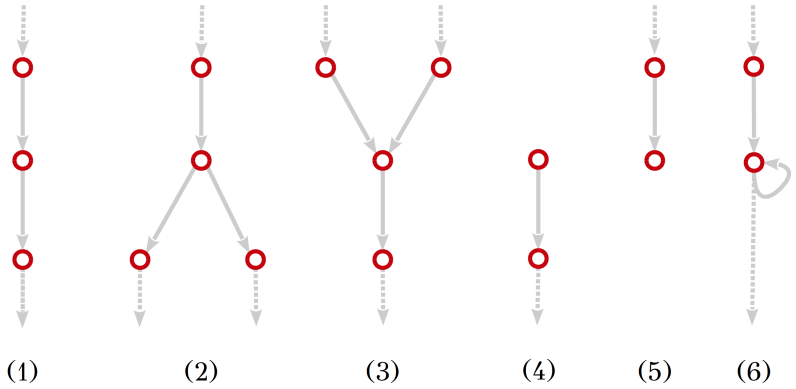


Figure 7: Elementary causal structures that can appear on a causal network (Definition 26). (1) chain, (2) divergent fork, (3) convergent fork, (4) source, (5) sink, (6) elementary (unitary) loop.

sources of change (*e.g.* producing a constant outflow from a randomly varying inflow, generating discrete events from a continuous flow input). When those outputs have causal consequences over the external world, they are also independent from external influence, and thus can be considered as *new* causal powers of the system. The existence and nature of a causal loop (referred to as *downard* and *upward* causation) between the system and its components, is at the core of definitions of emergence by Bedau (2003) and Searle (1992).

According to Ulanowicz (2009), causal loops may be either *cohesive* or *disruptive*. Cohesive causal loops (effects reinforcing their causes through iteration) result in system autonomy, and tend to maintain themselves with time, hence be stable, while disruptive causal loops (effects destroying their causes through iteration) are instable and result in system collapse. Being due to new causal powers, stability, autonomy, and collapse, constitute observer-independent emergent system properties. Extending the demonstration of Ulanowicz (2009) for flow networks to causal networks, we propose that they are due to the particular structure of the system's causal network:

Proposition 6. Ontological emergence.

1. *A hierarchical dynamic system S has* ontologically emergent *properties when its causal network $Q(S)$ is not a walk, or equivalently, contains loops.*

2. *The presence of particular causal structures (types 2-6 in Figure 7) in the causal network $Q(S)$ of a hierarchical dynamic system S disrupts linear causality.*

Point (2) of Proposition 6 is a conjecture generalising the demonstration by Ulanowicz (2009) that feedback loops in flow networks are responsible for emergent system properties. It is easy to demonstrate that these results hold for causal networks as they are a particular case of flow networks. It remains to be demonstrated that divergent and convergent causal forks, causal sources and sinks (Figure 7), may also produce emergent properties. Demonstrating this requires work beyond the scope of this paper.

Many people associate emergence with *complex* systems. Here we propose a definition of a *complex* system based soley on *ontological* emergence:

Definition 35. A *complex system* is a *hierarchical dynamic system* displaying *ontological emergence*.

Although this definition is not really useful for modelling, it enables clarification of the the vocabulary by specifying which type of emergence should be considered when speaking of complex systems. Using UML, we placed the definitions of emergence of Section 2 in the appendix, on the diagrams of Figures 1-5, yielding Appendix Figure 6.

Conclusion

The set of definitions we propose here, organised into a consistent framework called *the hierarchical system ontology* (see appendix), enable (1) the clarification and unification of many definitions of emergence and systems and (2) the building of rigorous models of hierarchical systems in a most generic way. All the definitions of emergence we have analysed to build our ontology are applicable to any hierarchical system and can be organised in order of increasing 'objectivity'.

- Despite very active recent research on the use of graphs in various sciences, ignoring relations in a system still seems to be common practice: people tend to focus on objects, especially when they belong to the real world and are concrete, rather than on the more abstract concept of relations (*e.g.*, even *relational* SQL databases are actually collections of objects (tables), where relations are not implemented explicitly, but rather as querying rules). As a result, some call emergence simply a discovery that relations are important. We called this type of emergence *trivial emergence* (Section 2.1).

- Beyond this, the observer often discovers that system behaviour and properties cannot be fully described by their initial system model. The discrepancy between the real-world system and the model is interpreted as a case of emergence from the system: a new, unpredicted behaviour. It may be that new descriptors are sufficient to eliminate this discrepancy. This process constitutes the very core of the experimental method: science is an iterative process where discoveries are made from successive experiments on the same system and allow, layer by layer, a more accurate image of the object under study to be elaborated. We called this type of emergence *discovery* (Section 2.2).

- At times, systems resist simplification; predicting system behaviour requires very difficult computations in order to link microscopic and macroscopic properties. This obstacle to understanding, called *computational irreducibility*, is considered by many as a signature of emergence. Following other authors, we called *weak emergence* the existence of a computationally irreducible link between macro- and micro-states. We also distinguished the particular case where a different look at the system may solve the irreducibility problem, called *context-dependent emergence* (Section 2.3).

- Most definitions of emergence based on causality do not delve into the details of causal propagation over a graph. Some disciplines, however, have identified that causal loops (or feedbacks) could behave as *novel* causal sources within a hierarchical system. Identifying causal loops as the source of autonomy, stability, or collapse, has two advantages in the definition of emergence: (1) it is based on a precisely defined causal mechanism; and (2) it does not rely only on observer surprise, but on a real process creating novelty that leads to system self-organisation, coherence, durability, or on the contrary, excessive fragility. We called this *ontological emergence* (Section 2.4), as it is the only type of emergence independent of the observer's point of view. We further propose that the qualifier *complex* should be reserved for dynamic hierarchical systems exhibiting this kind of emergence.

- Finally, we avoid the term *strong emergence*. As defined by various authors, strong emergence is the non-existence of a link between the micro-state and the macro-state, *i.e.* the impossibility of explaining the macro-state from the micro-states. To pursue a scientific explanation is to assume that the micro- to macro-state link always exists. Thus, strong emergence cannot exist in scientific discourse (Section 2.3).

Emergence and complex systems have been the topic of many papers and are still disputed concepts in many fields. While a number of definitions converge, some confusion and ambiguity still remains. Based on graph theory and on the common premises of all emergence definitions, we have proposed a consistent set of formal definitions that we hope will clarify these concepts in modelling and aid further discourse. As noted in the case of trivial emergence, what really makes the difference between a system and a complex system is a graph branching pattern - but not the expected one of the system itself. Rather, it is the architecture of the causal network associated with that system that will invoke true, ontological emergence.

Epilogue

An ecologist, having reached this point, may be left wondering what practical relevance this subject has for the everyday practice of ecology. But can ecologists be content simply to describe their basic unit of study – the ecosystem – as, in some mysterious way, complex, and leave it at that? If one message were to be taken from this paper, it is that *feedback loops are central for emergent properties.* The ecological literature is replete with discussions of ecosystem stability, resilience, resistance, bi-stability etc for a good reason: they are central concerns if ecology is to stake a claim as a predictive science. As all these behaviours are commonly assumed to be emergent properties, how can we sensibly conduct such discussions without first agreeing on a clear definition of emergence?

References

Ahl, V. & Allen, T. F. H. (1996). *Hierarchy theory, a vision, vocabulary and epistemology.* Columbia University Press.

Allen, T. F. H. & Hoekstra, T. W. (1992). *Toward a unified ecology.* Columbia University Press.

Assad, A. & Packard, N. H. (1992). Emergent Colonization in an Artificial Ecology. In F. Varela & P. Bourgine (Eds.), *Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life* (pp. 143–152).

Astolfi, A., Karagiannis, D., & Ortega, R. (2008). *Nonlinear and Adaptive Control with Applications.* Communications and Control Engineering, Springer.

Bar-Yam, Y. (2004). Multiscale complexity/entropy. *Advances in Complex Systems*, 7(01), 47–63.

Bedau, M. A. (1997). Weak Emergence. In *Philosophical Perspectives: Mind, Causation, and World*, number 11 (pp. 375–399). Malden, MA: Wiley-Blackwell.

Bedau, M. A. (2003). Downward Causation and the Autonomy of Weak Emergence. *Principia Revista Internacional de Epistemologica*, 6, 5–50.

Bedau, M. A. (2008). Is Weak Emergence Just in the Mind? *Minds and Machines*, 18(4), 443–459.

Bedau, M. A. & Humphreys, P., Eds. (2008). *Emergence: Contemporary Readings in Philosophy and Science.* Cambridge: MIT Press.

Bernouilli, J. (1713). *Ars conjectandi.* Bâle: Thurneysen brothers.

Boltzmann, L. (1871). Einige allgemenine Sätze über das Wärmegleichgewicht. *Wiener Berichte*, 63, 679–711.

Bonabeau, E. & Dessalles, J.-L. (1997). Detection and emergence. *Intellectica*, 25(2), 85–94.

Carnot, S. (1824). *Réflexions sur la puissance motrice du feu et sur les machines propres à développer cette puissance.* Paris: Bachelier.

Chen, C.-C., Nagl, S. B., & Clack, C. D. (2009). : (pp. 101–114). Springer Berlin / Heidelberg.

Cochran, W. G. (1977). *Sampling techniques.* New York: Wiley.

Cotsaftis, M. (2009a). An Emergence Principle for Complex Systems. In J. Zhou, O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, M. Gerla, H. Kobayashi, S. Palazzo, S. Sahni, X. S. Shen, M. Stan, J. Xiaohua, A. Zomaya, & G. Coulson (Eds.), *Complex Sciences*, volume 4 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering* (pp. 1105–1117). Springer Berlin Heidelberg.

Cotsaftis, M. (2009b). What Makes a System Complex? - An Approach to Self Organization and Emergence. In M. Aziz-Alaoui & C. Bertelle (Eds.), *From System Complexity to Emergent Properties*, volume 44 of *Understanding Complex Systems* (pp. 49–99). Berlin / Heidelberg: Springer.

Crutchfield, J. P. (1994). Is Anything Ever New? Considering emergence. In G. A. Cowan, D. Pines, & D. Meltzer (Eds.), *Complexity: Metaphors, Models and Reality*, number 19 in Santa Fe Institute Studies in the Sciences of Complexity (pp. 479–497). Reading, MA: Addison-Wesley.

de Haan, J. (2006). How emergence arises. *Ecological Complexity*, 3(4), 293–301.

De Wolf, T. & Holvoet, T. (2005). Emergence versus self-organisation: Different concepts but promising when combined. In *Engineering self-organising systems* (pp. 1–15). Springer.

Dessalles, J.-L., Muller, J.-P., & Phan, D. (2007). : (pp. 327–355). Oxford : The Bardwell Press.

Diestel, G. T. (2000). *Graph Theory*. New York: Springer Verlag.

Duboz, R., Ramat, E., & Preux, P. (2003). Scale Transfer Modeling: Using Emergent Computation for Coupling an Ordinary Differential Equation System with a Reactive Agent Model. *Systems Analysis Modelling Simulation*, 43(6), 793–814.

Gignoux, J., Davies, I. D., Flint, S. R., & Zucker, J. D. (2011). The ecosystem in practice: Interest and problems of an old definition for constructing ecological models. *Ecosystems*, 14, 1039–1054.

Gross, J. L. & Yellen, J. (1999). *Graph Theory and Its Applications*.

Guarino, N. (1995). Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human Computer Studies*, 43(5/6), 625–640.

Harary, F. & Gupta, G. (1997). Dynamic graph models. *Mathematical and Computer Modelling*, 25(7), 79–87.

Jax, K. (2007). Can we define ecosystems? on the confusion between definition and description of ecological concepts. *Acta Biotheoretica*, 55, 341–355.

Jordan, N. (1981). Some thinking about 'System'. In F. E. Emery (Ed.), *System Thinking: Selected Readings*, volume 2 (pp. 15–39). Penguin, 2 edition.

Kim, J. (1992). 'Downward-causation' in emergentism and nonreductive physicalism. In A. Beckerman, H. Flohr, & J. Kim (Eds.), *Emergence or reduction? Essays on the prospects of nonreductive physicalism*. Berlin: Walter de Gruyter.

Kim, J. (1999). Making sense of emergence. *Philosophical Studies*, 95, 3–36.

Muller, J.-P. (2003). Emergence of Collective Behaviour and Problem Solving. In A. Omicini, P. Petta, & J. Pitt (Eds.), *Engineering Societies in the Agents World IV, 4th International Workshop, ESAW 2003, London, UK, October 29-31, 2003, Revised Selected and Invited Papers*, volume 3071 of *Lecture Notes in Computer Science* (pp. 1–21).: Springer.

Object Management Group (2015). *OMG Unified Modeling Language (OMG UML)*. Version 2.5 formal/2015-03-01.

O'Neill, R. V., DeAngelis, D., Waide, J., & Allen, T. F. H. (1986). *A hierarchical concept of ecosystems*. Princeton University Press.

Patten, B. C. & Odum, E. P. (1981). The Cybernetic Nature of Ecosystems. *The American Naturalist*, 118, 886–895.

Searle, J. (1992). In *The Rediscovery of the Mind* (pp. 111–126). Cambridge: The MIT Press.

Tansley, A. G. (1935). The use and abuse of vegetational concepts and terms. *Ecology*, 16, 284–307.

Ulanowicz, R. E. (1990). Aristotelean Causalities in Ecosystem Development. *Oikos*, 57(1), 42.

Ulanowicz, R. E. (2009). The dual nature of ecosystem dynamics. *Ecological Modelling*, 220(16), 1886–1892.

Weiss, P. A. (1971). *Hierarchically organized systems in theory and practice*. New York: Hafner.

Wiener, N. (1948). *Cybernetics or Control and Communication in the Animal and the Machine*, volume 25. MIT press.

Zeigler, B., Kim, T. G., & Praehofer, H. (2000). *Theory of Modeling and Simulation : integrating discrete event and continuous complex dynamic systems*. New York: Academic Press.

Zwirn, H. & Delahaye, J.-P. (2013). Unpredictability and computational irreducibility. In *Irreducibility and Computational Equivalence* (pp. 273–295). Springer.

Emergence and complex systems: the contribution of dynamic graph theory Appendix

Jacques Gignoux[1*], Guillaume Chérel[2], Ian D. Davies[3], Shayne R. Flint[4], Eric Lateltin[1]

January 26, 2017

1. Institute of Ecology and Environmental Sciences, CNRS UMR 7618, 4 place Jussieu, aile 44-45 CC 237, 75005 Paris, France
2. Institut des Systèmes Complexes Paris Île-de-France, 113 rue Nationale, 75013 Paris, France
3. Fenner School of Environment and Society, The Australian National University, Canberra ACT 2601, Australia
4. Research School of Computer Science, The Australian National University, Canberra ACT 0200, Australia

* corresponding author: jacques.gignoux@upmc.fr

The hierarchical system ontology

The following figures represent the relations between the concepts defined in the main document. They are based on a UML[1] class diagram notation.

Thee is a class for every item defined in definitions and propositions of the main document. For easier reading, the whole diagram has been splitted into 6 figures, matching sections 1.1 to 1.5, and 2. On every figure,

- white boxes represent definitions from the current section while gray boxes represent definitions from outside this section;
- classes with a name in italics represent concepts defined outside this article;
- number in circles match the definition number as appearing in the main document; numbers prefixed by a 'P' refer to proposition numbers;
- the conventional UML triangular arrows represent *specialisation* relations (*e.g.* in Figure 1, a relation *is* an element);
- the conventional UML diamond-ending lines represent *aggregation* relations (*e.g.* in Figure 1, a hierarchical system *has* n_e elements);
- bold lines represent other UML *associations*;
- numbers or number ranges (*e.g.* $0..*$, meaning '0 to $n > 0$') on associations and aggregations represent *multiplicities* (default to 1);
- expressions on associations are *roles*, the little arrow showing in which direction the role should be read (*e.g.* in Figure 1, the incidence function *assigns* 2 components *to* 1 relation).

[1] Object Management Group (2015). OMG Unified Modeling Language (OMG UML). Version 2.5 formal/2015- 03-01.

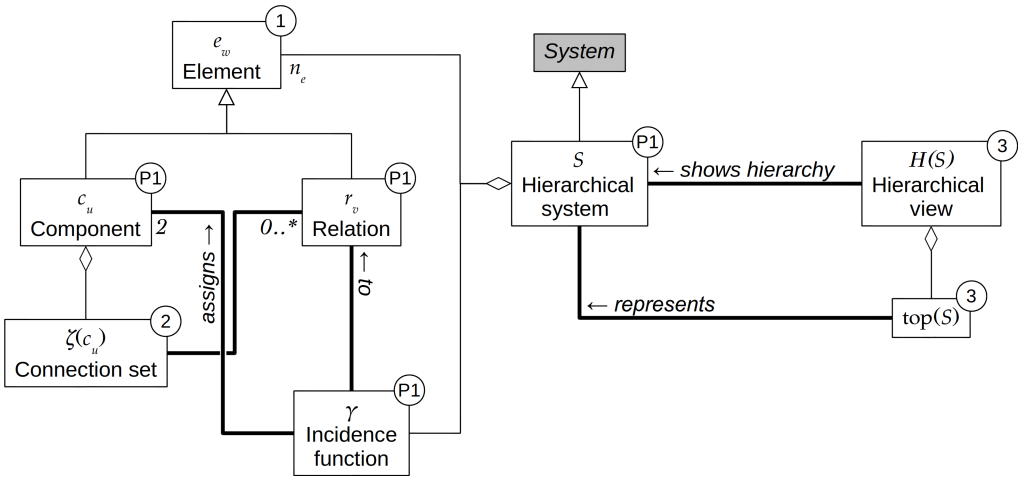


Figure 1: The hierarchical system ontology: concepts defined in Section 1.1.

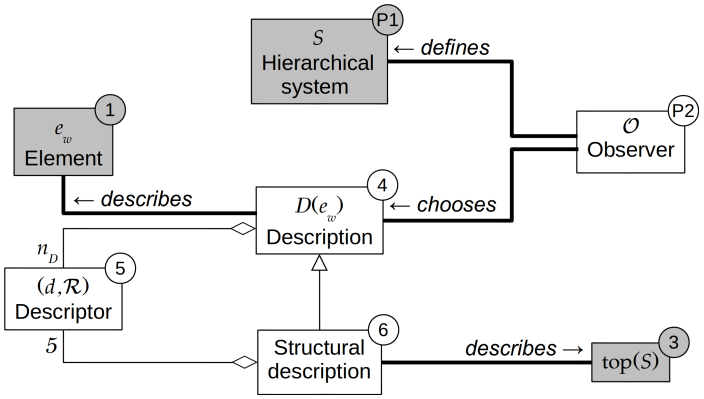


Figure 2: The hierarchical system ontology: concepts defined in Section 1.2.

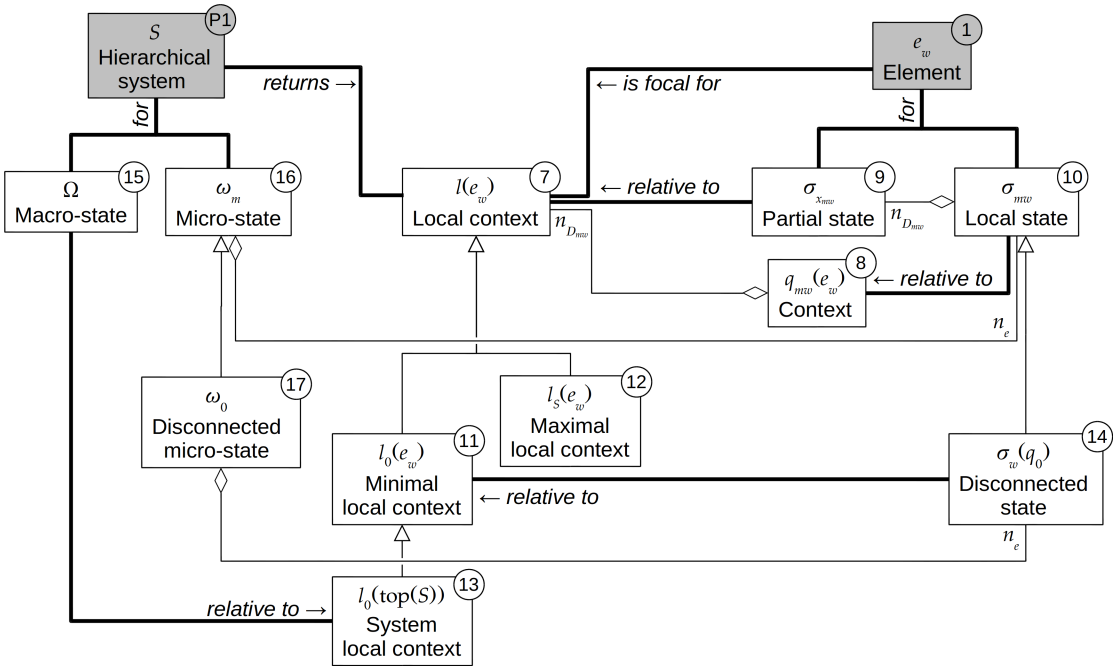


Figure 3: The hierarchical system ontology: concepts defined in Section 1.3.

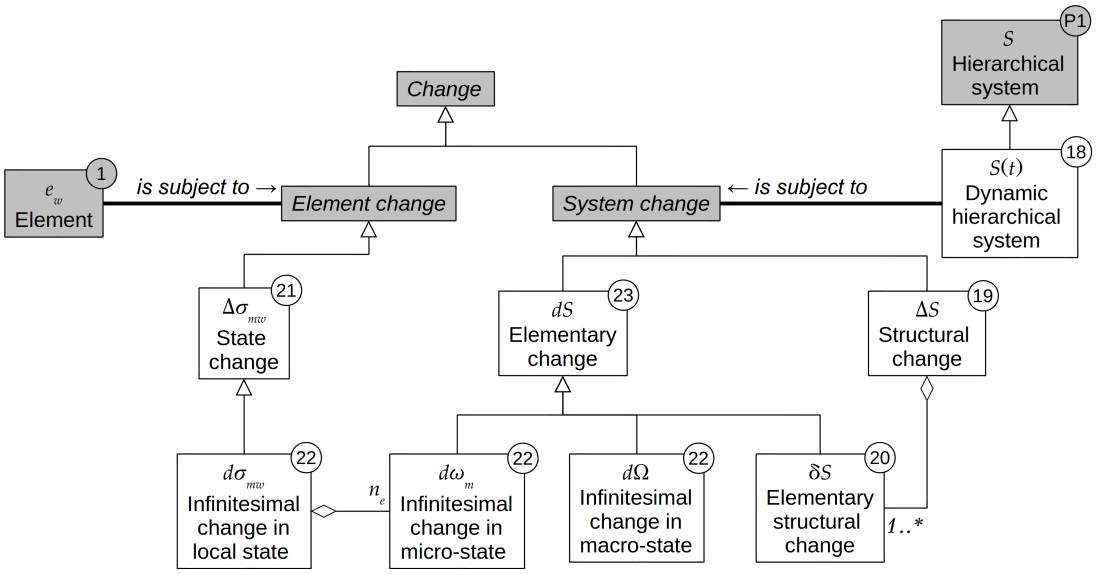


Figure 4: The hierarchical system ontology: concepts defined in Section 1.4.

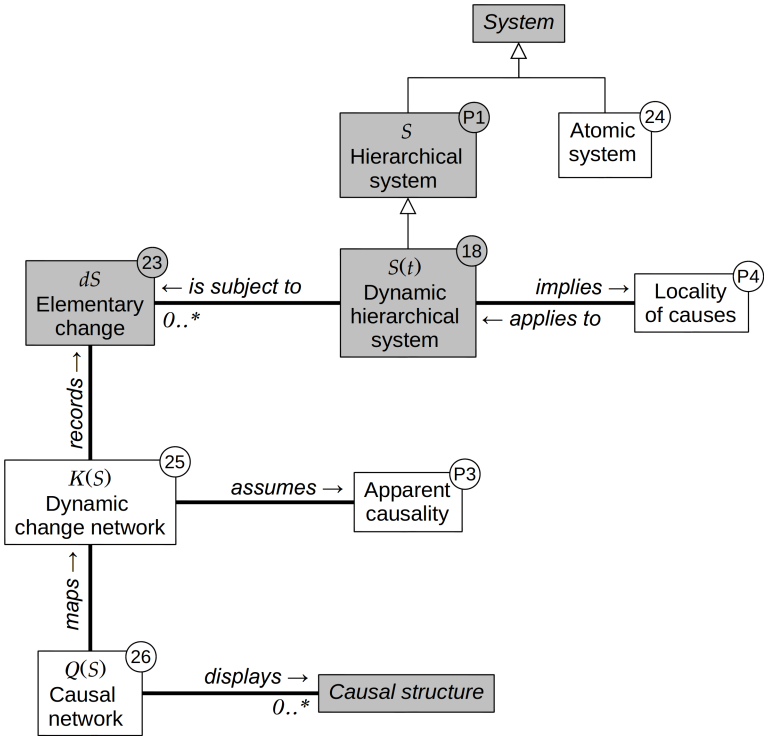


Figure 5: The hierarchical system ontology: concepts defined in Section 1.5.

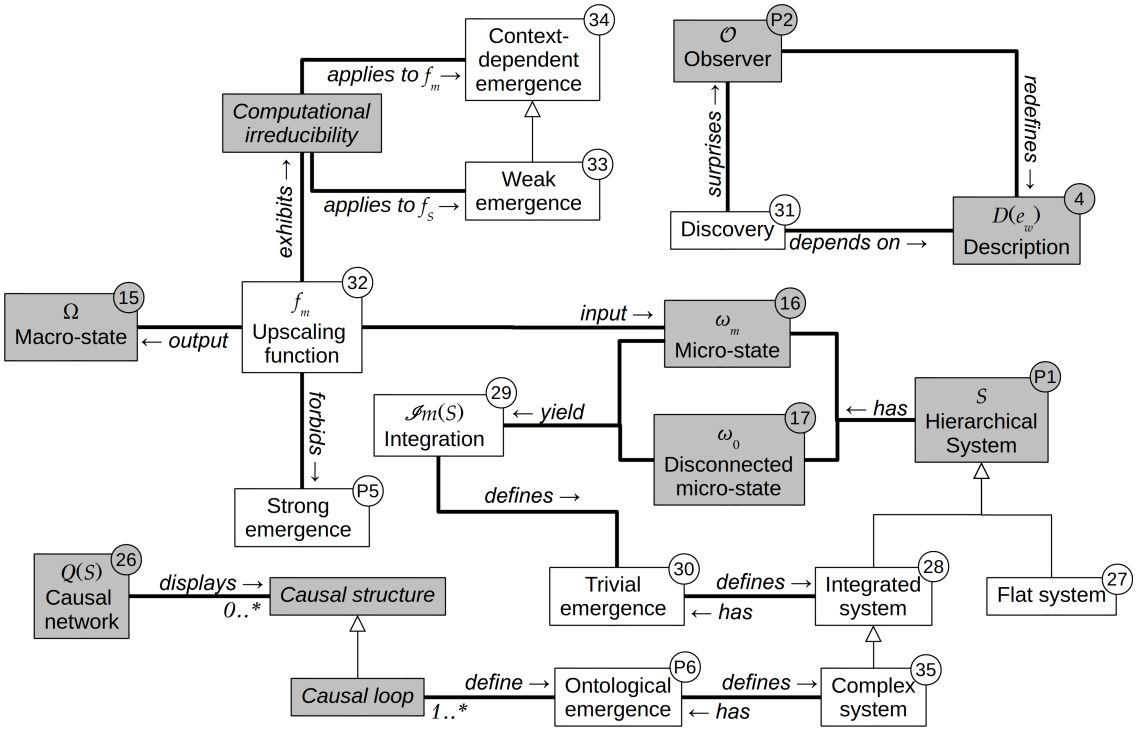


Figure 6: The relation of emergence concepts defined in Section 2 with the hierarchical system ontology.