What is a structural representation?

A proposal for an event-based representational formalism

Sixth Variation *

Lev Goldfarb, David Gay, Oleg Golubitsky, Dmitry Korkin, Ian Scrimger

Faculty of Computer Science
University of New Brunswick
Fredericton, Canada

July 17, 2008

> [W]e may again recall what Einstein stressed: that given a sufficiently powerful formal assumption, a fertile and comprehensive theory may ... be constructed without prior attention to the detailed facts, or even before they are known.
>
> L. L. Whyte, Internal Factors in Evolution, 1965

Abstract

We outline a formalism for structural, or symbolic, representation, the necessity of which has been acutely felt not just in artificial intelligence and pattern recognition, but also in the natural sciences, particularly biology. At the same time, biology has been gradually edging to the forefront of sciences, although the reasons obviously have nothing to do with its state of formalization or maturity. Rather, the reasons have to do with the growing realization that the objects of biology are not only more important (to society) and interesting (to science), but that they also more explicitly exhibit the evolving nature of *all* objects in the Universe. It is this view of *objects as evolving structural entities/processes* that we aim to formally address here, in contrast to the ubiquitous mathematical view of *objects as points in some abstract space*. In light of the above, the paper is addressed to a very broad group of scientists.

One can gain an initial intuitive understanding of the proposed representation by generalizing the temporal process of the (Peano) construction of natural numbers: replace the *single* structureless unit out of which a number is built by *multiple* structural ones. An immediate and important consequence of the distinguishability (or multiplicity) of units in the construction process is that we can now see *which* unit was attached

* For earlier versions of this paper, see [1]–[5]. In Part I of the paper, we use single quotes when we refer to the terminology introduced later in this paper.

and *when*. Hence, the resulting *(object) representation* for the first time embodies *temporal structural information* in the form of a formative, or generative, object "history" recorded as a series of (structured) events. Each such event stands for a "standard" interaction of several objects/processes.

We introduce the new concept of *class representation* via the concept of *class generating system*, which outputs structural entities belonging to that class. Hence, the concept of *class* is introduced as that of a class of similar structural entities, where the "similarity" of such entities is ensured by them being outputs of the same class generating system and hence *having similar formative histories*. In particular, such a concept of class representation implies that—in contrast to all existing formalisms—no two classes have elements in common. The evolving transformation system (ETS) formalism proposed here is the first one developed to support such a new vision of classes. Most important, since the operations that participated in the object's construction are, for the first time, made explicit in the representation, it makes the inductive recovery of class *representation* (on the basis of object *representation*) much more reliable.

As a result, ETS offers a formalism that outlines, for the first time, a tentative framework for understanding what a class is. Even this *tentative* framework makes it quite clear that the term "class" has been improperly understood, used, and applied: many, if not most, of the current "classes" should not be viewed as such. A detailed example of a class representation is included.

In light of ETS, the classical discrete "representations" (strings, graphs) appear as *incomplete* special cases at best, the proper adaptation of which should incorporate corresponding formative histories, as is done here.

The gradual emergence of ETS—including the concepts of structural object and class representations, the resulting radically different (temporal) view of "data", as well as the associated inductive learning processes and the representational levels—points to the beginning of a new field, *inductive informatics*, which is intended as a *class oriented* rival to conventional information processing paradigms.

Part I
Prolegomenon

> Those who still wish to build a computational empire on the basis of such troubling precedents [i.e. to assume that the Church-Turing thesis is an adequate scientific and/or epistemological basis for the science of information processing] would do well to pause first on the significance of Wittgenstein's ever-timely warning that "One keeps forgetting to go right down to the foundations. One doesn't put the question marks *deep* enough down."
>
> S. Shankar, Wittgenstein's Remarks on the Foundations of AI, 1998

1 Introduction

1.1 Obstacles toward a formalism for structural representation

In this paper we outline a vision of the concept of (temporal) structural representation which has been in gestation for twenty years. Since such a grand vision cannot be corroborated by one research group or in a short period of time, it is only natural to present it to the wider research community.

Despite the fact that the overwhelming importance of structural, or symbolic, representations in many sciences has become increasingly clear during the second half of the twentieth century, there have been no *systematic* attempts to address this topic at a fundamental level[1]. It is not that difficult to understand the main reasons behind this state of affairs. From a theoretical point of view, it appears there are two very formidable obstacles to be cleared: 1) the choice of the central "intelligent" process (among many possible candidates, e.g. induction, deduction, etc.), the structure and requirements of which would drive and justify the choice of a particular form of structural representation, and 2) the lack of any *fundamental* mathematical models whose roots are not directly related to numeric models. The order in which these obstacles must be addressed is important: obviously, one must first choose which intelligent process to model before attempting to look for a satisfactory formalism. Unfortunately, the second (and principal) of the above obstacles is usually underestimated or overlooked entirely.

Why has it been overlooked? Because, during mankind's *scientific* history, we have dealt only with numeric models and, during the last century, with their derivatives. The latter should not be surprising if we look carefully at the vast prehistory of science in general, and of mathematics in particular [6], [7]. New mathematical abstractions and overspecializations (with a resulting narrowing of historical perspective) during the second half of the twentieth century have also contributed to such a lack of understanding of the extent to which we depend on numeric models[2]. What has (barely) begun to facilitate this understanding,

[1] This situation is particularly puzzling from the point of view of computer science, in view of the central role played by data structures and abstract data types.

[2] There are, of course, rare exceptions (see [8], for example).

however, is the emergence of computer science in general, and artificial intelligence and pattern recognition (PR) in particular[3].

The relevant *concept* of a representational formalism is addressed briefly in Section 1.4 and in [9]. Here we simply mention that according to the view expressed therein, we presently have (excluding ETS) only one, albeit *very* primitive, representational formalism, i.e. the ubiquitous numeric formalism. In this sense, it is not surprising that the numeric formalism is, basically, the only scientific currency. With this paper, we aim to change this situation—a goal, on the one hand, absolutely unprecedented in the history of science, but, on the other hand, following from the development of PR and AI.

The complete monopoly of numeric models in science[4] suggests that it is unreasonable to expect a transition from numerically-motivated forms of representation, which have a millennia-old tradition behind them, to structural forms of representation to be accomplished in one or several papers. At the same time, one should not try to justify, as is often done in artificial intelligence, practically nonexistent progress in this direction by the long standing difficulties involved.

For an extended discussion of related issues, see [9].

1.2 Historical perspective on pattern recognition: the need for unification

In this work, we outline a fundamentally new formalism—evolving transformation system (ETS)—which is the culmination of a research program originally directed towards the development of a unified framework for pattern recognition [11]–[18].

In view of the fact that newer, more fashionable "reincarnations" of PR (see footnote 3) have missed what is probably the most important representational development within PR during the 1960s and 1970s, we now touch on this issue (which actually motivated the original development of the ETS framework). Over these two decades, it gradually became clear to a number of leading researchers in PR that the two basic approaches to PR—the classical vector-space-based, or statistical, approach and the syntactic, or *structural*, approach [19][5], each possessing the desirable features lacking in the other—should be unified [20]:

> Thus the controversy between geometric and structural approaches for problem of pattern recognition seems to me historically inevitable, but temporary. There are problems to which the geometric approach is ... suited. Also there are some well known problems which, though solvable by the geometric method, are more easily solvable by the structural approach. But any difficult problems require a combination of these approaches, and methods are gradually crystallizing to combining them; the

[3] Although several "new" areas *very* closely related to PR—such as machine learning (ML), neural networks (NN), etc.—appeared during the last twenty years, we will often refer to them collectively by the name of the original area, i.e. pattern recognition, or occasionally as inductive learning.

[4] For an insightful explanation of how the stage was set for this, see [10].

[5] The author of [19], King-Sun Fu, was not only instrumental to founding the International Association for Pattern Recognition (IAPR) and served as its first president, but was also the main driving force behind the emergence of structural PR as one of the main fields in PR. His untimely death in 1985 took the wind out of structural PR's sails.

structural approach is the means of construction of a convenient space; the geometric is the partitioning in it.

Although these original expectations for an impending unification were quite high, it turned out that such hopes were quite naive, not so much with respect to timeliness but with respect to the (underestimated) novelty of such a unified formalism: there was *no* formal framework which could naturally accommodate such a unification [11]. It is interesting to note that researchers working in the various above "reincarnations" of PR have only relatively recently become aware of the need for, and of the difficulties associated with, such an effort. The large number of conferences, workshops, and sessions devoted to so-called hybrid approaches (e.g. [21]–[28]) attests to the rediscovery of the need for unification.

In fact, as we advocate in this paper, a fundamentally new scientific language is necessary to adequately address the concept of (structural) object and class representations.

1.3 The general direction we have taken

Returning to the two formidable obstacles mentioned in Section 1.1, for us and many others the choice of the central intelligent/information process reduced to the pattern recognition process, or more accurately the pattern (or inductive) learning process[6], with an emphasis on the (inductive) *class representation*. On the other hand, overcoming the second obstacle, i.e. developing an appropriate mathematical formalism for modeling inductive processes, has been and will be a major undertaking.

What are some of the main issues *we* have encountered? In a roughly historical order, they are as follows. How does one approach the unification of the above two basic approaches to PR? [11] How should we approach the concept of inductive class representation (i.e. how should the Chomsky concept of generativity be rethought)? [13],[15],[16],[18] Initiating the formalization, how do we generalize the Peano axiomatic construction of natural numbers to the construction of structural entities (in other words, how do we formally capture the more general inductive, or generative, process of object construction)? How do we approach the concept of object representation as that of its formative process? What is the connection between a class description, or representation, and the process that generates class objects? How is an object representation connected to its class representation, and, moreover, how do these object representations change during the learning process? (See [1],[2],[3],[4] for the last several issues.) How do we introduce representational stages? [2],[3],[4] How do we deal with the periodic stages and non-periodic/transitory stages of processes? [3],[4] How do we treat object representations as processes [3] and how do we allow the processes to interact with each other? [4] How do we introduce multi-leveled class generating systems? It is understood that all of the above must be accomplished *naturally* and within a single general formalism.

[6] Inductive learning processes have been suggested as being the central intelligent processes by a number of philosophers and psychologists over the last several centuries (see, for example, [29], [30], [31]). An example of a more recent testament is: "This study gives an account of thinking and judgment in which ... everything is reduced to pattern recognition. ... That pattern recognition is central to thinking is a familiar idea" [32].

On the formal side, we chose the path of a far-reaching generalization of the Peano axiomatic construction of natural numbers ([33] or [34]), the axiomatics that forms the very foundation of the present construction of mathematics. This choice appears to be a very natural way to proceed. As well-known nineteenth-century German mathematician L. Kronecker aptly remarked, "God made the integers; all the rest is the work of man". Thus, in part, the *original* logic behind the *formalization* was this: take the only existing "representational" model, natural numbers, and generalize the process of their construction, i.e. replace the *single*, essentially structureless primitive out of which natural numbers are built (Fig. 12 (a), p. 30) by *various* structural ones (Fig. 9, p. 26). Then, one can build on that foundation (see also [9]).

1.4 On the concept of representation formalism

One should note that although the following concept of a representational formalism appears to be quite natural, this is, as far as we know, its first articulation (see also [9]).

First of all, on the formal side, we propose that *within* a representational formalism, the concept of a class should be generative and defined via the *basic (postulated in the formalism) operations*. For example, in a vector space, a class must be defined by linear operations only, i.e. one must be able to generate all class objects and only them (generativity) via linear operations. Within the Bourbaki architecture of mathematics, this should be considered as a standard mathematical requirement (see, for example, [35], [36]).

Second, having postulated the primacy of classes in nature, it is quite natural to demand from any representational formalism that the representational mapping f_{object} from the physical environment (PE) to the resulting set of object representations (OR),

$$f_{object}\colon PE \to OR\,,$$

(where $f_{object}(po)\,,\ po \in PE\,,$ is the representation of physical object po) *induces the corresponding class mapping*

$$f_{class}\colon PC \to RC\,,$$

of physical classes (PC) into the representational classes (RC), as depicted in Fig. 1.

The uniqueness of the ETS formalism (referred to at the end of Section 1.1) is related to its development being oriented towards the above two requirements, in addition to inductive considerations.

It is also important to emphasize the role of the above mapping $f_{class}\,,$ which, in fact, insists on treating classes not just as a figment of human imagination, but rather as a basic feature of reality, as the foundations of biology strongly suggest. There are several reasons why classes have not been perceived as real entities, the main two being the very abstract nature of classes (and their representations) and the total lack of representational formalisms that would support a satisfactory concept of class.

As to the mapping $f_{object}\,,$ it should be realizable (in principle) via some generalized sensory mechanism.

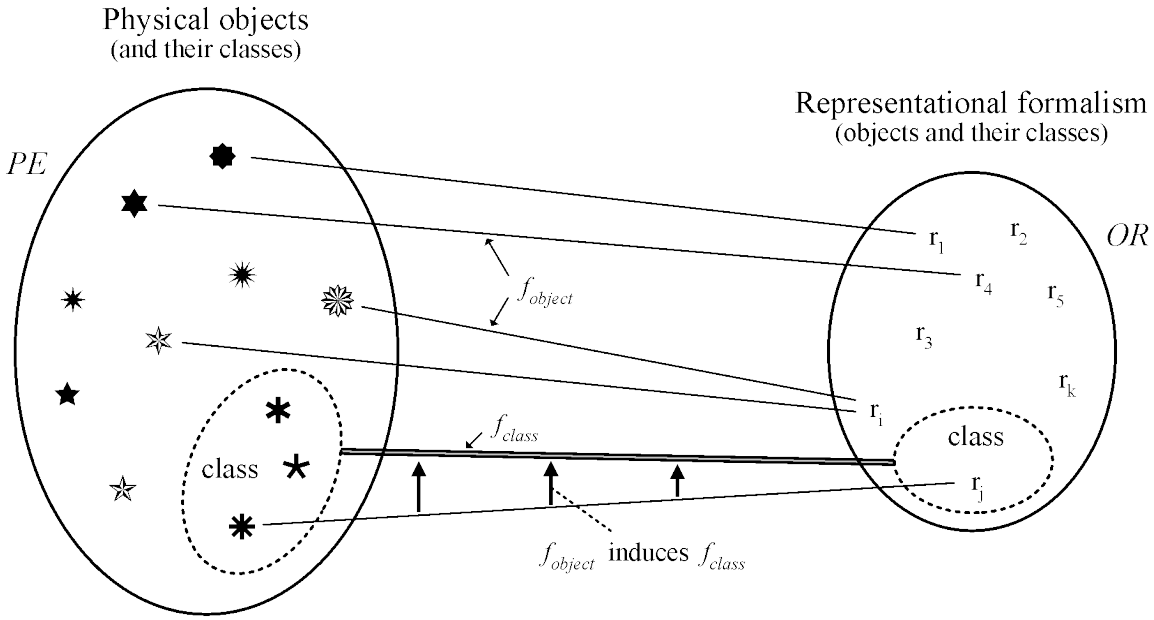


Figure 1: The main idea of a representational formalism.

1.5 ETS as the first proposal for a representational formalism

From an ETS perspective, there exist two related concepts or representations of object: the global (or Universe's) and an agent's. The global concept of object is that encapsulating the entire *formative history* of that object (as an information process in the Universe), while the agent's concept is that associated with the *agent's representation* of the object's formative history (relying, of necessity, on the agent's representational resources). Although we expect both concepts to be captured with the same formal means in the ETS formalism as it is presented here, we will usually address the agent's view, unless stated otherwise; in general, the context should clarify which sense is intended.

The foundations of ETS strongly suggests that the concept of *structural* object representation cannot be divorced from that of *generative* object representation, i.e. that of a representation capturing the formative/generative history of the object. Herein, we believe, lies the fundamental difference between classical numeric and structural, or symbolic, representations. In light of this, widely-used nonnumeric "structural representations" such as strings, trees, and graphs cannot be considered as such. *In short*, it becomes clear that, since such "representations" do not encode the formative object history[7], there is insufficient connection between the object representation and the corresponding class representation. Thus, for example, the framework of formal grammars proposed by Chomsky in the 1950s for generating syntactically-correct sentences in a natural language does not address these concerns, which is not quite surprising in view of his repeatedly-articulated opinion about the essential irrelevance of the inductive learning process to cognitive science (see for example [37], [38]).

[7] As was mentioned in the last paragraph, the latter should be understood not necessarily in the sense of the *actual* formative history, but rather in the light of the agent's "evolutionary" experience, i.e. from the point of view of the object's *recovered* (as a class element) formative history.

In particular, the generative issues so important to Chomsky simply cannot be properly addressed within the string setting, mainly because a string does not capture the object's formative history; as a result, there are exponentially many formative histories[8] hidden behind a string representation. From the vantage point of the ETS formalism, it becomes clear that the main reason why the various generative grammar frameworks have not succeeded as representational models has to do with their neglect of *more fundamental*, representational issues, i.e. the basic inadequacy of the string as a form of structural representation (see also [9]).

With respect to (generative) representation, it is useful to note that a number of philosophers and scientists have pointed out the importance of an object's past for that object's representation. Here is a recent example of one such expression [39]:

> [W]e shall argue that memory is always some *physical* object, in the present—a physical object that some observer *interprets as holding information about the past*.
>
> .....
>
> ... The past, about which the object is holding information, is the past *of the object itself*. In fact, an object becomes memory for an observer when the observer examines certain features of the object and *explains* how those features were *caused*.
>
> We shall argue ... that all cognitive activity proceeds via the recovery of the past from objects in the present. Cognitive activity *of any type* is, on close examination, the determination of the past.

As mentioned in the abstract, we expect that the scientific environment for understanding and investigating the nature of structural representation, together with the concepts of structural object and class representations and its various application areas—e.g. pattern recognition, data mining, information retrieval, bioinformatics, molecular phylogenetics, cheminformatics—will delineate a new information processing paradigm: inductive informatics. To understand the power of this paradigm, it is sufficient to mention how it would transform the widely applicable standard setting of information retrieval, relied on, for example, by all current search engines, in which a (very poor) class representation directly relies on a list of keywords, or on the latter plus logical connectives. Inductive informatics would reduce all such search problems to those of retrieving all class elements based either on a small class sample (possibly a single element) or *directly on the more powerful structural class representation*. Such a uniform formulation becomes possible only with the emergence of the more powerful concept of class representation (see Part III)

In light of the enormous difficulties related to the development of a formalism for structural representation, the best we can hope for as a result of the present attempt is to propose and outline a possible skeleton of such a formalism. We intend to use the proposed outline as a *guide* that will be modified in the course of extensive experimental work in numerous application areas. At the same time, as is always true in science, in our immediate experimental and theoretical work, we will also be guided by a reasonable interpretation of the present tentative formalism (now in its sixth version) In general, it is important to understand that,

[8] They correspond to sequences of transformations responsible for the formation of this string as an element of a particular class of strings.

when facing such a radical shift in representational formalism, one has *no other choice* but to begin with a theoretical framework, and only then move to the "data". Einstein emphasized this point in physics, but in this case the point should be even more apparent, since the notion of *data without a framework for data representation* is absolutely meaningless: it is the framework that dictates how "data" is to be obtained and interpreted. In particular, it is obvious that *any* sensor can only be built after the corresponding representational formalism has been proposed: a sensor can output only the *specified* object representations.

Ultimately, what should make or break ETS as a representational formalism? Since it explicitly postulates fundamentally different forms of object and class representation, the utility of these forms can now be experimentally verified. It is interesting to observe that the latter is not possible for any of the current inductive learning models, since they neither insist on, nor even propose, any—formally or otherwise—*meaningful and verifiable form of inductive class representation*, but simply *adapt* existing formalisms to fit the learning problem (without the availability of adequate concepts of both object and class representations in such formalisms). Thus, one of the immediate values of the ETS formalism is that it is the first formalism developed *specifically* to address the needs of the inductive learning process, and this paper should be interpreted as a *program for action* rather than simply a philosophical deliberation.

The framework's basic tenets both elucidate the nature of information (or "intelligent") processes in the Universe and are subject to experimental verification. In this respect, it is critical to keep in mind the accumulated scientific wisdom regarding the main value of a scientific model: "Apart from prediction and control the main purpose of science is ... explanation ..." [40] and "Whatever else science is used for, it is explanation that remains its central aim" [41]. *Current inductive learning models explain essentially nothing about the nature of these information processes*, since, as we firmly believe, hardly anything *can* be explained outside an adequate representational formalism.

Finally, ETS suggests a very different picture of reality than that implied by modern mathematics: equational descriptions of physical reality are replaced by structural descriptions (or representations) of evolving classes of objects. We believe that the proposed formalism provides radically different insight into the nature of objects and classes and offers a guiding metaphor badly needed by various sciences. As to progress in the development and applications of ETS, we believe that it would be accelerated within multidisciplinary groups (in which natural sciences are well represented), which, sadly enough, are presently lacking.

1.6 Organization of the paper

The paper is divided into five parts. Part I includes two introductory sections, the second of which proposes a new, informational view of the Universe. The substantial size of this Part is explained by our desire to make the main ideas accessible to a wider spectrum of scientists. Part II (Sections 3, 4) presents the basic formal concepts, and in Part III (Sections 5–7), the central part, we introduce the concepts of multi-level struct and class representation. A detailed example illustrating the main concepts covered up to that point is also presented. Another pair of central concepts—transformation and multi-stage inductive structure—are introduced in Part IV (Sections 9 and 10). Part V (Sections 11, 12) sketches some of our

preliminary thoughts on learning and suggests how one should approach the ETS formalism.

In view of the tentative nature of the present outline, it does not make sense to strive for a *very* rigorous form of exposition, though, as can be seen from our presentation, formal considerations have not been ignored. At the same time, due to the presence of many illustrative figures, the paper allows for an alternative, visual form of reading, which should benefit those who prefer to visually skim the material first. Also, as one would expect from an outline of a new representational formalism, we put our efforts into definitions, their illustrations, and their justification from the point of view of the ETS rationale. (As it turns out, even without theorems, the size of the paper is still substantial.)

For earlier expositions of the ETS formalism, see [1], [2], [4], [5], and [43]. Some preliminary applications of the earlier variations of ETS to cheminformatics are discussed in [44], to information retrieval in [45], [46], [47], to bioinformatics in [48], and to speech representation in [49]. Three theses on learning algorithms related to an earlier, pre-formal, stage of ETS are [50], [51], and [52].

We wish to thank Alexander Gutkin for his numerous helpful comments and Reuben Peter-Paul for his generous help with producing the figures in Part III and Appendix.

2 Proposed informational view of the Universe

What is the informational structure of the Universe that allows for the emergence of biological information processing? First of all, a biological information processing model must be fundamentally consistent with a prebiological information processing model: the former was built *on top* of the latter. We have assumed that the common, or unifying, central theme (for both kinds of models) has to do with the information centered around the concept of an evolving class of immediately related objects. Moreover, the "description" of the class is closely related to the formative structure of its elements.

Before discussing the existing state of affairs in relation to the concept of class, we present a simple (but, we believe, important) argument supporting the *above uniformity of the informational view of the Universe*—as suggested by ETS and in sharp contrast to conventional scientific paradigms. It is not difficult to see that accepting a qualitative difference between the informational capabilities of the prebiological Universe and of biological species inevitably leads to the acceptance of a scientific principle similar to the well-known and now almost unanimously rejected principle of vitalism[9]. Indeed, if biological species "invented" *fundamentally* new forms of representation that have not previously existed in nature, then this is tantamount to saying that biological information processing—and therefore biological representation mechanisms themselves—cannot be understood on the basis of physical representation mechanisms. In other words, postulating two fundamentally different informational mechanisms in nature *leads to the same undesirable situation* (in view of the ramifications) as postulating two fundamentally different classes of forces acting in the Universe.

[9] Vitalism suggests that, in addition to the known physical forces, there are not-yet discovered "vital forces" that are active in living organisms.

2.1 The lack of any adequate concept of class

It is well known that the concept of a class of objects is absolutely pervasive, both within science (e.g. isotope families, biological taxons, categories in cognitive science) as well as outside it (e.g. library classification schemes, fall shoes versus summer shoes). In view of the ubiquity of the class concept, many areas of information processing—e.g. PR (including speech and image recognition), data mining, information retrieval, bioinformatics, cheminformatics—rely on this concept as the central one. So, since the main burden of addressing the concept of class and the process of classification fell on these areas, of necessity they had to settle on some formalisms, and unfortunately, but not surprisingly, the researcher's (subconscious) choice has typically been the classical numeric and logical formalisms. However, as was mentioned in Sections 1.4–1.5, these conventional formalisms were not developed to address the needs of class representation, so the researchers have tried to adapt them for this purpose[10], to the extent that, presently, it is these adaptations that have become an obstacle to be overcome.

Thus, although together with many researchers, we firmly believe in the indisputability of the need for a theory of inductive learning, where we differ with others is in our insistence on evaluating such a theory based on the quality of class representation the theory offers: we need a (fundamentally) new formalism that clarifies what the concept of class *is*, since conventional theories contribute practically nothing towards this goal.

In light of this, on the technical side, the development of the ETS model was motivated by two considerations: the fundamental inadequacies of existing formalisms for class description and by the vision of the class description as a 'generating system' (Part III). As a result, it turned out that the differences between the conventional (numerical and logical) views of class and class description and those of the ETS formalism are as substantial as they could possibly be: one main difference with numeric formalisms being our insistence on a particular form of generativity in the definition of a class representation (Section 1.4). The main point is that, so far, the term "class" has been improperly understood, used, and applied: most of what are currently labelled as "classes" should not be viewed as such, but rather as *sets* of somewhat related objects that, however, do not share similar formative histories.

In particular, from the emerging perspective, many universally used *verbal descriptions of classes*, including those often offered in problems in various "ML challenges", should not be considered as such, mainly because most of them are not descriptions of classes, if the term "class" is to be understood as advocated here.

2.2 The ETS tenet: evolution of the Universe as the evolution of class generating systems

In general, the ETS framework is inspired by the view of the universe as a variety of interconnected and interacting, evolving classes (of processes) and hence of the corresponding class generating systems. What do we mean by a class generating system?

[10] Note that, even in natural languages, words simply *name* classes of objects/events rather than capture their (evolving) *representations*.

For a particular class of entities, by its generating system[11] we understand a non-deterministic system operating on actual entities and assembling them into larger entities (and eventually into class objects), guided by some hierarchical description of the class. The latter does not mean that the system "reads" this description, but rather that its instantiation *at a given location* is guided in this particular manner. The appearance of a new, or modification of an existing, generating system is a result of the interaction of several such systems.

As will be clarified in Part III, we think of a class generating system as a hierarchical generating system—in some sense similar to an abstraction of the embryo's development system[12]—that should be viewed as a class generating system that produces class objects. It is important to emphasize that the concept of a class generating system, or *class representation system*, now becomes, in a sense, more fundamental than that of the class itself, i.e. class objects become a product of the corresponding generating system.

Thus, we think of each object, as well as that object's representation, as a (relatively) stable structural object/entity that is being produced, or constructed, by the class generating system. The adjective stable, to which we will return in the next section, refers to the fact that the same local structural patterns appear in the same structural setting. For example, a single (vibrating) water molecule, observed over a very short time interval, can be thought of as a stable molecular process, which itself is composed of faster-running subatomic stable processes. Also, the production of *different* water molecules is guided by the *same* generating system, and thus quite naturally, we get a *class* of water molecules. In science, the more abstract class generating systems have, so far, remained behind the scene, while their products, i.e. the class elements they produce, are more familiar to us.

What is the relation between an object and its ETS representation? Again, ETS representation is a temporal event-based representation, which is supposed to capture an object's "informational" structure, i.e., we are postulating that "informational" structure is the same as temporal event-based structure. So, the above question becomes: What is the relation between a "physical" object and its event-based representation? We hypothesize that a "physical" object is a physical instantiation of its event-based (informational) representation, i.e., as each event is being played out, the physical "flesh" is automatically being put on the informational "bone". Thus, as some leading physicists have anticipated, the *informational structure becomes primary* (see, for example [72], pp. 340, 341).

In view of its motivation and structure, we also expect ETS to be the first formalism suitable for modeling developmental processes, a need that has been acutely felt within the field of developmental biology:

> *Morphogenesis* remains one of the most poorly understood aspects of development. Although the genetic blueprints underlying the formation of many organs and structures are beginning to be worked out, the mechanisms by which encoded genetic information is translated into structure remain obscure. [53, p. 81]
>
> [N]o unified theory of modularity that captures the various uses of the term in evolutionary and developmental biology currently exists. *Nor do we have all the formal*

[11] For a more formal description, see Part III.

[12] For a popular (but inherently sloppy for a formally-trained reader) exposition, see any of [53], [54], [55], [56].

tools for the analysis of such systems. [53, p. 341, emphasis ours]

2.3 Structural processes, their classes, and their transformations

It has been known for a long time that "[t]ime has its origin *in the existence of both kinds of physical change, cyclic and non-cyclic, in the natural world that we know*" [57, p. 14].[13] What is the importance of the *two different kinds* of processes?

> Any physical process that is purely cyclical may be treated as an atemporal process as long as it is considered only in its own domain, without reference to the larger, temporal universe surrounding it. The successive oscillations of a photon as it moves through a vacuum, for example, are identical, and there is no change that gives a time coordinate. The motion of an electron about a nucleus, or the cycles of two isolated astronomical bodies rotating about each other, have a similar atemporal quality. On interaction with other systems these purely cyclic processes may become non-cyclic and thereby gain a temporal character.
>
> The cyclic process, then, can become a temporal process only by reference to non-cyclic processes. We may refer to the latter as *progressively* changing processes, in contrast to the cyclic processes in which we do not have progressive change except within any one cycle. We have seen that the progressively changing process does not give us our time concept because change in general does not provide a basis for time standardization and measurement. But likewise, uniform cycles of change will not by themselves give us our time concept, because without progressive change there is no distinction of any one cycle from another. [57, p. 14]

Consistent with the above subdivision of all processes into two categories, in the ETS formalism, the two central concepts (besides, of course, that of class) are those of the *structural process* and the *transformation*, where the former corresponds to a generalization of the above "cyclic" process and the latter to the above "non-cyclic" process.

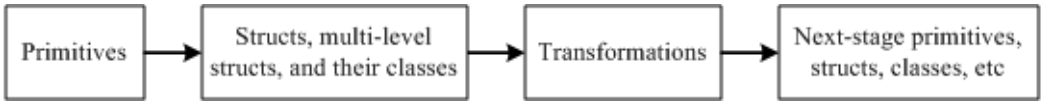


Figure 2: Logical sequence of the central ETS concepts.

Before informally discussing some of the central concepts, Figure 2 gives their logical sequence. Also, throughout the paper, *we reserve the following special meaning for the noun "event"*: an event is the interaction ('transformation') of one or several adjacent 'processes' resulting in the creation of new 'process(es)', e.g. the event of a photon emission by an electron (see Fig. 3), the event of a fertilized egg formation out of sperm and egg processes, the event of a cell division, the event of a two-car collision.

The concept of stable structural process—which is viewed and represented as a (temporal) *sequence of interconnected structured events*—is associated with the concept of a structural

[13] Although, in Section 2, we often come back to these two terms, they are not used within the formalism itself.

object representation. Such representation can be approached from two perspectives: the global and an agent's perspectives. Or, more accurately, in the case of an agent's perspective, it is supposed to encapsulate the concept of an object *observed* over some period of time by the agent, who must rely on its own representational resources. The global perspective deals with the (complete) object representation, relying on the entire (or restricted) evolution of the universe. From the global perspective, the adjective "stable" refers to the regular (or periodic) nature of the process responsible for the object's regeneration. From the agent's perspective, stable refers to the stability of the corresponding perceptual process: e.g. it refers to a simple fact that, during the continuous observation of an object by the agent, the same "perceptual features", or subpatterns, must reappear as the observation "returns to the same place". Since *objects are represented as structural entities* in the ETS formalism, a more accurate paraphrasing of the last sentence is: the object representation—i.e. the corresponding above sequence of (sensed) structured events—is actually constructed during the *interaction between* the observing (sensing)[14] agent and the target object process. It is not difficult to see that the choice of subpatterns/subentities—out of which a structural representation is built—must depend on both the agent's built-in sensory capabilities as well as its (inductive) experience[15].

At a particular (hierarchical) 'stage' of representation, a subpattern in the represention of an object—in our terminology a 'substruct'—is a temporal assembly of the very basic building units, which we call primitive transformations, or simply 'primitives' illustrated in Fig. 3. Some substructs are delineated in Fig. 4 by various lines. *Having fixed the representational stage*, the primitives at this stage are treated as indecomposable units designating the basic events associated with the disruption (transformation) of the above—indecomposable at this stage—cyclic processes. In Fig. 3, cyclic processes are called 'primal' processes and are shown as lines connecting the primitives. At the *previous* representational stage, however, a primitive can be "opened up" into a *non-primitive* transformation (Fig. 5). The nature of such a transformation can most naturally be understood as denoting a macro-event that is responsible for the transformation of one set of (interacting) structural processes into another set of processes. In other words, 'initial' interacting processes are transformed into 'terminal' processes [16] as shown in Fig. 5, where the structural processes delineated with solid lines are, in fact, opened up prototypes of the next-stage indecomposable primal elements/processes which connect primitives.

It is useful to think of a primitive as capturing the non-regular event—or the above "non-cyclic" process—corresponding to the "standard", i.e. entrenched, interaction of initial primal processes (or the above "cyclic" processes). A typical primitive would have a terminal process that doesn't belong to any of the initial classes of processes, and hence it can be thought of as capturing an event responsible for the generation of new, or transforming existing, *classes* of structural processes.

[14] The sensors involved, in contrast to the conventional ones, are structural. It is also important to note that all biological sensors are of a chemical nature, and are therefore structural.

[15] A substantial part of the previous experience of the agent's "species", for efficiency considerations, could be embedded in hardware rather than in software. From a biological point of view, the entire cell structure (including DNA, cytoskeleton, etc.) constitutes such hardware.

[16] Recall the analogy of several particles before and after a collision.

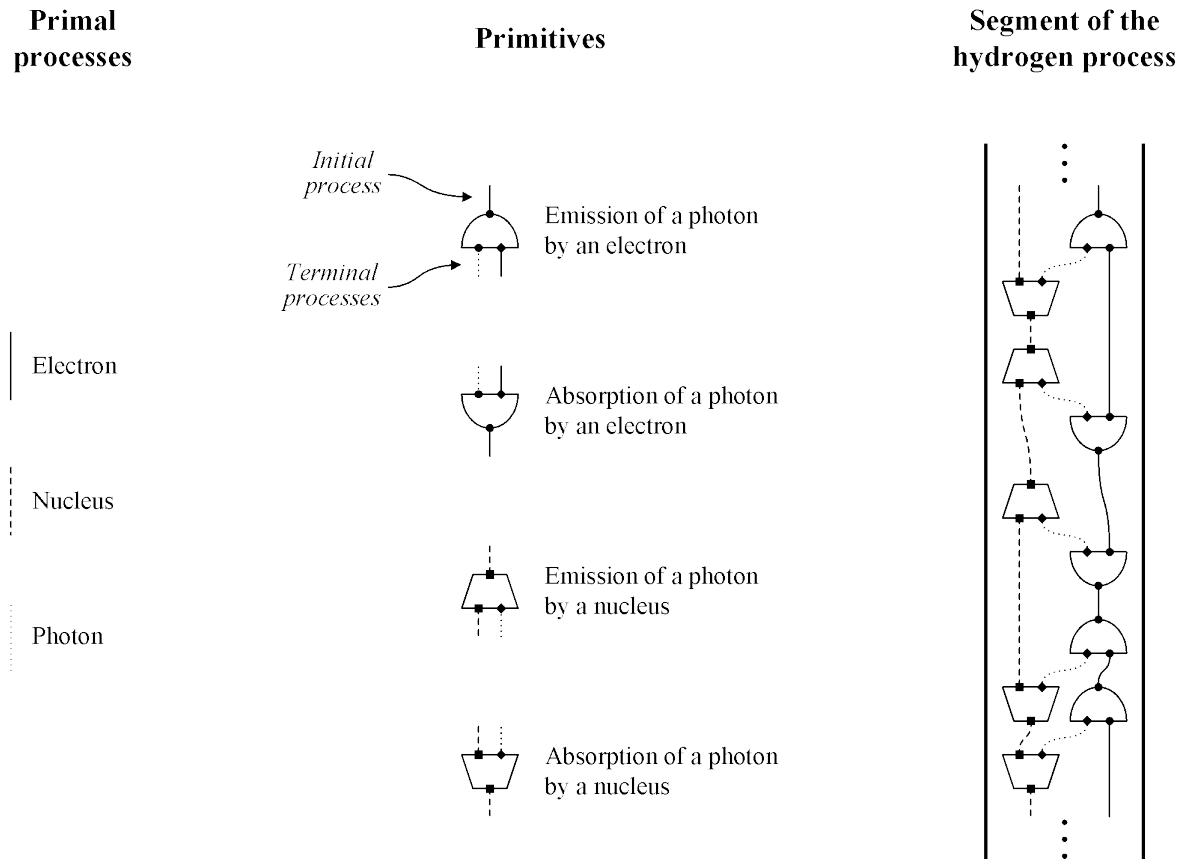


Figure 3: Pictorial representation of a process corresponding to a hydrogen atom over an extremely short time interval: three kinds/classes of subatomic entities (first column), four primitive events (second column), and one conceivable event scenario for the hydrogen process (third column).

In light of the above (and what follows), we stress that in spite of some *superficial* similarity with graphs, relying on this similarity is very counter-productive, mainly in view of the temporal nature of ETS representation.

So, what is the mechanism responsible for maintaining the integrity of a class of structural entities? In Part III we introduce the concept of (hierarchical) class representation, whose main component is the concept of multi-level class generating system, responsible for "overseeing" the construction of the entities forming that class. The construction proceeds step-wise by initiating the (class) scheme for assembling the higher-level *components* of a class element and then realizing this scheme by (recursively) propagating structural constraints to the *corresponding previous-level construction processes* (see Figs. 22–28) [17].

We should point out that the initial impression of ETS that one might have from the above figures as a very visual formalism is in no way false.

[17] Note that, in the context of *class representation*, we speak of 'levels', while in the context of *class interactions* (i.e. the transition from a transform to a next-level primitive as shown in Fig. 5), we speak of 'stages'. More accurately, when we speak of 'levels', we always refer to those associated with the *descriptions of classes within a fixed 'stage' of representation*, where each such stage refers to an organizational unit in the (global) multi-stage representational hierarchy.

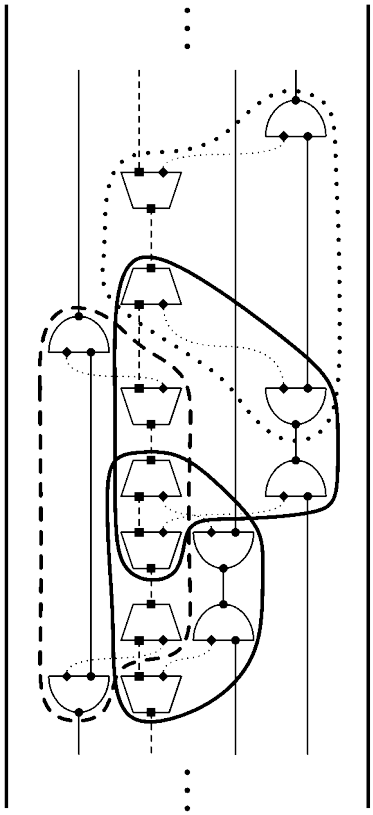


Figure 4: Pictorial representation of one conceivable event scenario for a lithium atom process/struct with four of its substructs delineated. As can be seen from the figure, the two substructs identified by solid lines are elements of the same constituent class of patterns, which participate with elements of other classes in the construction of the shown lithium process.

2.4 Some preliminary connections with physics

To see the connection between the above concept of structural process and the concept of the elementary particle in modern physics, the following insight should be useful:

> De Broglie's argument began with the supposition that "the basic idea of quantum theory is the impossibility of considering an isolated fragment of energy without assigning a certain frequency to it." The particles of radiation—and of matter as well—had a *level of existence that was fundamentally a "periodic [structural] process"*. [58, emphasis ours]

Moreover, as also mentioned on p. 27, the very useful in quantum physics concept of Feynman diagram ([61]–[64]) can be considered as a "pictorial" physical version of the structural representation proposed here. However, most interestingly, the ETS formalism suggests a compelling explanation of what some refer to as the most profound mystery in modern physics, i.e. that of entanglement. Entanglement is a term introduced by Schrödinger, and it presently refers (in the case of two particles) to the relatively well observed phenomenon of the *instantaneous* transfer of the effects of measurement on one particle to another particle that interacted with the first one at some earlier time, *independent of the present distance*

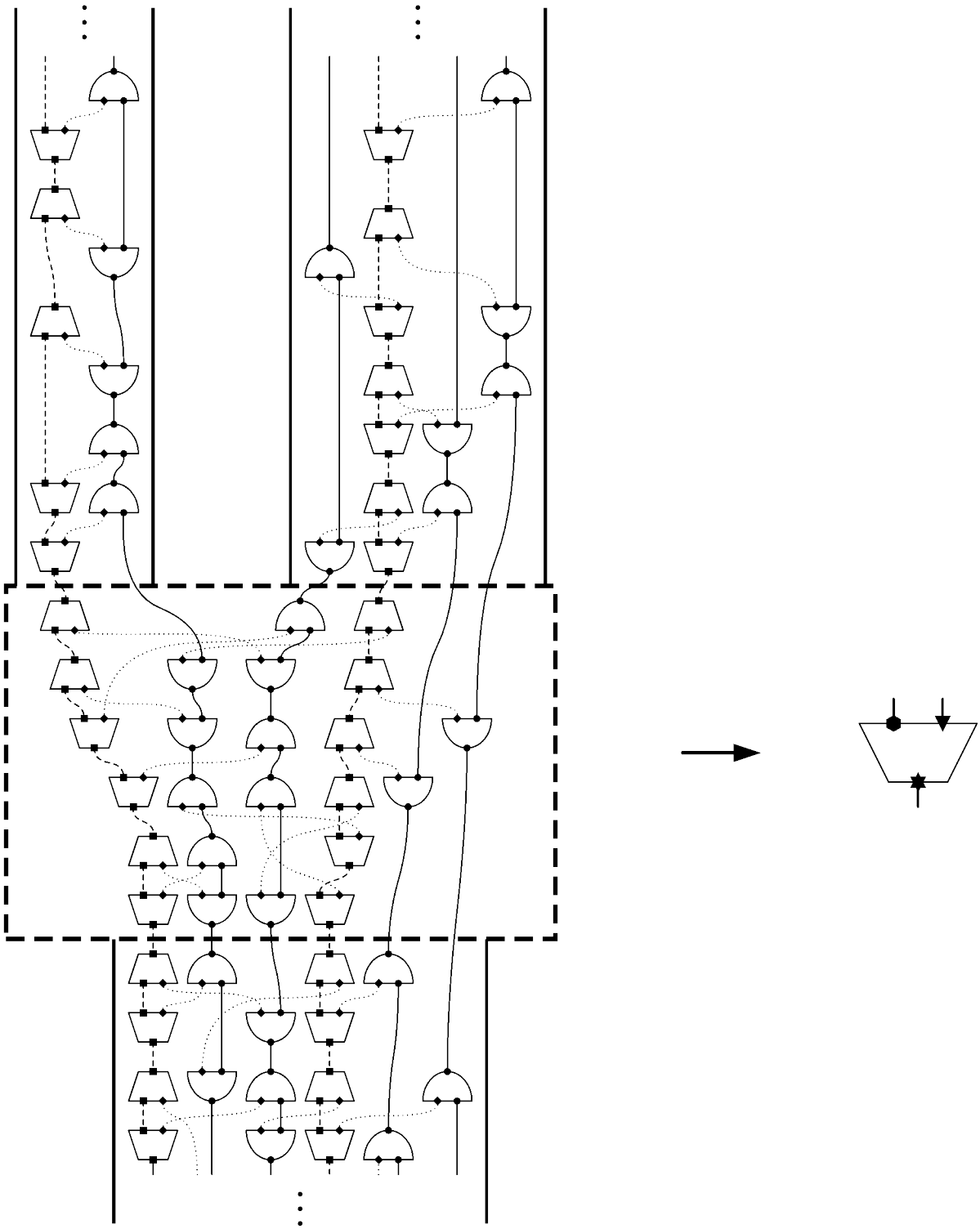


Figure 5: Pictorial representation of a transformation corresponding to the formation of a lithium hydride molecule (bottom process) from particular hydrogen (Fig. 3) and lithium processes (top: note the reoccurring temporal/structural patterns). The "body" of the transform (heavy dashed line) depicts an *over-simplified* restructuring of the two initial entities into the terminal one. On the right, we show the corresponding *next-stage* (lithium hydride formation) primitive.

between them (for popular expositions, see [65], [66]). From the point of view of the ETS formalism—whose main underlying assumption is the indispensability of formative history for representation—the above "transfer" of the effects of measurement can be explained as reflecting the following fact: conventional measurements on the first particle contribute to the global formative history (by recording in it the effects of the measurement process itself), which in turn affects all *future* measurement processes (even those addressing "past" events), since they now have to deal with this modified formative history.

In Figs. 3–5, we offer a (crude) illustration of an ETS representation in physical chemistry, i.e. we present a *naïve* structural representation of the formation of a lithium hydride molecule from its two constituent atoms: hydrogen and lithium. Note that, for simplicity, otherwise important electron-electron interactions are neglected, a nucleus is treated as an indivisible or single entity, and the energy, momentum, etc. of the various subatomic constituents are also neglected. It is interesting that one does not need to differentiate between atoms and small molecules in the ETS formalism[18]: both may be considered as being stable structural processes at the same stage.

Turning to the concept of time scales (see Fig. 6)—and keeping in mind that, in ETS, a transition to a new stage is related to the incorporation of a new transformation as a next-stage primitive transformation (Fig. 5)—the ETS formalism suggests that a change in the discretely structured scale of time in the universe is associated with some such transitions: a coarser time scale appears as more complex (next-stage) structural entities are instantiated. In other words, once the generating system begins to assemble new, more complex entities/processes, the overall generation time increases. Historically, some of these transitions were associated with transitions to atomic levels, molecular levels, etc.

We will come back to connections with physics at the end of the next section.

2.5 Objects as epiphenomena of class generating systems

As the section heading implies, according to the ETS formalism, observed objects are not what they appear to be. This point is not as controversial as it seems if all objects are treated as organisms, having developmental as well as evolutionary histories. Thus, a developed organism should also be viewed as an epiphenomenon of *both* these histories as is the case in modern biology: if we tinker with either one of the histories, we change the organism, and the more we tinker, the bigger the changes become. This is what actually happens during evolution. As far as objects are concerned, when we look at such an object as a chair, it also has its "developmental" (i.e. production) and "evolutionary" (i.e. conceptual) histories. In light of this, it makes perfect sense to assume that any biological representational model should be based on such principles: we believe that during the period of time in which a visual system interacts with a chair, for instance, it actually constructs a generative representation of the chair *as perceived by the viewer*. The ETS formalism suggests such a view of reality.

It appears that one can "blame" classical physics for the current state of affairs in which objects (together with the corresponding measurements), rather than their formative histories, are at the center of attention. The latter, in turn, is the result of the state of affairs in

[18] This does not at all preclude the emergence, at higher stages of representation, of macromolecular structures such as proteins.

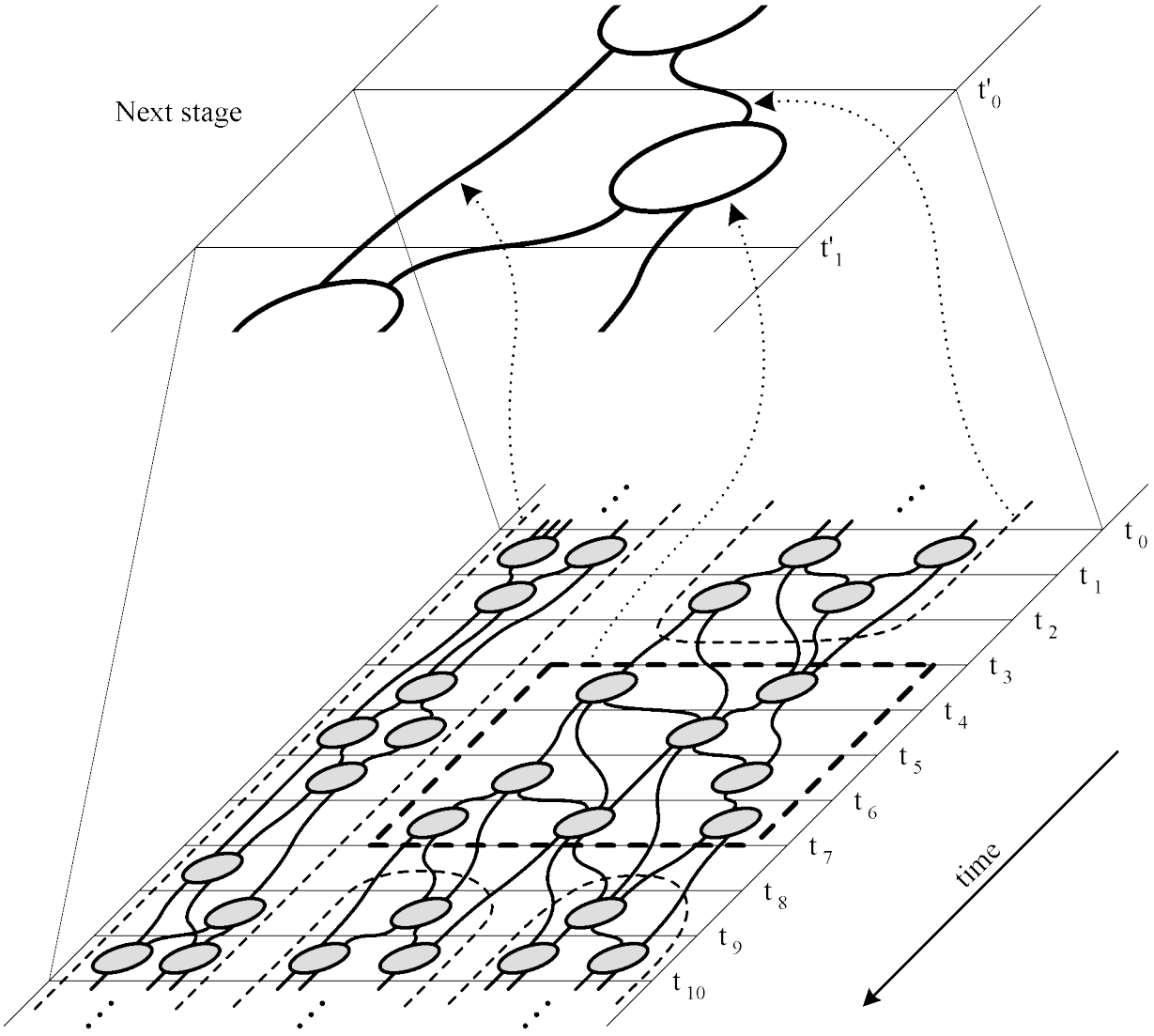


Figure 6: Simplified ETS representation with different time scales for each of the two depicted stages. The circles stand for primitive transformations and the lines between them signify basic structural entities for the corresponding stage. The thin dashed lines identify entities and the heavy dashed line identifies the only shown transformation. The hierarchical decomposition of each structural entity is suppressed. (Note that the upper stage's time scale is measured in coarser units, i.e. t'_0 corresponds to t_0, t'_1 corresponds to t_{10}, etc.)

mathematics which, historically, has been concerned only with numeric forms of representation. The concept of structural representation proposed here, on the other hand, brings to the fore the question of how the object's structure has emerged. Any structure must have emerged incrementally, and both level-wise (within a class) and stage-wise (class interactions), which is what we observe in the universe. For example, molecular structures are now inconceivable without reference to atomic structures. We—together with other scientists and philosophers, e.g. see Sec. 1.5 and the epigraph on p. 36—believe that it is such currently unobservable processes that are responsible for the generation of observable structure, and therefore *they* should be of primary interest, as opposed to the observable structures themselves. The latter point of view is quite consistent with the one that emerged about 80 years ago in physics, as can be seen from the following quotations from the book authored by an outstanding physicist and astronomer Sir James Jeans (emphasis ours).

> [Matter] cannot of itself make a direct impression on our senses; such impressions are only made by *physical "events" occurring in matter*. Strictly speaking, we do not see the sun; we see events taking place in the sun. The sun only affects our senses because a continuous re-arrangement of electrons in the solar atoms results in the emission of light. In the same way, we do not see a chair, but the event[s] of daylight or electric light falling on a chair. If we stumble against the chair in the dark, we do not feel the chair, but the event of a transfer of energy and momentum between the chair and our bodies. [59, p. 11]

The remaining quotes are taken from the chapter characteristically titled "Events" of the same book [59, pp. 293, 295]:

> Thus the "world-line" of a particle is, strictly speaking, not a line at all, but is a continuous and unbounded curved region, and must logically be separated into small curved spots—*the particle resolves itself into events. Most of these events are unobservable; it is only when two particles meet or come near to one another that we have an observable event which can affect our senses. We have no knowledge of the existence of the particle between times, so that observation only warrants us in regarding its existence as a succession of isolated events.*
>
> . . . . . .
>
> Matter gives us a rough and easily understood, but not a true, picture of the reality underlying physical phenomenon. *But we now begin to suspect that events and not particles constitute the true objective reality*, so that a piece of matter becomes, in Bertrand Russell's words,
>
> "not a persistent thing with varying states, but a system of inter-related events. The old solidity is gone, and with it the characteristics that, to the materialist, made matter seem more real than fleeting thoughts."

Then Jeans (and Russell) go on to suggest a view of nature consistent with the informational view expounded by ETS:

> This at once takes all force out of the popular objection that mind and matter are so unlike that all interaction is impossible. With matter replaced by events, the objection

is no longer tenable. We see the territory on both sides of the mind-body bridge occupied by events, and as Bertrand Russell says (*[An] Outline of Philosophy*, [1927,] p. 311):

"The events that happen in our minds are part of the course of nature, and we do not know that the events which happen elsewhere are of a totally different kind."

2.6 ETS as a multi-stage representational formalism

As was also discussed in Section 2.3, the emergence of each *new stage* is associated with the discovery and consolidation by an agent of the *first transformation at the currently highest stage*. Thus, together with the new stage, *the first primitive transformation* at this stage is then created on the basis of that transform as shown in Fig. 6. However, the discovery of a transform at any, except the last, stage results in the expansion of the set of primitive transformations at the next stage.

From the agent's perspective, the (external) input is a sequence of sensory events, represented as primitive transformations of the initial stage (see Fig. 7). It is quite reasonable to assume, however, that agents might be equipped with additional kinds of sensors, capable of directly identifying some subpatterns, or classes of processes.

A useful metaphor for capturing the above multi-stage structure is a multi-stage "evolving representational tower" which can directly interact with (external) processes *only* at the initial stage(s). In the language of the ETS formalism, each stage records, or represents, observable processes by means of its own primitives, with which it represents the corresponding structural fragment from the previous stage (thus compressing the *representation* of the external data). Thus, each stage k of this tower is responsible for the detection of regularities, i.e. stable processes and transforms, in the external events at the k-th resolution stage, relying on the (condensed) representation passed up from the previous stage (Fig. 6).

We note that, although it may appear that the transition to a new stage of representation is necessitated only by the need to deal more effectively with the complexity of representation[19], the resulting higher stage representations in turn begin to play an *active* organizing role influencing the appropriate lower stages: the degrees of freedom at a lower stage are restricted by the structures emerging at higher stages (e.g. the symbiosis of cells influences the degrees of freedom of the constituent proteins involved).

2.7 Facing the present state of science

As an *initial (but partial)* applied insight and until structural sensors are built, we propose to face the *present* scientific reality with the help of the following two perspectives: *object view* and *event view*. The classical object view encapsulates the *common* scientific view of reality, while the proposed event view encapsulates the ETS, or information process, view.

The conventional scientific view of reality is related to observations in the *object environment*. E.g. in chemistry, observations are those related to atoms and molecules (two separate oxygen atoms covalently bond), and physical theories attempt to describe these observations in terms of states of *objects*, represented via the vector space formalism. On the other hand,

[19]This results in the well-recognized phenomenon called "chunking" ([60]).

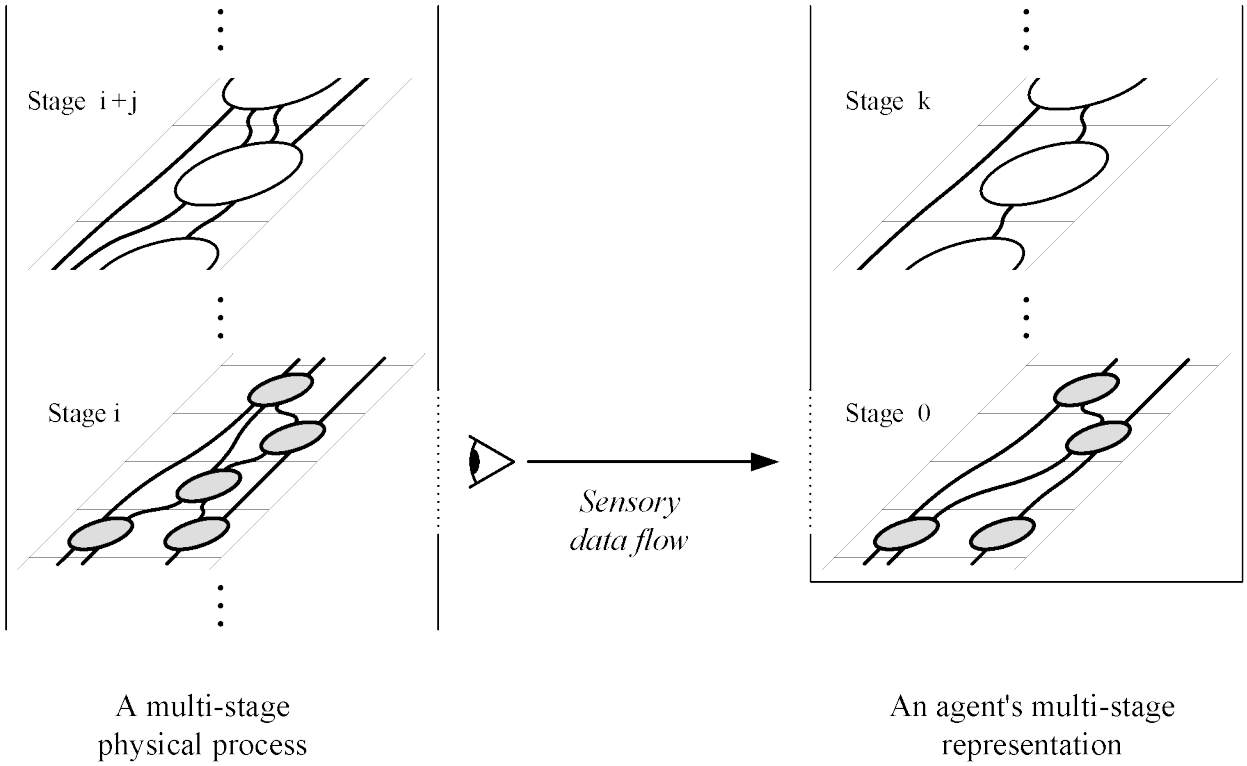


Figure 7: An actual multi-stage physical process (left) and an agent's representation of it (right), mediated by sensors (center).

the ETS model emphasizes a *process* view of reality, in which, as was mentioned above, *transforming* events in the object environment, rather than the objects themselves, are the basic subject of study (e.g. in the oxygen example, the focus is on the event corresponding to the *transformation* or *change* responsible for the formation of an oxygen molecule). Again, the latter view of the environment, which we call the *event environment*, insists on the primacy of the information process rather than on the primacy of the objects themselves.

Given the present state of science, i.e. an object-centered view, one may choose to face this situation, for the time being, by admitting the above two environments and creating an interface between them: the ETS model operates with **ideal events** that correspond to **real events** in the object environment. A real event is accounted for (in the event environment) by its idealized version (its **idealization**), while in the object environment a real event is accounted for by the **realization** of an ideal event (see Fig. 8). Note that the event environment can easily account for, in particular, such events as the "appearance" and "disappearance" (e.g. in the case of interacting particles) of objects as well as various changes in the relationships among the objects.

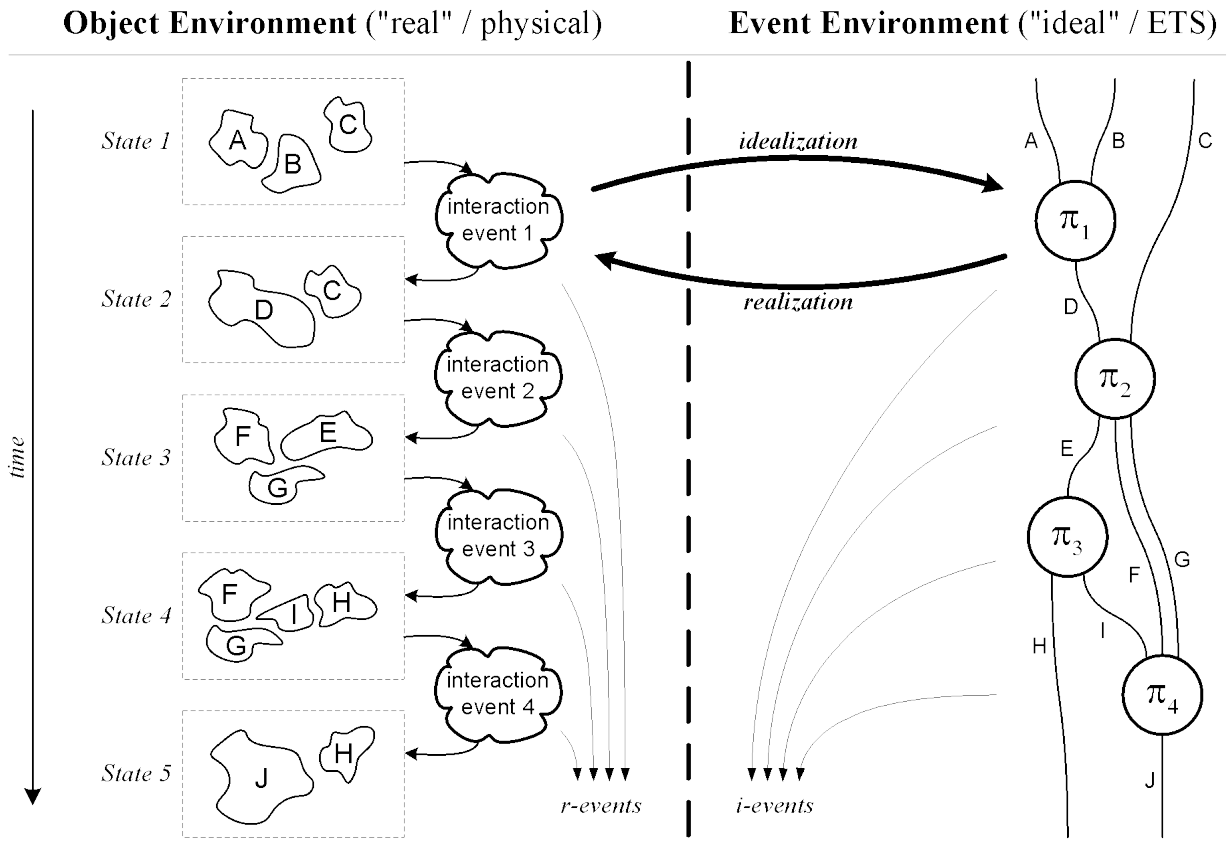


Figure 8: Event environment versus object environment. In State 1, three objects/processes (A, B, C) are shown. As a result of the first real event, A and B merge to form D in State 2. The corresponding ideal event (primitive π_1) is depicted on the right. Three subsequent state changes are also illustrated: D and C are transformed into E, F, and G; E is split into I and H; F, G, and I merge to form J.

Part II
ETS basics

> [T]he above remarks ... prove that whatever the [mathematical language of the central nervous] system is, it cannot fail to differ considerably from what we consciously and explicitly consider as mathematics.
>
> J. von Neumann, The Computer and the Brain, 1958

3 Basic level primitives and class links between them

In this section we introduce the basic constructive elements of the formalism, in our case the elementary transformations (or *elementary "ideal events"*).

In what follows, the term **enumerable set** refers to either a finite or countably infinite non-empty set.

We would also like to emphasize that, in view of the present lack of an appropriate basic mathematical language for dealing with structured entities, we must of necessity rely on conventional set theoretic language. We do expect that this situation will be remedied once the issue of structural representation has received adequate attention.

Notational convention 1. In this paper, the names of sets and their tuples begin with a capital letter, while the names of mappings are in small letters.

We use the hat accent in $\widehat{\alpha}$ to denote the *special* entity called the name of the associated formal object α. Of course, "naming" can be viewed as an injective mapping from a set of such objects to the set of their names. ◗

The following definition introduces the first basic concept, that of a primitive transformation. As was mentioned in the Introduction, by 'primitive transformation' we mean a microevent responsible for transforming one set of adjacent/interacting structural processes into another set of processes.[20] In other words, the concept of primitive transformation encapsulates that of a *"standard" interaction* of the several processes involved.

Throughout the paper, when we speak of a "structural process"—often as being an element of a class of such processes—this should be more accurately understood as a "*representation* of the process", i.e. as a *structural entity* standing for our current representation of the process.

We assume that a set of standard or 'primal' disjoint[21] classes of processes has been specified. Each element of such a class is some *structural entity* whose intrinsic structure is suppressed, i.e. each entity must, at this initial stage, be treated as unstructured, or indivisible, and indistinguishable from any other entity in the class. For this reason, in the

[20] Recall the analogy of several particles before and after a collision.

[21] In general, *assuming complete knowledge of the classes*, including their representations/descriptions, classes in nature appear to be disjoint (see the discussion after Def. 13 on p. 43).

following definitions we *de-emphasize the concept of "process/entity"* and speak simply of elements of classes.[22] Still, one *must not forget* that each such element is a representation of an observed process (see Remark on p. 81).

Definition 1. Let m, n be small positive integers and

$\mathbb{C} = \{ C_1, C_2, \ldots, C_m \}$ — C_i is a given (enumerable) **primal class**, $\forall\, i, j \quad i \neq j \qquad C_i \cap C_j = \varnothing$,

$\widehat{\mathbf{\Pi}} = \{ \widehat{\boldsymbol{\pi}_1}, \widehat{\boldsymbol{\pi}_2}, \ldots, \widehat{\boldsymbol{\pi}_n} \}$ — $\widehat{\boldsymbol{\pi}_i}$ is a given **name of an abstract primitive**.

We first introduce the following auxiliary concepts and notations for each $\widehat{\boldsymbol{\pi}_i}$, $1 \leq i \leq n$:[23]

$\mathrm{Init}(\widehat{\boldsymbol{\pi}_i}) = \langle C_{j_1}, C_{j_2}, \ldots, C_{j_{p(i)}} \rangle$ — is a *given* tuple of primal classes called the **tuple of initial classes**, or **initials**, $p(i) > 0$,

$\mathcal{L}_i, \quad \mathcal{L}_i \subseteq C_{j_1} \times C_{j_2} \times \ldots \times C_{j_{p(i)}}$, — is a *given* **set of labels associated with** $\widehat{\boldsymbol{\pi}_i}$, such that no two constituent elements of any tuple $\langle c_{j_1}, \ldots, c_{j_{p(i)}} \rangle \in \mathcal{L}_i$ are equal,

$\mathrm{Term}(\widehat{\boldsymbol{\pi}_i}) = \langle C_{k_1}, C_{k_2}, \ldots, C_{k_{q(i)}} \rangle$ — is a *given* tuple of primal classes called the **tuple of terminal classes**, or **terminals**.

On the basis of the above givens, define $\boldsymbol{\pi}_i$ as a set

$$\boldsymbol{\pi}_i \;=\; \{ \boldsymbol{\pi}_i(\boldsymbol{a}) \mid \boldsymbol{a} \in \mathcal{L}_i \}$$

whose generic element $\boldsymbol{\pi}_i(\boldsymbol{a})$ is:

$$\forall\, \boldsymbol{a} \in \mathcal{L}_i \qquad \boldsymbol{\pi}_i(\boldsymbol{a}) \;=\; \pi_{i\boldsymbol{a}} \;\stackrel{\text{def}}{=}\; \langle \widehat{\boldsymbol{\pi}_i}, \mathrm{Init}(\widehat{\boldsymbol{\pi}_i}), \mathrm{Term}(\widehat{\boldsymbol{\pi}_i}), \boldsymbol{a} \rangle$$

(see Fig. 9) [24]. Set $\boldsymbol{\pi}_i$ is called an **abstract primitive transformation**, or simply **abstract primitive**, and any one of its elements $\pi_{i\boldsymbol{a}}$ is called a corresponding (concrete) **primitive**. We denote by $\mathbf{\Pi}$ the finite set of all abstract primitives $\boldsymbol{\pi}_i$, $1 \leq i \leq n$, and by Π the set of all (concrete) primitives.

[One should think of a concrete primitive $\pi_{i\boldsymbol{a}}$ as designating a particular kind of interaction between the (initial) processes in label-tuple $\boldsymbol{a}$, the outcome of which is a *non-deterministically*[25] specified element of $C_{k_1} \times C_{k_2} \times \ldots \times C_{k_{q(i)}}$, which is the reason why label $\boldsymbol{a}$ cannot point to a particular element in the latter set product: if one were to observe the single event denoted by a concrete primitive, then in contrast to the initial processes, all of the corresponding terminal processes would be "in progress".] ▶

[22] Clearly, the corresponding *real* processes from the same class must, in *some* sense, be similar to each other. The sense in which they are similar is addressed later, in Part III.

[23] Note that we do not forbid, for example, that $C_{j_1} = C_{j_2}$. Moreover, the index $p(i)$ of $j_{p(i)}$ simply signifies a natural number that is a function of i.

[24] Both of the following notations, i.e. $\boldsymbol{\pi}_i(\boldsymbol{a})$, $\pi_{i\boldsymbol{a}}$, will be used. Note that $\pi_{i\boldsymbol{a}}$ should more accurately be interpreted as $[\pi_i]_{\boldsymbol{a}}$.

[25] This is true since a terminal entity cannot be fully identified until it is "absorbed" by another primitive event (see beginning of Section 4).

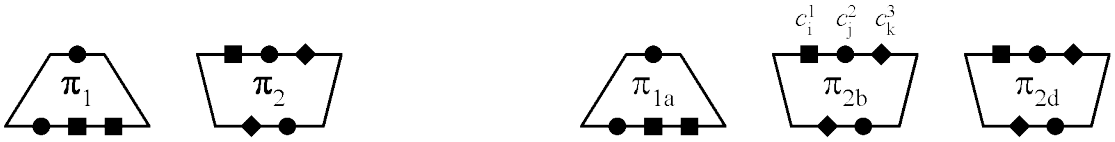


Figure 9: Pictorial illustration of two abstract primitives (left) and three corresponding concrete primitives (right). The last two concrete primitives belong to the second abstract primitive. The initial classes are marked as various shapes on the top, while the terminal classes are shown on the bottom. Unfortunately, the corresponding shape does not distinguish between the class, in an abstract primitive, and its element, in the concrete primitive. (The only processes identified are the initials of π_{2b}: $b = \langle c_i^1, c_j^2, c_k^3 \rangle$, where c_t^s is the t^{th} process from primal class C_s.)

As was mentioned in the Introduction, in general, one can think of an abstract primitive $\boldsymbol{\pi}_i$ as an entity responsible for the modification of existing or generation of new primal classes. In a formal setting, however, an abstract primitive can be thought of as simply transforming a tuple of initial classes, $\text{Init}(\widehat{\boldsymbol{\pi}_i})$, into a tuple of terminal classes, $\text{Term}(\widehat{\boldsymbol{\pi}_i})$.

[Important] **Remark 1.** For a generalization of the concept of primitive transformation that is very useful in an applied setting, see Appendix. However, we advise not to study the appendix before becoming comfortable with the basic concepts in Parts II and III, since its main utility becomes much more apparent when one begins to work on applications. ◗

Remark 2 (neurons). We would like to draw one's attention to the similarity between our primitives and biological neurons, which in this case is less superficial than that between the units of (artificial) neural networks and neurons: we expect that a neuron plays a similar *representational* role as a primitive does, possibly participating with several other neurons in implementing a primitive. ◗

Remark 3 (selection of primitives).[26] Regarding the selection of primitives, one cannot overestimate the following point: as was mentioned above, the ETS formalism insists on approaching the process of "data" representation afresh and in a much more careful manner than is traditional in information processing and other sciences. ETS suggests that, from an informational point of view, "data" should now be approached and treated in a *generative setting*. In other words, as will become clear from Part III, data emerges already partitioned into classes, where the elements of a class are entities "produced" by the corresponding class generating system, which essentially represents the class. Thus, a concrete data representation must now be treated as the result of an *appropriate* class generating system, ensuring that the object and class representations are properly correlated. This is definitely not the case with conventional generative (including graph) grammar models, in which the *nature of data representation* is not addressed and is, therefore, not at all correlated with the corresponding generative grammar (e.g. strings do not come with grammars embedded in, or attached to, them). Hence, the selection of primitives should be approached in this light: they are basic class transforming events in a carefully chosen process view of the environment, i.e. *the representation of an object refers to a relevant structural process in such an*

[26] See also Important Remark 4 on p. 86.

environment. In particular, we strongly recommend that all attempts to adapt conventional discrete "representations" (e.g. strings, trees, and graphs) to the above generative setting be abandoned, as they *impede investigation into the relevant generative mechanisms based on the hypothesized formative histories of objects in the environment*.

In regard to the number and structure of primitives, we expect that an appropriate *generalization* of the situation discovered in elementary particle physics will hold true: the number of different vertices in Feynman diagrams is quite small and all of them have similar structure (see, for example, [63]). ◗

It is useful to note, however, that despite the immediate analogy between our primitives and the "vertices" of Feynman diagrams (see, for example, [61]–[64]), the approach proposed here is a much more careful and general development, from the more abstract point of view of a representational formalism.

Notational convention 2. To simplify the notation in what follows, instead of $\mathrm{Init}(\widehat{\boldsymbol{\pi}})$ or $\mathrm{Term}(\widehat{\boldsymbol{\pi}})$, we will use the notation $\mathrm{Init}(\boldsymbol{\pi})$ or $\mathrm{Term}(\boldsymbol{\pi})$, respectively, i.e. we drop the hat accents.

Also, for the above primitives $\boldsymbol{\pi}_i$ and π_{ia}, we will use the following convenient notations:

$\overline{class}(\boldsymbol{\pi}_i\,, r)$ as referring to the r$^{\mathrm{th}}$ class C_{j_r} in the tuple $\mathrm{Init}(\boldsymbol{\pi}_i)$,

$\underline{class}(\boldsymbol{\pi}_i\,, r)$ as referring to the r$^{\mathrm{th}}$ class C_{k_r} in the tuple $\mathrm{Term}(\boldsymbol{\pi}_i)$. ◗

The following definition might be viewed as a continuation of (or as closely related to) the previous one: in contrast to Definition 3, it introduces two closely connected relations that are *all potential*—but *not yet observed*—*relations* among pairs of primitives.

Definition 2. Based on the definition of the abstract primitive, one can introduce the following relation $\mathbf{CL}_{\mathbb{C},\boldsymbol{\Pi}}$,

$$\mathbf{CL}_{\mathbb{C},\boldsymbol{\Pi}} \;\subseteq\; \boldsymbol{\Pi} \times \mathbb{N}_{\mathrm{Term}} \times \boldsymbol{\Pi} \times \mathbb{N}_{\mathrm{Init}}\,,$$

with

$$\begin{aligned} \mathbb{N}_{\mathrm{Term}} &= \Big\{ u \in \mathbb{N} \;\Big|\; 1 \le u \le \max_i\big(\,|\mathrm{Term}(\boldsymbol{\pi}_i)|\,\big) \Big\} \\ \mathbb{N}_{\mathrm{Init}} &= \Big\{ v \in \mathbb{N} \;\Big|\; 1 \le v \le \max_j\big(\,|\mathrm{Init}(\boldsymbol{\pi}_j)|\,\big) \Big\}, \end{aligned}$$

defined as follows

$$\mathbf{CL}_{\mathbb{C},\boldsymbol{\Pi}} = \Big\{ \langle\, \boldsymbol{\pi}_i\,,\, u_i\,,\, \boldsymbol{\pi}_j\,,\, v_j \,\rangle \;\Big|\; \boldsymbol{\pi}_i\,,\, \boldsymbol{\pi}_j \in \boldsymbol{\Pi}\,,\quad \underline{class}(\boldsymbol{\pi}_i\,,\, u_i) = \overline{class}(\boldsymbol{\pi}_j\,,\, v_j)\,, \\ u_i \le |\mathrm{Term}(\boldsymbol{\pi}_i)|\,,\quad v_j \le |\mathrm{Init}(\boldsymbol{\pi}_j)| \Big\}.$$

We call this relation a **class link between abstract primitives**.

We also apply similar terminology and notation to the corresponding (concrete) primitives and speak of the (enumerable) relation $\mathrm{CL}_{\mathbb{C},\Pi}$,

$$\mathrm{CL}_{\mathbb{C},\Pi} \subseteq \Pi \times \mathbb{N}_{\mathrm{Term}} \times \Pi \times \mathbb{N}_{\mathrm{Init}} ,$$

defined as follows

$$\mathrm{CL}_{\mathbb{C},\Pi} = \Big\{ \langle \pi_{ia} , u_i , \pi_{j\mathit{b}} , v_j \rangle \;\Big|\; \pi_{ia} , \pi_{j\mathit{b}} \in \Pi , \quad \underline{class}(\boldsymbol{\pi}_i , u_i) = \overline{class}(\boldsymbol{\pi}_j , v_j) , \quad u_i \leq |\mathrm{Term}(\pi_{ia})| , \quad v_j \leq |\mathrm{Init}(\pi_{j\mathit{b}})| \Big\} .$$

Since, *in contrast to the situation with initial entities*, none of the (concrete) terminal entities for *independently considered* concrete primitive π_{ia} can be identified (see the last paragraph in Def. 1), it is natural to postulate that, if the above pair $\langle \pi_{ia} , \pi_{j\mathit{b}} \rangle$ of concrete primitives has been *actually observed*, the non-determinism with respect to the corresponding concrete terminal process is eliminated, i.e. what has in fact been observed is a concrete process "connecting" the two primitives. In other words, we postulate that the observed "connecting" process is, in fact, the v_j'th element in label b. We call relation $\mathrm{CL}_{\mathbb{C},\Pi}$ a **class link between concrete primitives** (see Fig. 10).

For any set Π_0, $\Pi_0 \subseteq \Pi$, we will use the notation $\mathrm{CL}_{\mathbb{C},\Pi_0}$ to denote the subset of $\mathrm{CL}_{\mathbb{C},\Pi}$ in which all π_{ia}'s and π_{jb}'s are from Π_0. ▶

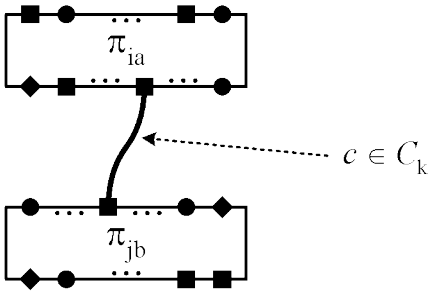


Figure 10: Generic element of the relation $\mathrm{CL}_{\mathbb{C},\Pi}$, where c is a structural entity from class C_k.

Thus, a class link between concrete primitives signifies the fact that one of the structural entities produced by the first primitive becomes (at certain point in time) an "input" to the second primitive.

4 Structs and struct assemblies

From the point of view of an agent interacting with its environment, it is quite natural to expect that, in addition to the *ability to observe primitives*, the agent's sensors must have the *capability of recording the interrelationships* among observed pairs of primitives. (The latter is accomplished by detecting primal structural processes first, and then inferring the primitives which are "connected" by them.) These primitives and their interconnections can be thought of as representing a *macroevent*, or "structural history".

Definition 3. A **struct** σ is defined as the following pair

$$\sigma \;=\; \langle\, \Pi_\sigma,\, \mathrm{SL}_\sigma \,\rangle\,,$$

where Π_σ is a finite subset of Π satisfying the condition that, for any two of its elements $\pi_{i\mathfrak{a}},\, \pi_{j\mathfrak{b}} \in \Pi_\sigma$, no constituent element of tuple $\mathfrak{a}$ is equal to a constituent element of tuple $\mathfrak{b}$, and the relation **struct link** SL_σ is a finite subset of $\mathrm{CL}_{\mathbb{C},\,\Pi_\sigma}$ such that:
$\forall\, \langle\, \pi_{i\mathfrak{a}}\,,\, u_i\,,\, \pi_{j\mathfrak{b}}\,,\, v_j \,\rangle \in \mathrm{SL}_\sigma \quad (\, \mathfrak{a} \in \mathcal{L}_i,\;\; \mathfrak{b} \in \mathcal{L}_j \,)$

(i) the *directed* graph of the following (auxiliary) binary relation[27] representing the projection of SL_σ onto $\Pi_\sigma \times \Pi_\sigma$

$$\mathrm{ATTACH}_\sigma \;=\; \left\{\, \langle\, \pi_{i\mathfrak{a}}\,,\, \pi_{j\mathfrak{b}} \,\rangle \;\middle|\; \langle\, \pi_{i\mathfrak{a}}\,,\, u_i\,,\, \pi_{j\mathfrak{b}}\,,\, v_j \,\rangle \in \mathrm{SL}_\sigma \,\right\}$$

is connected and acyclic[28]

(ii) $\forall\, \langle\, \pi_{i\mathfrak{a}}\,,\, u_i\,,\, \pi_{j\mathfrak{b}}\,,\, v_j \,\rangle, \langle\, \pi_{i'\mathfrak{a}'}\,,\, u_{i'}\,,\, \pi_{j'\mathfrak{b}'}\,,\, v_{j'} \,\rangle \in \mathrm{SL}_\sigma$

$$\pi_{i\mathfrak{a}} \,=\, \pi_{i'\mathfrak{a}'}\,,\; u_i \,=\, u_{i'} \quad\Longleftrightarrow\quad \pi_{j\mathfrak{b}} \,=\, \pi_{j'\mathfrak{b}'}\,,\; v_j \,=\, v_{j'}\,,$$

i.e. any terminal process can be connected to at most one initial process, and visa versa.

The set of all structs will be denoted Σ, and when $\Pi_\sigma = \varnothing$ $(\Rightarrow \mathrm{SL}_\sigma = \varnothing)$, struct σ will be called the **null struct**, denoted θ. ▶

Thus, when forming a struct σ, ATTACH_σ is the result of the sensors having recorded the various observed interrelationships among pairs of primitives. Note the role of the ATTACH_σ relation: two *separately-observed* primitives cannot reliably be attached to each other, since, by Def. 1, the specification of a terminal class yields an "incomplete" element of that class; however the "completed" element is obviously necessary to equate this terminal with a particular initial (which is a completed element of an initial class). Hence, the basic role of a sensor is to specify the relation ATTACH_σ, and therefore structs themselves.

We note that *when drawing* a struct σ, its primitives should be positioned in the following way: for any two *attached* primitives, the top primitive should be the one that "precedes" the bottom one in the binary relation ATTACH_σ (see Fig. 11).

It is important to note that the above definition suggests a far-reaching structural generalization of the Peano (inductive) construction of natural numbers ([7], [6], see also Fig. 12): we still deal with the temporal, or inductive, order of steps (not structures), in which small sets of primitive transforms are added atemporally in a *single* inductive step (e.g. $\{\, \pi_{1a},\, \pi_{2f} \,\}$ in σ_1 in Fig. 11). However, the resulting binary relation between primitives cannot now be interpreted, or understood, as a simple linearly ordered relation but is a more complex structure.

[27] The vertices of such an **attachment graph** are the elements of the relation's underlying set and the edges are defined by the ordered pairs of the relation. We will call this temporal relation between primitives **attachment**.

[28] A graph without cycles is called acyclic.

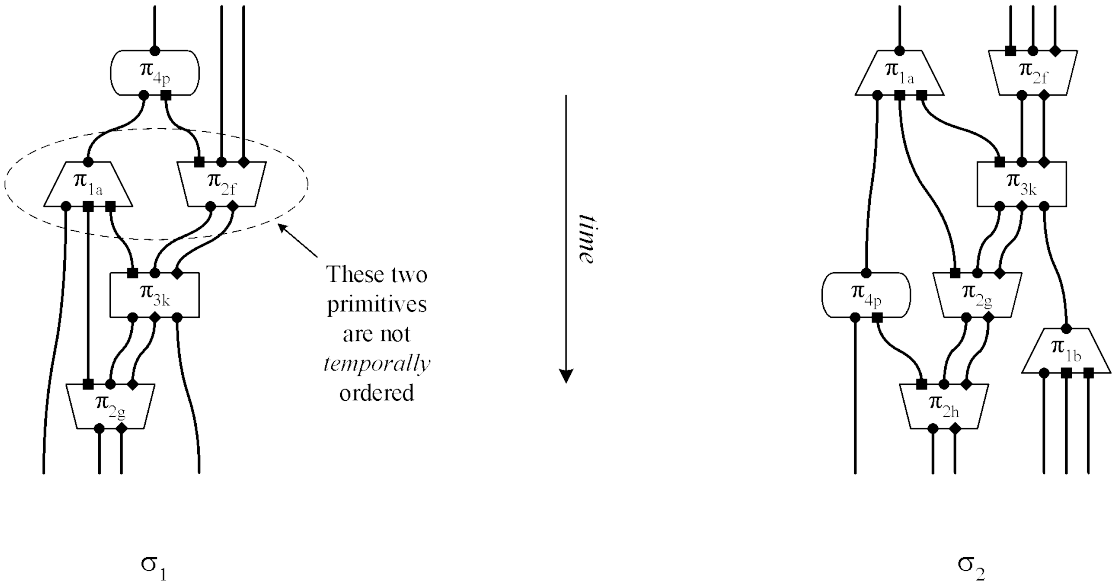


Figure 11: Two structs σ_1 and σ_2.

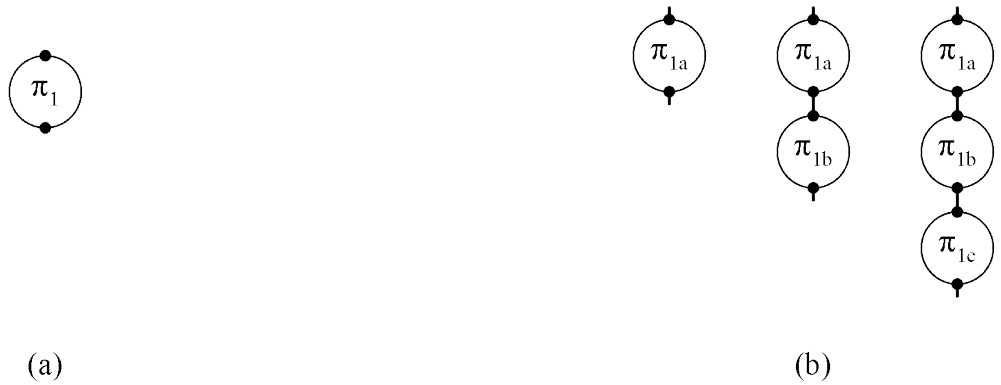


Figure 12: (a) The single primitive involved in the ETS representation of natural numbers. (b) Structs representing the numbers 1, 2, and 3.

[Important] **Remark 4.** Besides this immediate side of the generalization, one should note that, while a natural number records the simplest, linear sequence of (structurally) identical events, a struct records a temporal structure composed out of a variety of structural events. Thus, obviously, there are reasons to believe that *various non-temporal structures can be extracted more reliably* from this richer "temporal tapestry" as compared to that provided by a numeric representation. ◗

We now introduce a part-whole relation between two structs.

Definition 4. For two structs $\sigma_1 = \langle\, \Pi_{\sigma_1}\,,\, \mathrm{SL}_{\sigma_1}\,\rangle$ and $\sigma_2 = \langle\, \Pi_{\sigma_2}\,,\, \mathrm{SL}_{\sigma_2}\,\rangle$, we say that σ_1 is a **substruct** of σ_2, denoted $\sigma_1 \Subset \sigma_2$, if

$$\Pi_{\sigma_1} \subseteq \Pi_{\sigma_2} \qquad \text{and} \qquad \mathrm{SL}_{\sigma_1} \subseteq \mathrm{SL}_{\sigma_2}\,.$$

▶

The following important but non-central definition will be useful throughout the paper.

Definition 5. By a **relabeling** we understand an injective mapping f with domain $\mathcal{L}$, $\mathcal{L} \subseteq \bigcup_{i=1}^{n} \mathcal{L}_i$ ($\mathcal{L}_i$ is the set of labels associated with $\widehat{\boldsymbol{\pi}_i}$),

$$f\colon \mathcal{L} \to \bigcup_{i=1}^{n} \mathcal{L}_i$$

such that

$$\forall\, i \qquad f(\mathcal{L} \cap \mathcal{L}_i) \subseteq \mathcal{L}_i\,.$$

▶

To relate two structs based on their common substructures (for example, see Fig. 13), we must be able to "intelligently" (but consistently) modify the incidental labeling of the constituent primitives of one of the structs in such a way that the substructures, i.e. substructs, of interest become identical.

Definition 6. For a struct

$$\sigma \;=\; \langle\, \Pi_\sigma\,,\, \mathrm{SL}_\sigma\,\rangle$$

and a relabeling $f\colon \mathcal{L} \to \bigcup_{i=1}^{n} \mathcal{L}_i$, where

$$\mathcal{L} \;\supseteq\; \{\, \mathit{a} \mid \boldsymbol{\pi}_i(\mathit{a}) \in \Pi_\sigma \,\}\,,$$

the **relabeled struct**

$$\sigma\{f\} \;\stackrel{\text{def}}{=}\; \langle\, \Pi_{\sigma\{f\}}\,,\, \mathrm{SL}_{\sigma\{f\}}\,\rangle$$

is defined as

$$\begin{aligned}
\Pi_{\sigma\{f\}} &= \Big\{\, \pi_{if(\mathit{a})} \;\Big|\; \pi_{i\mathit{a}} \in \Pi_\sigma \,\Big\} \\
\mathrm{SL}_{\sigma\{f\}} &= \Big\{\, \big\langle\, \boldsymbol{\pi}_i\big(f(\mathit{a})\big)\,,\, u_i\,,\, \boldsymbol{\pi}_j\big(f(\mathit{b})\big)\,,\, v_j \,\big\rangle \;\Big|\; \big\langle\, \boldsymbol{\pi}_i\big(\mathit{a}\big)\,,\, u_i\,,\, \boldsymbol{\pi}_j\big(\mathit{b}\big)\,,\, v_j \,\big\rangle \in \mathrm{SL}_\sigma \,\Big\}
\end{aligned}$$

(see Fig. 13).

Also, for a set of primitives Π_1, $\Pi_1 \subseteq \Pi$, we will use the notation $\Pi_1\{f\}$ to denote the corresponding set of f-relabeled primitives, where the *domain $\mathcal{L}$ of relabeling f is always assumed to be appropriately large*. Similar assumption about domain $\mathcal{L}$ will always be implicitly made when introducing any relabeling f without explicitly specifying its domain.

▶

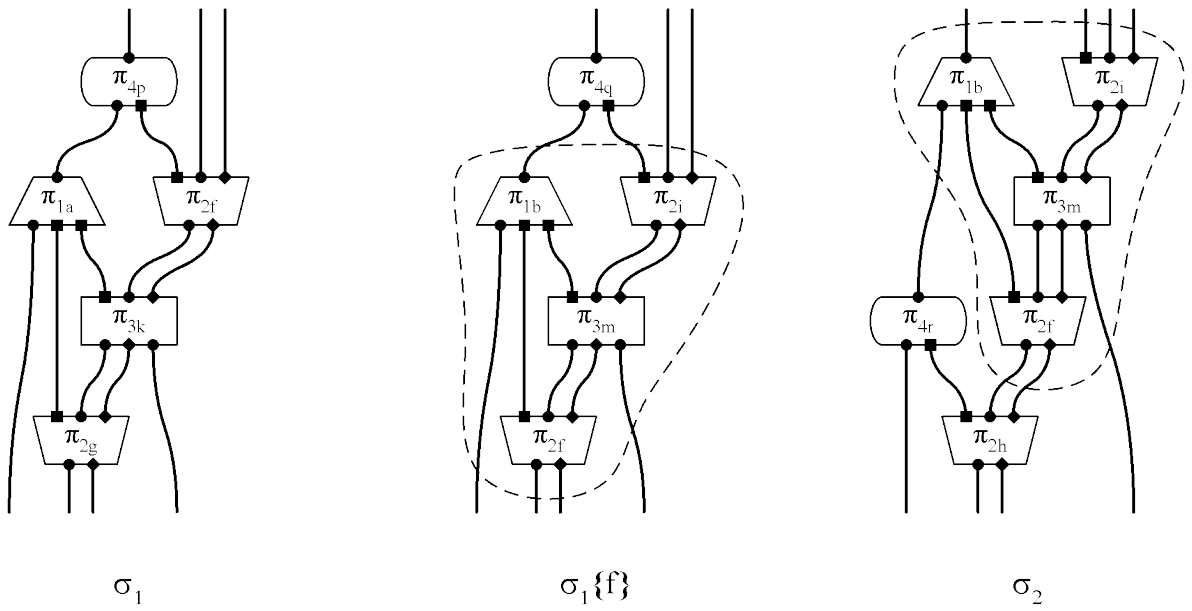


Figure 13: A struct σ_1 (left), a relabeled struct $\sigma_1\{f\}$ (center), and a third struct σ_2 (right). Relabeling reveals some structural similarity (dashed line) between $\sigma_1\{f\}$ and σ_2.

The following definition introduces the concept of a *structurally identical* class of structs, i.e. those structs which differ *only* in the (provisional) labelings of their primitives.

Definition 7. Two structs σ_1, σ_2 will be called **structurally identical**, denoted $\sigma_1 \sim \sigma_2$, if

$$\exists\, f \qquad\qquad \sigma_2 \;=\; \sigma_1\{f\}\,.$$

The corresponding equivalence class containing σ_1 will be denoted $[\![\sigma_1]\!]$ and is called an **abstract struct** (see also Appendix and caption to Fig. 1 in it). ▶

We now introduce the basic operation on structs, in which at least some of the involved structs must overlap. The latter condition is useful, for example, when putting together several observed overlapping structs.

Definition 8. Given several structs $\sigma_1, \sigma_2, \ldots, \sigma_r$, where $\sigma_i = \langle\, \Pi_{\sigma_i}, \mathrm{SL}_{\sigma_i} \,\rangle$, if the pair

$$\sigma \;=\; \Big\langle\; \bigcup_{i=1}^{r} \Pi_{\sigma_i}\;,\; \bigcup_{i=1}^{r} \mathrm{SL}_{\sigma_i} \;\Big\rangle$$

is a valid struct then this struct is called the **assembly of structs** $\sigma_1, \sigma_2, \ldots, \sigma_r$ and is denoted

$$\sigma \;=\; \mathcal{A}\big(\,\sigma_1, \sigma_2, \ldots, \sigma_r\,\big)\,.$$

▶

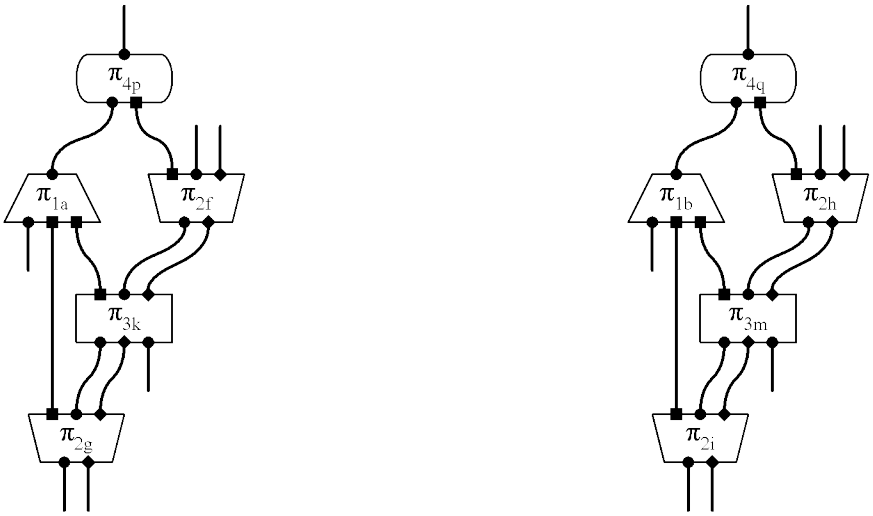


Figure 14: Two structurally identical structs.

We note that if several structs are not assemblable, it means that they either represent an inconsistent, contradictory, or simply non-overlapping views of the environment, i.e. it is not possible that all corresponding macroevents occur, or they form disconnected structs (or both). However, the absence of a link between two particular concrete primitives in a struct does not preclude its assemblability with another struct that contains a link between the same two primitives.

Moreover, although the legality and result of the assembly of several structs are affected by the particular labeling of their primitives—i.e. a relabeling of one of the structs may change the *legality* and/or *result* of their assembly (see Figs. 15 and 16)—*in practice*, any two observed structs either do or do not share some primitives, which also resolves the issue of their assemblability.

Similar to the above situation with attaching two separately-observed primitives (see p. 29), two separately-observed structs *with no overlap* (i.e. with no primitives in common) cannot reliably be "connected" to each other. The only way to "connect" such structs is via a chain of pairwise overlapping structs.

In general, one should keep in mind that if the representations of two processes share events, i.e. their structs overlap, this indicates that the processes themselves are *interacting*.

It is not difficult to see that the following properties hold:

(i) for a struct σ and relabelings f of struct σ and g of struct $\sigma\{f\}$, we have

$$\big(\sigma\{f\}\big)\{g\} \;=\; \sigma\{g \circ f\}$$

(ii) for structs σ and γ, if $\sigma = \gamma\{f\}$, then there exists the **inverse relabeling** f^{-1} of struct σ such that $\sigma\{f^{-1}\} = \gamma$

(iii) if struct $\sigma = \mathcal{A}(\alpha, \beta)$ and f is a relabeling of σ, then

$$\mathcal{A}(\alpha\{f\}, \beta\{f\}) \;=\; \big(\mathcal{A}(\alpha, \beta)\big)\{f\}\,.$$

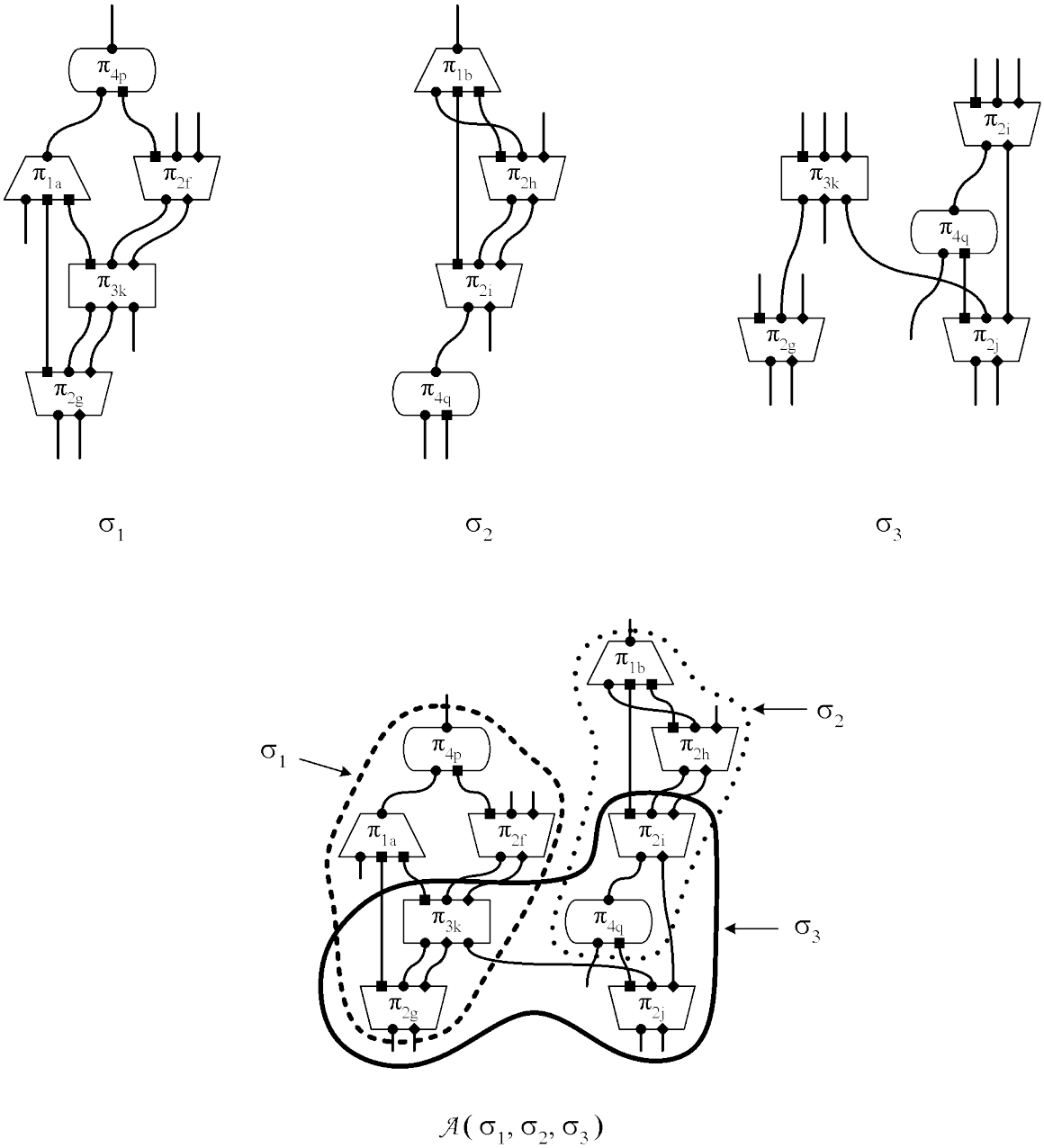


Figure 15: Three structs (top row) and their assembly. Note that the second link connecting π_{3k} and π_{2g} in the assembly comes from σ_1 (but not from σ_3, *despite the delineation of* σ_3 *with a bold line*). Also note that assembly $\mathcal{A}(\sigma_1, \sigma_2)$ is not legal, i.e. does not exist.

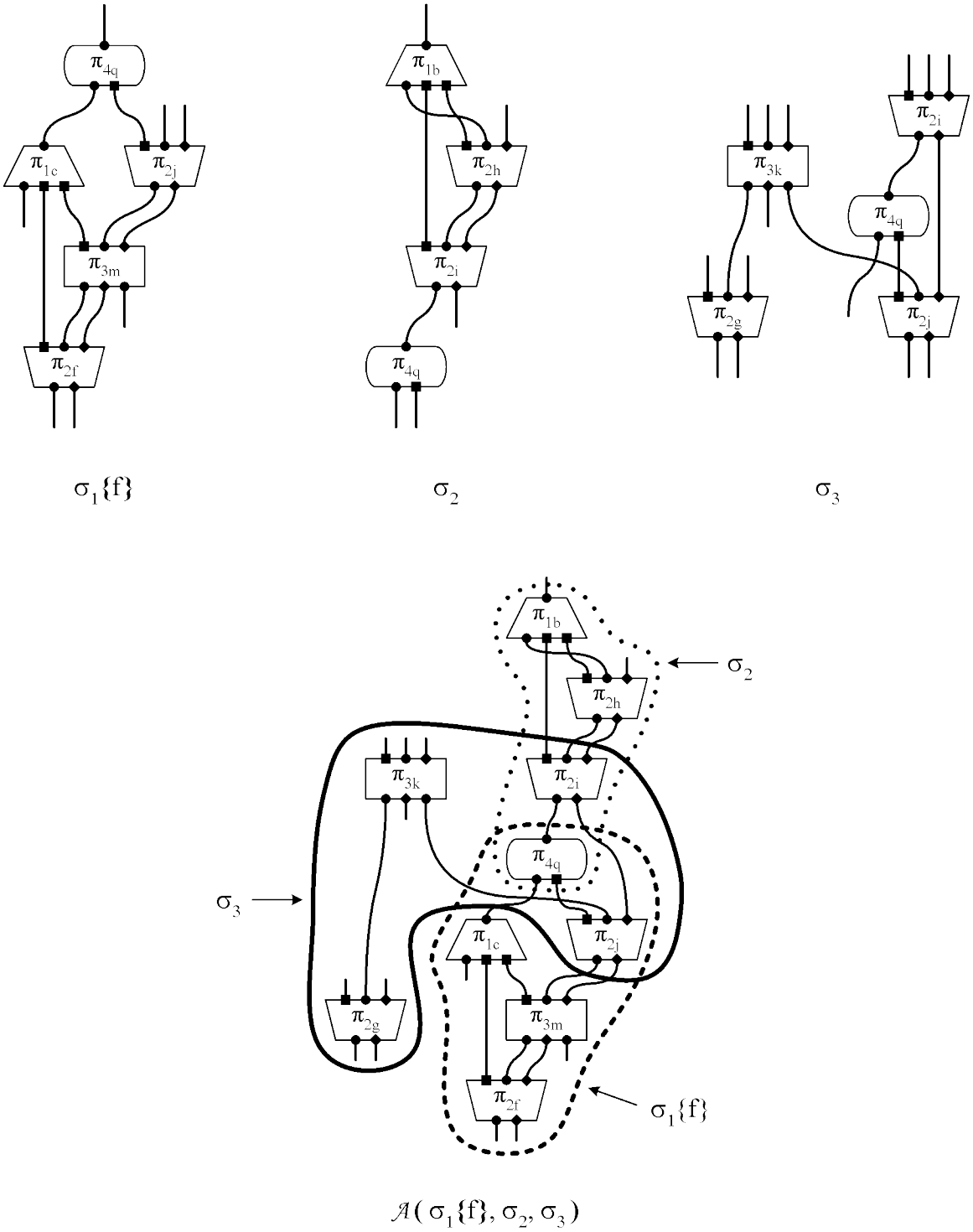


Figure 16: The three structs from Figure 15, the first one of which is *relabeled*, and their assembly.

Part III
Classes of structural entities

> The Universe does not consist of ready, finished objects, but instead represents a collection of processes in which objects continuously appear, change, and are destroyed. Nevertheless, from this it does not follow that they [objects] do not have a definite form of existence, that they are unstable, or that they are indistinguishable among themselves. However much an object changes, up to a certain point, it remains particularly *that*—and not any other—qualitatively definite object. ...
>
> *Quality* is the essential definiteness of an object, due to which it is, first, *that* object and not any other, and second, that it is different from other objects. The quality of an object, as a rule, is not reducible to its individual properties; rather it is connected with the object as a whole, captures it fully, and is from it.[a] ... Together with qualitative definiteness, all objects also possess quantitative definiteness: definite size, number, volume, ...
>
> *Quantity* is that definiteness of an object due to which (in reality or in thought) it could be subdivided into homogeneous parts[b] that are then agglomerated. Homogeneity (the resemblance, similarity) of parts of objects is the distinguishing feature of quantity.
>
> *The distinctions between objects that are not similar to each other carry qualitative character [i.e. they belong to different classes], while distinctions among similar objects carry quantitative character.* ... The exceptionally broad applicability of mathematical theories in the different domains of natural sciences and engineering can be explained by the fact that mathematics studies mainly quantitative relationships. Quality cannot be reduced to quantity, as metaphysicists attempt.
>
> Entry on "Quality and Quantity" in *Philosophical Dictionary*, ed. I. T. Frolov, 5th edition, Moscow, 1987 (our translation from Russian and our emphasis)

[a] Note that, in ETS, it is the concept of class representation that captures this concept of quality.

[b] See also Fig. 12.

In this part, we propose a radically different view of the Universe as formed solely of *interacting and evolving classes*. We must reemphasize that the material presented in this part should be viewed as being an early formalization of the principal ETS concepts, centered around the concept of class.

We briefly address the nature of *classes of structural processes*, which also clarifies the nature of primal classes (Def. 1). As was mentioned in the Introduction, the concept of class outlined here is much closer to that emerging from the recent research in evolutionary developmental biology[29], in which such processes as those implemented by regulatory (Hox) genes [68] play a central role, in contrast to processes more popular in classical biology and pattern recognition (i.e. in contrast to feature-based classes): "The embryo does not contain a description of the animal to which it will give rise, rather it contains a generative program for making it. ... There are thus no genes for "arm" or "leg" as such, but specific genes which become active during their formation." [56, pp. 199–200]

[29] Three popular references are [56], [54], and [55], and two of the standard ones are [67] and [68].

Moreover, the entire embryo development process (starting from a single fertilized egg cell) could serve as a suggestive example of a physical embodiment of the class generating process postulated below.

The concept of representational level is directly associated with those of *class, and hence object*, representations. Such *levels* should not be confused with the *stages* introduced in the next Part, which are associated with still-larger jumps in the "resolution" of representation (involving the substitution of observed macro-transformations by next-stage primitives).

This part concludes with a detailed illustrative example.

5 Single-level class representations

In this section, we first introduce the simplest concept of class, i.e. of a class whose elements are "regularly" composed out of some subset of primitives.

In what follows, the concept of struct introduced in Def. 3 will sometimes be referred to as that of *level 0 struct*, since we are about to embark on the construction of structs of *various levels*.

To proceed to the definition of a single level class representation, we need to introduce the following four definitions, upon which the former relies. They properly belong to Part II but are placed here for convenience of reference, since the central definition of this section relies heavily on them.

In Def. 10, we present one *tentative* (and in this paper the only) way of specifying a family of structs sharing a particular structural "backbone" [30]. We intend to use such structural descriptions to specify *restrictions* on a set of (local) substructs admissible as structural modules *in a single step* of a systematic process for constructing a *class* of "similar" structs.

Definition 10 will rely on the following definition, which introduces the concept of a family of structs, each of which could be thought of as realizing an admissible structural unit joining *two fixed concrete primitives*. Such family could also be thought of as extending the concept of link between two primitives.

Definition 9. Let π_{1a} and π_{2b} be some primitives, called **pivot primitives**, $\mathbf{\Pi}^*$ be a subset of the set of abstract primitives, and FR be the fixed **formation rules** for specifying various admissible sets Π^*'s consisting of concrete primitives from $\mathbf{\Pi}^*$ [31]. The $\mathbf{\Pi}^*$–**unit-constraint**, or simply **unit-constraint**, $\mathrm{UCon}\langle\, \pi_{1a}\,, u_1 \,\rangle\langle\, \pi_{2b}\,, v_2 \,\rangle\,(\mathbf{\Pi}^*, \mathrm{FR})$ **between** $\langle\, \pi_{1a}\,, u_1 \,\rangle$ **and** $\langle\, \pi_{2b}\,, v_2 \,\rangle$ (in that order), where u_1 is the index of one of π_{1a}'s terminals and v_2 is the index of one of π_{2b}'s initials, is defined as a family of structs $\{\sigma = \langle\, \Pi_\sigma,\, \mathrm{SL}_\sigma \,\rangle\}$ satisfying the following conditions: [32]

for one of the above Π^*, $\Pi^* \cap \{\pi_{1a}, \pi_{2b}\} = \varnothing$, we have

- $\{\pi_{1a}, \pi_{2b}\} \subseteq \Pi_\sigma \subseteq \Pi^* \cup \{\pi_{1a}, \pi_{2b}\}$

[30] Think of a "structural formula".

[31] For example, one may want to allow Π^* to have up to three primitives from $\boldsymbol{\pi}_i$, exactly two from $\boldsymbol{\pi}_j$, etc. In general, the formation rules are application-specific.

[32] See Defs. 2 and 3.

- $\exists \langle \pi_{ic}, v_i \rangle$, $\pi_{ic} \in \Pi^* \cup \{\pi_{2b}\}$, such that $\langle \pi_{1a}, u_1, \pi_{ic}, v_i \rangle \in \mathrm{SL}_\sigma$
- $\exists \langle \pi_{jd}, u_i \rangle$, $\pi_{jd} \in \Pi^* \cup \{\pi_{1a}\}$, such that $\langle \pi_{jd}, u_j, \pi_{2b}, v_2 \rangle \in \mathrm{SL}_\sigma$
- in ATTACH_σ, π_{1a} has exactly one child and is an ancestor of *all* π_{ie}, $\pi_{ie} \in \Pi_\sigma \setminus \{\pi_{1a}\}$
- in ATTACH_σ, π_{2b} has exactly one parent and is a descendant of *all* π_{jf}, $\pi_{jf} \in \Pi_\sigma \setminus \{\pi_{2b}\}$

Moreover, $\mathbf{\Pi}^*$ is such that there exists Π^* for which there exists at least one struct σ satisfying the above conditions. For any struct from the above family, we say that such a **struct satisfies unit-constraint** $\mathrm{UCon}\langle \pi_{1a}, u_1 \rangle \langle \pi_{2b}, v_2 \rangle (\mathbf{\Pi}^*, \mathrm{FR})$. ▶

Note that, in general, $\mathbf{\Pi}^*$ could be empty. Also note that in the case when $\underline{class}(\boldsymbol{\pi}_{1a}, u_1) = \overline{class}(\boldsymbol{\pi}_{2b}, v_2)$ [33], $\mathrm{UCon}\langle \pi_{1a}, u_1 \rangle \langle \pi_{2b}, v_2 \rangle (\mathbf{\Pi}^*, \mathrm{FR})$ always contains the struct whose only primitives are π_{1a} and π_{2b}, even when $\mathbf{\Pi}^* = \varnothing$ (see the second and third bullets in the definition; also see Fig. 17).

As will become clear later in this section, it is convenient to think of $\mathbf{\Pi}^*$ primitives as "non-primary", i.e., as being contributed by the "environment", including "noise" primitives. Thus, on the one hand, possible environmental variations are being taken into consideration, since the basic concept of a link between two primitives is now extended to allow for environmental influences. On the other hand, two structs that differ in substructs, both of which satisfy the unit-constraint, could be treated as equivalent with respect to this unit-constraint.

Definition 10. A (structural, level 0) **constraint** $\mathrm{Con}(\overline{\Pi}, \mathcal{UT})$ —involving a set of **pivot primitives** $\overline{\Pi}$ and a tuple of sets of primitives $\mathcal{UT}$ —is defined as a tuple of unit-constraints

$$\mathrm{Con}(\overline{\Pi}, \mathcal{UT}) = \left\langle \mathrm{UCon}\langle \pi_{1a}, u_1 \rangle \langle \pi_{2b}, v_2 \rangle (\mathbf{\Pi}^*_1, \mathrm{FR}_1), \ldots, \mathrm{UCon}\langle \pi_{2k-1,c}, u_{2k-1} \rangle \langle \pi_{2k,d}, v_{2k} \rangle (\mathbf{\Pi}^*_k, \mathrm{FR}_k) \right\rangle,$$

such that

- $\forall i, j \quad i \neq j \quad \langle \pi_{ie}, w_i \rangle \neq \langle \pi_{jg}, w_j \rangle$
- $\overline{\Pi} = \{\pi_{1a}, \pi_{2b}, \ldots, \pi_{2k-1,c}, \pi_{2k,d}\}$
- $\mathcal{UT} = \left\langle \langle \mathbf{\Pi}^*_1, \mathrm{FR}_1 \rangle, \ldots, \langle \mathbf{\Pi}^*_k, \mathrm{FR}_k \rangle \right\rangle$
- the graph, whose vertices correspond to pivot primitives and edges to unit-constraints, is connected.

The tuple of unit-constraints must also satisfy the following condition: there exists struct α, $\alpha = \langle \Pi_\alpha, \mathrm{SL}_\alpha \rangle$, and its relabeling f, such that

for *each* unit-constraint $\mathrm{UCon}\langle \pi_{2i-1,e}, u_{2i-1} \rangle \langle \pi_{2i,g}, v_{2i} \rangle (\mathbf{\Pi}^*_i, \mathrm{FR}_i)$, i = 1, . . . , k, there exists a struct $\sigma_i = \langle \Pi_{\sigma_i}, \mathrm{SL}_{\sigma_i} \rangle$ satisfying this unit-constraint and

[33] See notational convention 2.

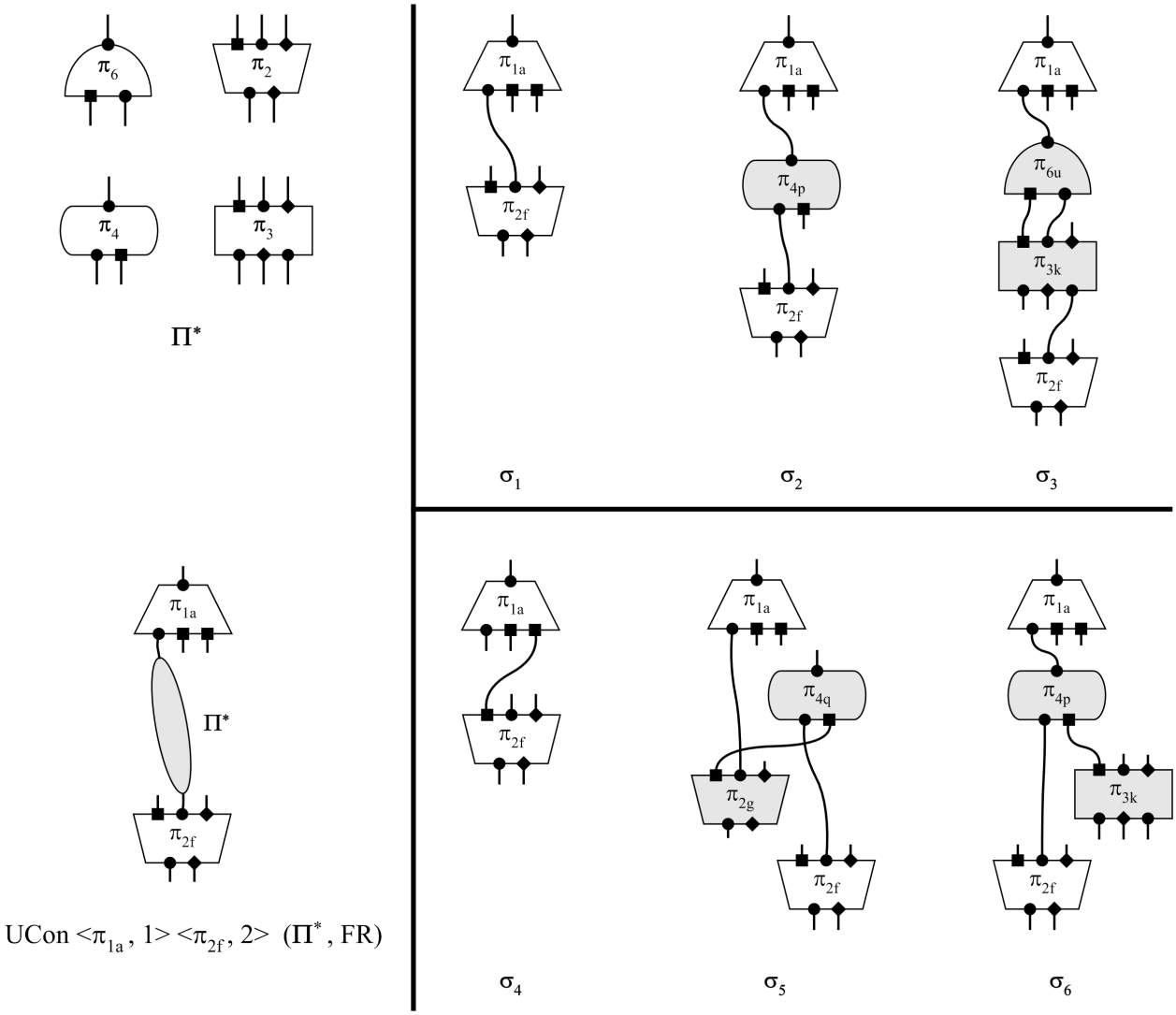


Figure 17: A unit-constraint $\mathrm{UCon}\langle\,\pi_{1a}\,,1\,\rangle\langle\,\pi_{2b}\,,2\,\rangle\,(\mathbf{\Pi}^*,\mathrm{FR})$ and two sets of structs, satisfying $(\,\sigma_1\,,\sigma_2\,,\sigma_3\,)$ as well as not satisfying $(\,\sigma_4\,,\sigma_5\,,\sigma_6\,)$ this constraint: for example, in σ_5, π_{4q} is not a descendant of π_{1a} and in σ_6, π_{3k} is not an ancestor of π_{2f}. An example of the rules FR is: at most two occurrences of $\boldsymbol{\pi}_2$, at most five occurrences of $\boldsymbol{\pi}_3$, one occurrence of $\boldsymbol{\pi}_4$, and at most three occurrences of $\boldsymbol{\pi}_6$. At bottom-left is a pictorial representation of the unit-constraint with pivot primitives π_{1a} and π_{2f}, where the light gray ellipse connecting them stands for a corresponding admissible substruct. On the right, non-pivot primitives are shown shaded light gray.

- $\Pi_{\alpha\{f\}} \;=\; \bigcup\limits_i \Pi_{\sigma_i}$
- $\bigcup\limits_i \mathrm{SL}_{\sigma_i} \;\subseteq \mathrm{SL}_{\alpha\{f\}}$.

We say that **struct** α **satisfies constraint** $\mathrm{Con}\,(\overline{\Pi},\,\mathcal{UT})$, see Fig. 19. The set of primitives in α that are relabeled primitives in $\overline{\Pi}$ is also called **the set of pivot primitives of** α, and denoted $\overline{\Pi}_\alpha$. In the case when the above tuple defining $\mathrm{Con}\,(\overline{\Pi},\,\mathcal{UT})$ is the null tuple (its length is 0), we call the constraint the **null constraint** and denote it Θ. ▶

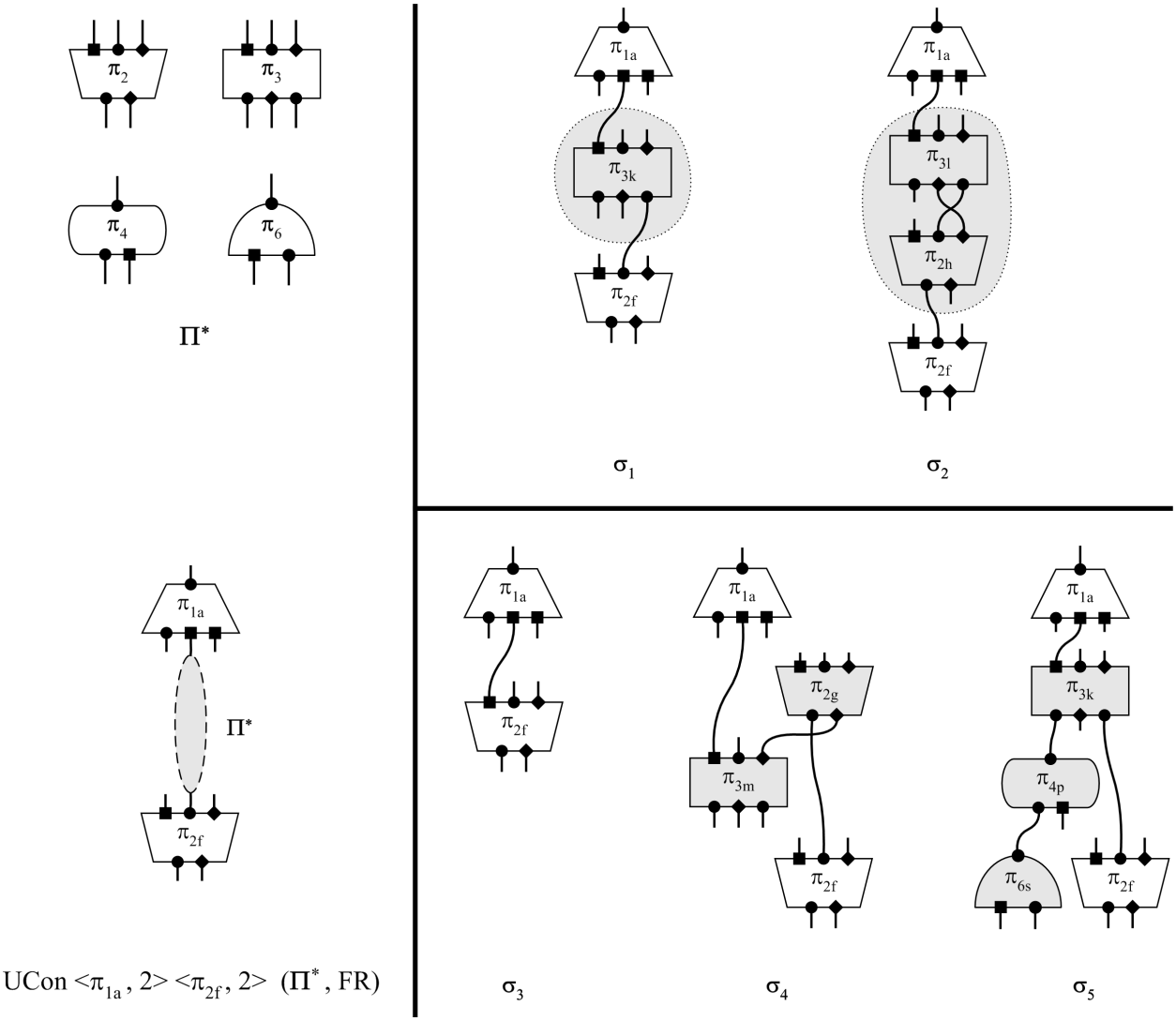


Figure 18: Another example of a unit-constraint, with only the top two structs σ_1 and σ_2 satisfying it.

Thus the pivot primitives ($\overline{\Pi}$) and the unit-constraints involved specify the *necessary* structural components in every struct α satisfying constraint $\mathrm{Con}\,(\overline{\Pi},\, \mathcal{UT})$, while the primitives in $\mathbf{\Pi}_i^*$ are *incidental* events that *may* accompany the necessary component of the constraint in α, allowing for a reasonable degree of variation. As was mentioned above, such incidental events may include "environmental" and/or noise events.

Definition 11. For a constraint $\mathrm{Con}\,(\overline{\Pi},\, \mathcal{UT})$, a corresponding **active constraint** is defined as the following triple

$$\mathrm{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\mathrm{anc}},\, \overline{\Pi}^{\mathrm{opn}}\,) \;\;=\;\; \big\langle\, \mathrm{Con}\,(\overline{\Pi},\, \mathcal{UT})\,,\, \overline{\Pi}^{\mathrm{anc}},\, \overline{\Pi}^{\mathrm{opn}}\,\big\rangle,$$

where

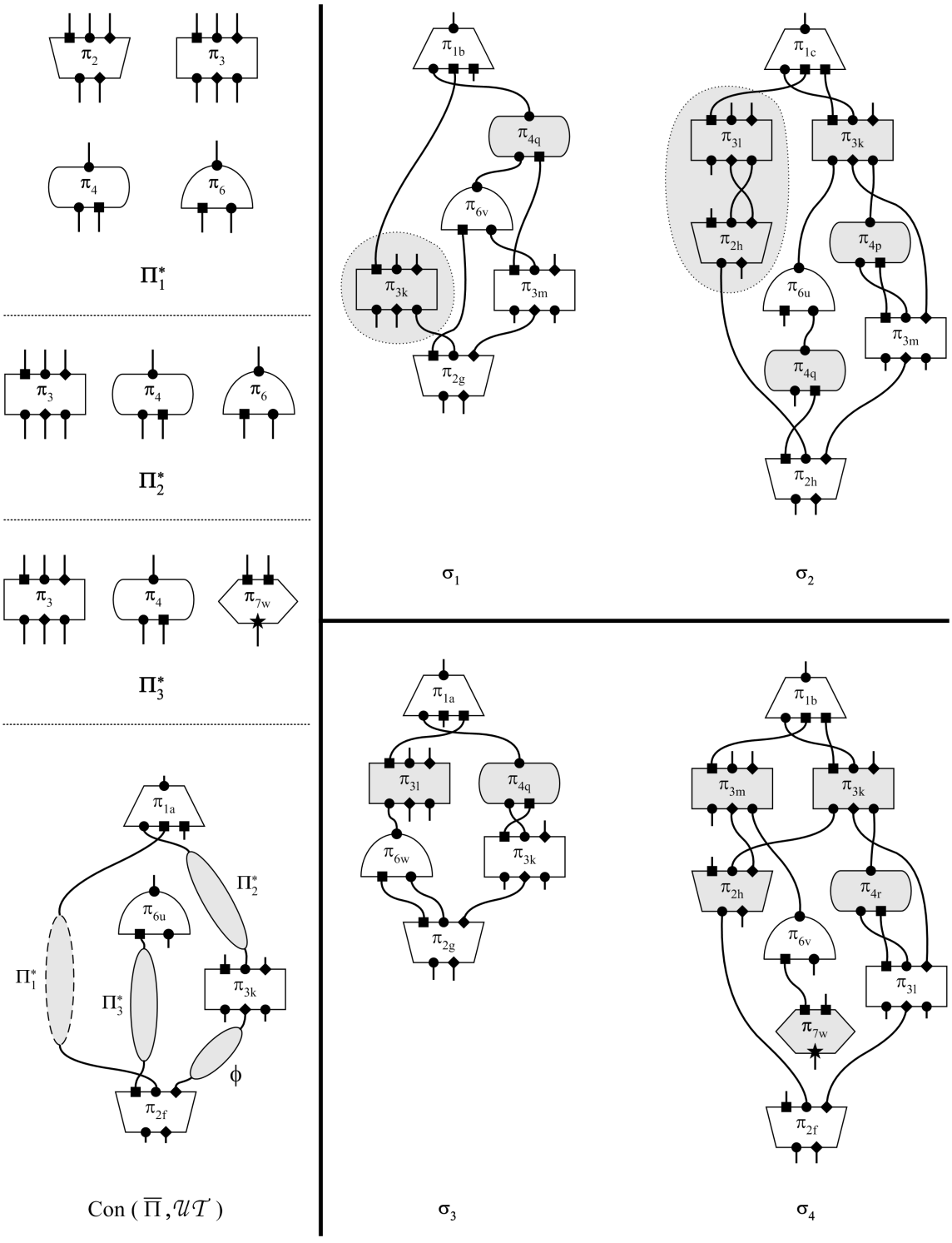


Figure 19: An example of a structural constraint $\mathrm{Con}\,(\overline{\Pi},\, \mathcal{UT})$, with $\overline{\Pi} = \{\, \pi_{1a}\,,\, \pi_{2f}\,,\, \pi_{3k}\,,\, \pi_{6u}\,\}$ and Π_i^*'s as shown on the top left; FR_i's are not included. On the right are two sets of structs: σ_1 and σ_2 satisfy the constraint (the two light-shaded ellipses correspond to the dashed ellipse at bottom left, which is part of the unit-constraint shown in Fig. 18) , while σ_3 and σ_4 do not: in σ_3 , π_{3l} is attached to the wrong terminal of π_{1a} , and in σ_4 , pivot primitive π_{6v} is not an ancestor of pivot primitive π_{2f} , according to the corresponding unit-constraint shown on the left.

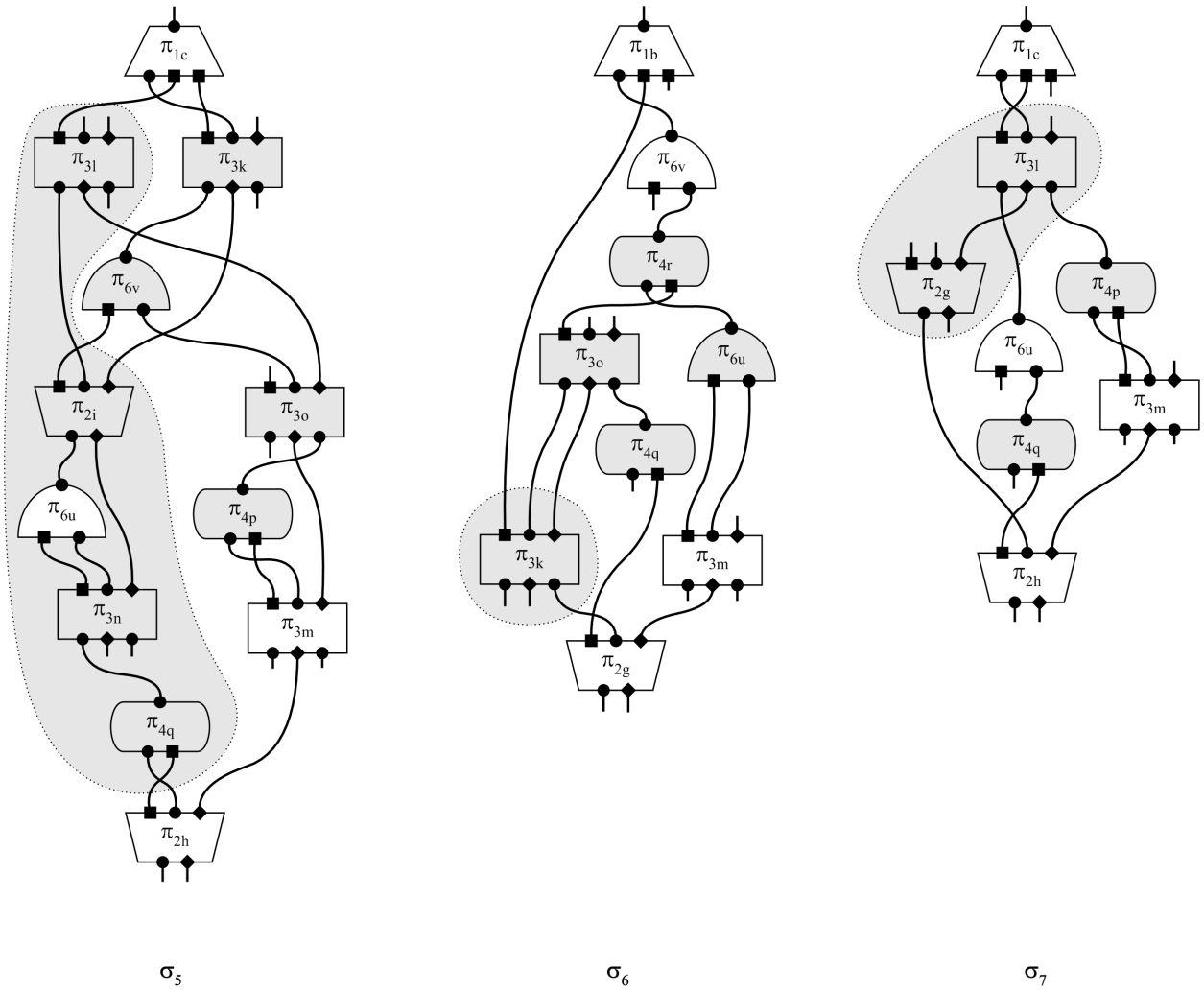


Figure 20: Three more structs satisfying the constraint shown in Fig. 19.

- the set of **anchor primitives**, denoted $\overline{\Pi}^{\text{anc}}$, is a *non-empty* proper subset of $\overline{\Pi}$ ($\overline{\Pi}^{\text{anc}} \subset \overline{\Pi}$)
- the set of **open primitives**, denoted $\overline{\Pi}^{\text{opn}}$, is a subset of $\overline{\Pi}$.

A **struct** α **satisfying an active constraint**, or simply **active struct**, is a struct that satisfies the corresponding constraint $\text{Con}(\overline{\Pi}, \mathcal{UT})$ and its *pivot* primitives inherit (under relabeling) all anchor and open markings from the active constraint $\text{ACon}(\overline{\Pi}, \mathcal{UT}, \overline{\Pi}^{\text{anc}}, \overline{\Pi}^{\text{opn}})$. The **sets of anchor and open primitives in** α are denoted $\overline{\Pi}_\alpha^{\text{anc}}$ ($\overline{\Pi}_\alpha^{\text{anc}} \neq \varnothing$) and $\overline{\Pi}_\alpha^{\text{opn}}$, respectively.

In the case when $\text{Con}(\overline{\Pi}, \mathcal{UT}) = \Theta$, we call the active constraint the **null active constraint** and denote it Θ also. ▶

The role of open primitives will be clarified in the next (important) definition, which *restricts* the family of active structs satisfying constraint $\text{ACon}(\overline{\Pi}, \mathcal{UT}, \overline{\Pi}^{\text{anc}}, \overline{\Pi}^{\text{opn}})$ to a particular subfamily.

Definition 12. A **working struct** σ^w is a struct some of whose primitives, subset $\overline{\Pi}^{\text{opn}}_{\sigma^w}$, $\overline{\Pi}^{\text{opn}}_{\sigma^w} \subseteq \Pi_{\sigma^w}$, are marked as open [34]. For a non-null active constraint $\text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)$ [35], the **set** $\text{Ext}\,\big(\sigma^w,\, \text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)\,\big)$ **of active extensions of working struct** σ^w **with respect to the active constraint** is defined as the set of active structs α satisfying the following conditions (see Fig. 21):

(i) α satisfies $\text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)$

(ii) assembly $\mathcal{A}(\sigma^w, \alpha)$ exists

(iii) $\overline{\Pi}^{\text{anc}}_{\alpha} \subseteq \overline{\Pi}^{\text{opn}}_{\sigma^w}$ [36]

(iv) $\overline{\Pi}_{\alpha} \setminus \Pi_{\sigma^w} \neq \varnothing$

In this case, the resulting assembly $\mathcal{A}(\sigma^w, \alpha)$, also denoted $\sigma^w \underset{\text{Ext}}{\triangleleft} \alpha$, is defined to be a working struct with open primitives specified as follows: they are the primitives in $\big(\,\overline{\Pi}^{\text{opn}}_{\sigma^w} \setminus \overline{\Pi}_{\alpha}\,\big) \cup \overline{\Pi}^{\text{opn}}_{\alpha}$.

For the null active constraint, we define the set $\text{Ext}\,\big(\sigma,\, \text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)\,\big)$ to be $\{\,\theta\,\}$. ▶

Note that since $\overline{\Pi}^{\text{anc}}_{\alpha} \neq \varnothing$, (iii) ensures that α and σ^w share at least one pivot primitive. Condition (iv) ensures that $\alpha \notin \sigma^w$. Moreover, even if $\text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,) \neq \Theta$, set $\text{Ext}\,\big(\sigma,\, \text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)\,\big)$ could still be $\varnothing$.

The anchor primitives in α specify structural "anchors" on which the two structs α and σ^w must overlap. The open primitives in σ^w specify all of the *allowable* "interface points", where an active extension is allowed to overlap with σ^w. Hence, for α to be a non-null active extension of σ^w, *all* anchor primitives in α must be open in σ^w.

We are now ready to introduce the central concept of this section.

Definition 13. A **single-level class representation** $\mathfrak{R}$ is a triple

$$\mathfrak{R} \;=\; \big\langle\, \overline{\mathbf{\Pi}}_{\mathfrak{R}},\, \mathbf{\Pi}^{*}_{\mathfrak{R}},\, \mathscr{G}_{\mathfrak{R}} \,\big\rangle,$$

where $\overline{\mathbf{\Pi}}_{\mathfrak{R}}$ is a set of **constituent pivot** abstract **primitives** (for the class), $\mathbf{\Pi}^{*}_{\mathfrak{R}}$ is a set of **constituent non-pivot** abstract **primitives** (for the class), and $\mathscr{G}_{\mathfrak{R}}$ is a partial *stepwise* specification of a **(level 0) class generating system**:

$$\mathscr{G}_{\mathfrak{R}} = \Big\langle \big\{\, \text{ACon}_{1,i}(\,\overline{\Pi}_{1,i},\, \mathcal{UT}_{1,i},\, \overline{\Pi}^{\text{anc}}_{1,i},\, \overline{\Pi}^{\text{opn}}_{1,i}\,)\,\big\}_{i\in I_1},\;\; \big\{\text{ACon}_{2,i}(\,\overline{\Pi}_{2,i},\, \mathcal{UT}_{2,i},\, \overline{\Pi}^{\text{anc}}_{2,i},\, \overline{\Pi}^{\text{opn}}_{2,i}\,)\,\big\}_{i\in I_2},$$

[34] Note that "anchor" markings are not applicable in this case., while "open" markings **are always specific to a particular process generating a class element, see Def. 13.**

[35] Struct σ^w has no relation to the active constraint.

[36] Note that when $\overline{\Pi}^{\text{opn}}_{\sigma^w} = \varnothing$, this condition is violated (since by definition $\overline{\Pi}^{\text{anc}}_{\alpha} \neq \varnothing$) and hence $\text{Ext}\,\big(\sigma,\, \text{ACon}\,(\,\overline{\Pi},\, \mathcal{UT},\, \overline{\Pi}^{\text{anc}},\, \overline{\Pi}^{\text{opn}}\,)\,\big) = \varnothing$.

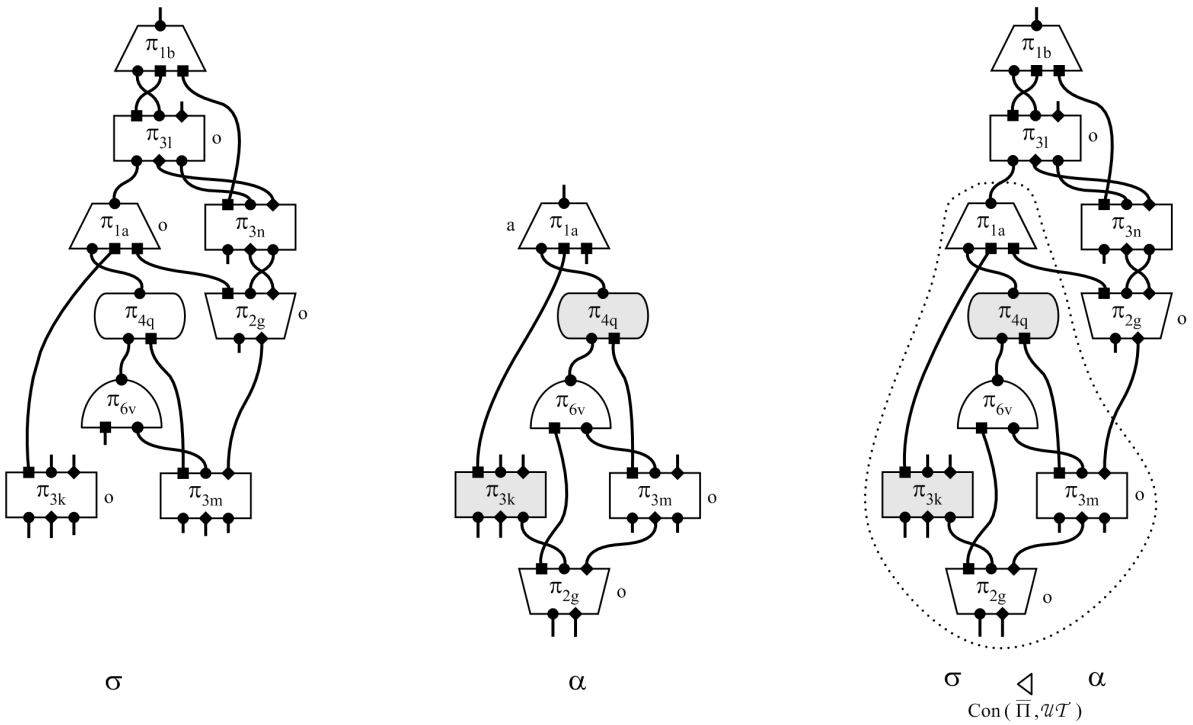


Figure 21: Active struct α satisfying the active constraint corresponding to the constraint given in Fig. 19. Struct α is also an active extension of working struct σ, and their assembly is shown on the right.

$$\ldots\,,\ \Big\{\mathrm{ACon}_{t,i}(\,\overline{\Pi}_{t,i},\ \mathcal{UT}_{t,i},\ \overline{\Pi}^{\mathrm{anc}}_{t,i},\ \overline{\Pi}^{\mathrm{opn}}_{t,i}\,)\Big\}_{i\in I_t}\Big\rangle\,,$$

where each $\pi \in \overline{\Pi}_{j,i}$ is taken from some $\boldsymbol{\pi}$, $\boldsymbol{\pi} \in \overline{\boldsymbol{\Pi}}_{\mathfrak{R}}$, each tuple $\mathcal{UT}_{i,j}$ is formed from subsets of $\boldsymbol{\Pi}^{*}_{\mathfrak{R}}$, and each of the above sets of active constraints specifies the corresponding step by $\mathscr{G}_{\mathfrak{R}}$. The *actions* of this generating system are described next.

Each step in the operation of the generating system $\mathscr{G}_{\mathfrak{R}}$ *follows* a "step" in the action of the **level 0 environment** $\mathfrak{E}$. Such environment could be thought of as comprised of a finite set of level 0 classes that can immediately interact with $\mathscr{G}_{\mathfrak{R}}$. Each such "step" by $\mathfrak{E}$ is specified by its own sets of active constraints. This interaction typically manifests itself in the environmental structs being attached to the current incomplete class element *in between* the steps of the generating system. Specifically, at step j, the system "responds" to the following j$^{\mathrm{th}}$ step by $\mathfrak{E}$ in a manner specified below:

- step j by $\mathfrak{E}$: the environment $\mathfrak{E}$ may assemble several structs (each taken from a corresponding class) to the *previous working struct* $\sigma_{2(j-1)}$ to produce the *current working struct* σ_{2j-1}, whose pivot primitive *markings are unchanged*; such step by $\mathfrak{E}$ could in fact be comprised of several "actual" steps (depending on the nature of the classes in $\mathfrak{E}$ participating in this step);

- step j by $\mathscr{G}_{\mathfrak{R}}$: in turn[37], denoting by γ_{j-1}, $\gamma_{j-1} \Subset \sigma_{2j-1}$, the $j-1$ **working class**

[37] The class generating system acts without reference to the structure of $\mathfrak{E}$, i.e. it does not know the structure of the classes that affected the working struct.

element, i.e., the substruct of the current working struct formed by the primitives that have been contributed so far by $\mathscr{G}_{\mathfrak{R}}$ only, the class generating system

- first nondeterministically/probabilistically chooses struct β_j from one of the non-empty sets of structs $\left\{\mathrm{Ext}\left(\gamma_{j-1},\, \mathrm{ACon}_{j,i}(\,\overline{\Pi}_{j,i},\, \mathcal{UT}_{j,i},\, \overline{\Pi}^{\mathrm{anc}}_{j,i},\, \overline{\Pi}^{\mathrm{opn}}_{j,i}\,)\,\right)\right\}_{i\in I_j}$ and also satisfying:
  - $\Pi_{\beta_j} \setminus \overline{\Pi}_{\beta_j} \subseteq \Pi_{\sigma_{2j-1}}$
  - $\mathcal{A}(\beta_j,\, \sigma_{2j-1})$ exists
- assembles β_j to σ_{2j-1} to produce the next working struct σ_{2j}; moreover, it is not difficult to see that performing the latter assembly also accomplishes the following assembly (since $\gamma_{j-1} \Subset \sigma_{2j-1}$) producing the next working class element $\gamma_j = \gamma_{j-1} \underset{\mathrm{Ext}}{\lhd} \beta_j$ (to appropriately allocate "open" markings). See Fig. 22.

Note that the initial step is a simplified version of the above generic step, in which the very first step by the environment produces not only (the current working) struct σ_1, but also its substruct, the initial working class element γ_0 with the appropriate markings.

The generating system $\mathscr{G}_{\mathfrak{R}}$ contains its own terminating condition (not presently specified) for completing struct γ, where $\gamma = \mathcal{A}(\beta_1, \beta_2, \ldots, \beta_s)$, $\gamma \Subset \sigma_{2s}$ (σ_{2s} is the final working struct). For any such struct γ, the pair

$$\mathfrak{c} \;=\; \langle\, \gamma\,,\, \mathscr{G}_{\mathfrak{R}}\,\rangle$$

is called a **class element** of the **single-level class** $\mathfrak{C}(\mathfrak{R}, \mathfrak{E})$—or simply $\mathfrak{C}$—**induced by** $\mathfrak{R}$ **in environment** $\mathfrak{E}$ (see Fig. 23). ▶

Note that the presence of a null constraint in the constraint set adds the option of leaving the working class element unchanged at the corresponding step. Moreover, for a given step, (in the case of a non-null constraint) the action of the generating system must lead to the extension of the working class element, but it does not have to extend the working struct: this is the case when *all* primitives added to the working class element at this step are already present in the working struct (because they were previously added by the environment).

Also note that some of the active constraints may account for various environmental contingencies, allowing generating system to respond sensibly to the environmental reality. Thus, the generating system outputs not only ideal class elements but also so-called "noisy" ones, i.e. it should be able to produce class elements of appropriate (for that class and its environment) structural variability, some being more typical than others. On the other hand, there are some situations which modify class elements in such a way that the *representation of the class changes*. The relevant ETS mechanism—a transformation—that addresses such changes of class representations (as well as the production of new classes) will be considered in Section 9.

We draw your attention to the following *synonymous terms*: single-level class (element) and level 0 class (element).

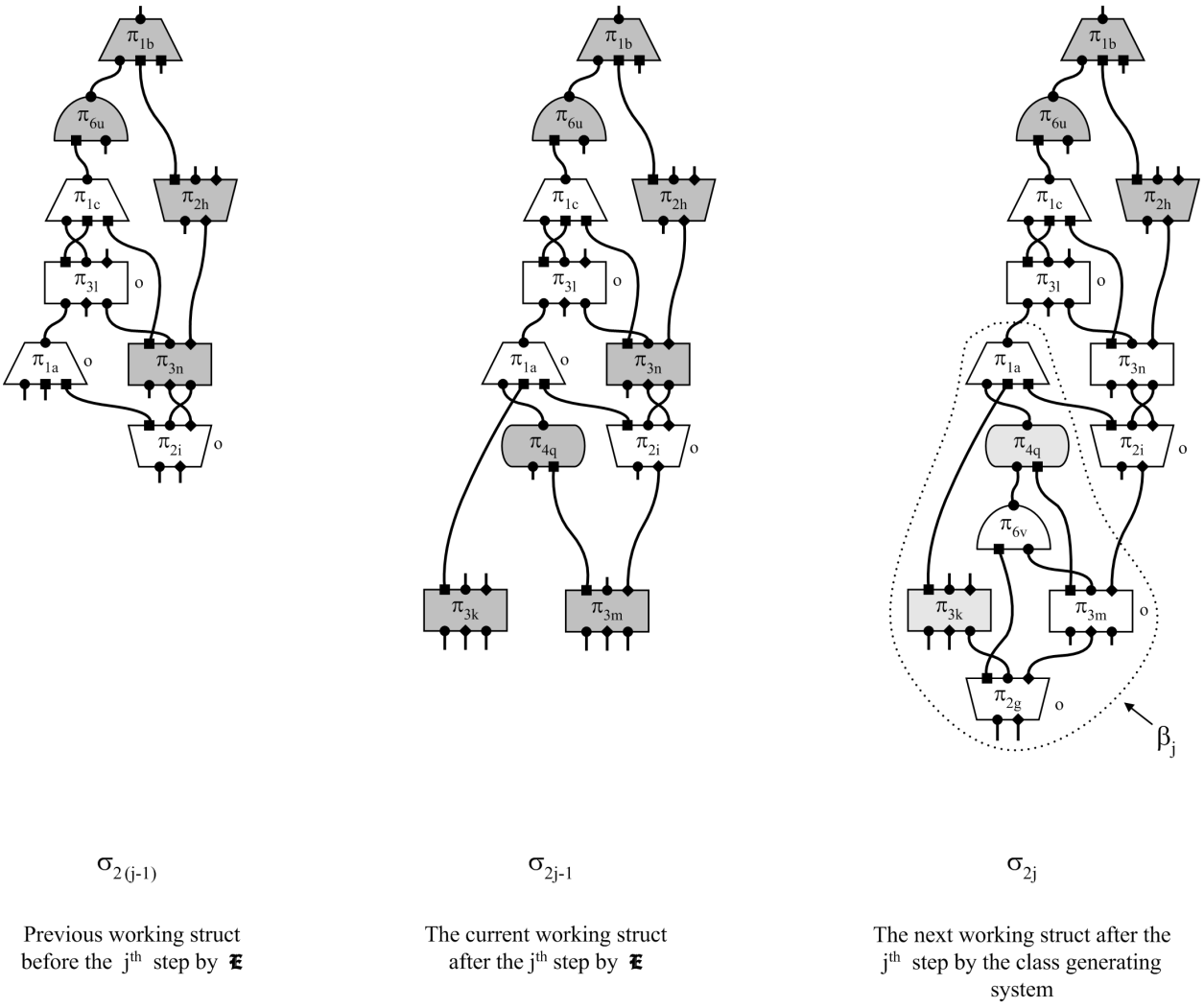


Figure 22: Pictorial representation of a two-step generative unit—a step by the environment $\mathfrak{E}$ and the corresponding step by the class generating system $\mathscr{G}_{\mathfrak{R}}$—in the construction of class element $\mathfrak{c}$, $\mathfrak{c} \in \mathfrak{C}$. Dark shaded primitives are those added by the environment, and the dotted line delineates the active struct assembled to σ_{2j+1} (and satisfying the corresponding active constraint). Primitives that remain dark shaded at the end of the generating process are not part of the class element $\mathfrak{c}$ that is being generated. Note that because primitives π_{3k}, π_{3m} in σ_{2j+1} happened to be in β_j, they *became* part of $\mathfrak{c}$ in σ_{2j+2}

The adjective "partial" qualifying the specification of a level 0 class generating system (in the first paragraph of the definition), suggests that this specification is, in a sense, incomplete: not only is the terminating condition implicit, but the above "actions" of the system are also not integrated into its specification.

We draw your attention to the fact that the class representation $\mathfrak{R}$—in contrast to its output, i.e. the associated class $\mathfrak{C}$—is independent[38] of the environment $\mathfrak{E}$.

Moreover, $\mathscr{G}_{\mathfrak{R}}$ can be thought of as being a semi-autonomous system "responsible for producing" class elements. Also note that, when a generating system terminates, it does

[38] Or, more accurately, *acts* independently.

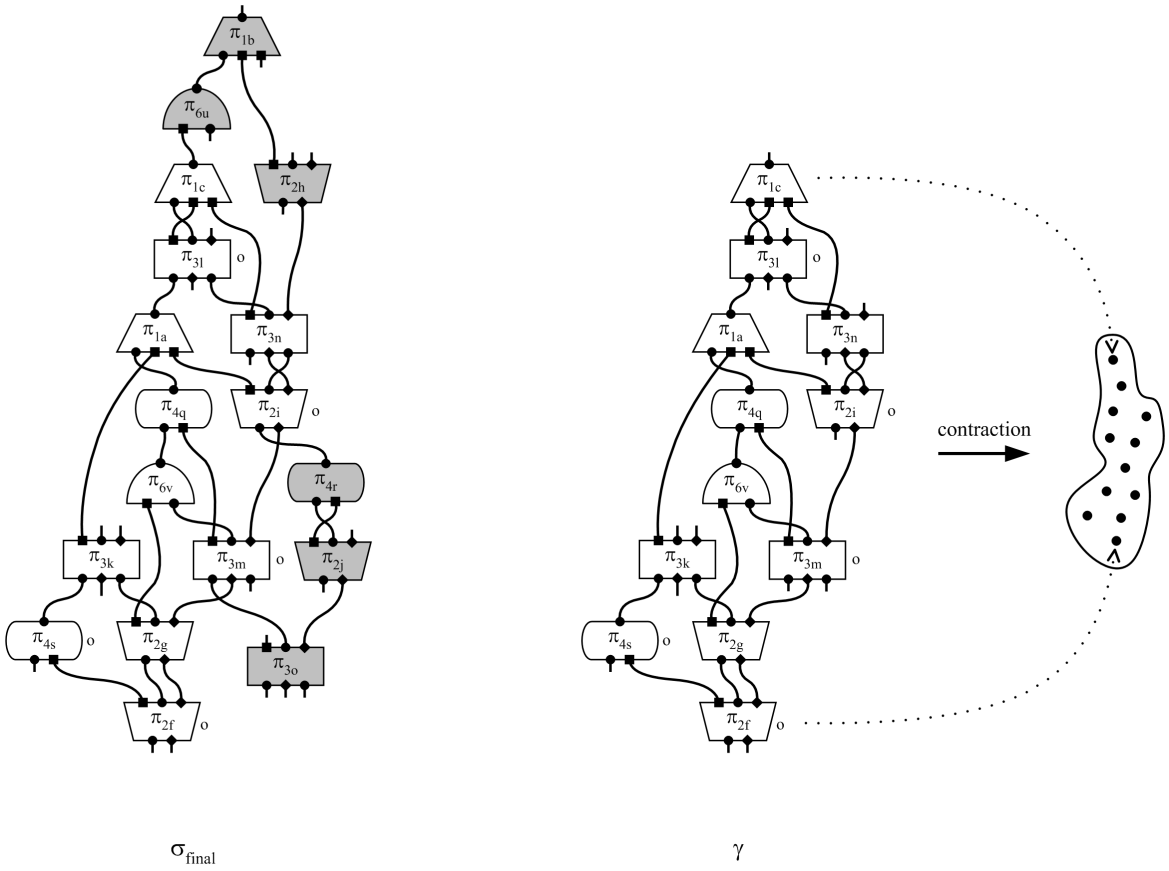


Figure 23: *Left*: Pictorial representation of a final output of the class generating system, the working struct σ_{final} (which is a "completion" of the last struct shown in Fig. 22). *Right*: The final working class element γ, which is a substruct of struct σ_{final}, where γ is the struct of the class element $\mathfrak{c}$, $\mathfrak{c} = \langle \gamma, \mathscr{G}_{\mathfrak{R}} \rangle$. On the far right, we show a *contracted* γ *as a (potential) constituent element of a level 1 struct* (Def. 15), a simplified depiction which will be used in the sequel: solid dots stand for contracted primitives and struct links are omitted.

not necessarily mean that the corresponding output process is fully "terminated", but it is possible that another, closely related, generating system takes over, e.g. a fetus versus a newborn infant.

An important difference between the ETS formalism and other formalisms (e.g. generative grammar models) is that *ETS is structured in such a way* as to allow for an inductive "bridging" of the two components of the pair defining $\mathfrak{c}$. In other words, it is expected that the temporal information embodied in the first component (the struct) allows for a *reliable* recovery of the second component, based on a small training set of structs.

Recalling our assumption about the disjointness of primal classes (before Def. 1), indeed any two classes are disjoint, since each class element carries within itself its class representation. A fundamental pragmatic implication of this is that the stored form of every class element *must include the corresponding class representation*[39], though this is not necessarily accessible to an external agent.

[39] A well-known example of class representation is that with which DNA is associated.

As to the importance of the role of the environment, it suffices to quote one of the leading developmental biologists, Scott Gilbert [68, p. 721]:

> [R]ecent studies have shown that the environmental context plays significant roles in the development of almost all species, and that animal and plant genomes have evolved to respond to environmental conditions.... Moreover, symbiotic associations, wherein the genes of one organism are regulated by the products of another organism, appear to be rule, rather than the exception.

How tentative is the above generating system specification? At present, one should approach that issue with due caution, in view of the lack of relevant experience. The main difficulty in addressing this issue is related to the lack of a precise language for describing such a generative mechanism in the structural setting outlined above. However, it appears that a hint in this direction may lie in the appropriate modification of the structural constraint concept introduced in Def. 10.

Important Remark 2. For a fixed level 0 class setting, each class generating system has a range of "states", having passed though which the system always remains in a "mature" state, and continues to produce mature class elements. Having passed this range of states, the struct γ within class element $\mathfrak{c}$ is supposed to carry almost all structural information associated with the class. Moreover, some class elements are "extensions" of other class elements—some immature, some mature—and one branch of the class generating system responsible for producing a complete (temporal) chain of class elements may be called a **class system instance**. In the Introduction (and also throughout the paper), when we speak of "structural processes", it is the latter concept that comes closest to describing such processes. Hence, it is quite reasonable to expect that, in the future, a class element will become a more dynamic entity, capturing the idea of a concrete structural process.

For an organism, one can view its embryonic stage as corresponding to an immature state of the class generating process. The adult stage of an organism corresponds to the mature part of a class system instance, i.e. when the organism/process becomes ready for interaction with other processes. ◗

Definition 14. We introduce a finite set of single-level classes

$$\mathcal{C} \;=\; \left\{ \mathfrak{C}_1 \,,\, \mathfrak{C}_2 \,,\, \ldots \,,\, \mathfrak{C}_{l_0} \right\},$$

l_0 is the number of single-level classes, which we call a **level 0 class setting**. ▶

The next definition introduces the concept of level 1 struct, as that of a *struct assembled from several level 0 class elements*.

Definition 15. Having fixed class setting $\mathcal{C}$, a **level 1 struct** σ^1 is defined as a pair

$$\sigma^1 \;=\; \langle\, \mathscr{C}_\sigma \,,\, \sigma \,\rangle \,,$$

where the **level 0 class-based representation of level 0 struct** σ

$$\mathscr{C}_\sigma \;=\; \left\{ \mathfrak{c}_1 \,,\, \mathfrak{c}_2 \,,\, \ldots \,,\, \mathfrak{c}_{s_0} \right\}$$

is defined in such a way that $\mathfrak{c}_i \in \mathfrak{C}_j$, $\mathfrak{c}_i = \langle\, \gamma_i \,,\, \mathscr{G}_{\mathfrak{R}_i} \,\rangle$, and $\sigma = \mathcal{A}\big(\, \gamma_1 \,,\, \gamma_2 \,,\, \ldots \,,\, \gamma_{s_0} \big)$. We refer to $\mathfrak{c}_i$ as a **constituent element of** σ^1 . See Figure 24.

The set of all level 1 structs will be denoted Σ^1, and when $\mathscr{C}_\sigma = \varnothing$, struct σ^1 will be called the **null level 1 struct**, denoted θ^1. ▶

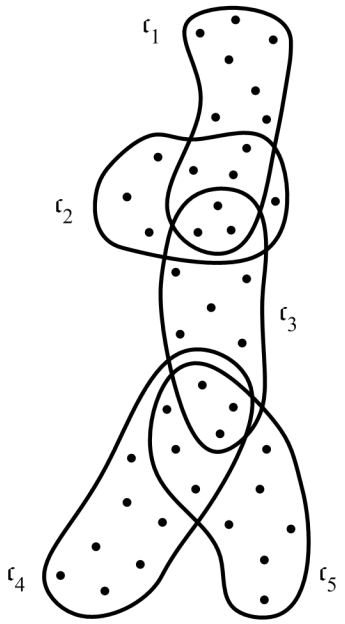


Figure 24: Simplified pictorial representation of a level 1 struct (using the contracted form of its constituent elements, as shown in in Fig. 23).

We now generalize the concept of substruct to level 1 structs.

Definition 16. For two level 1 structs $\alpha^1 = \langle\, \mathscr{C}_\alpha \,,\, \alpha \,\rangle$ and $\beta^1 = \langle\, \mathscr{C}_\beta \,,\, \beta \,\rangle$, we say that α^1 is a **substruct** of β^1, denoted[40] $\alpha^1 \Subset \beta^1$, if

$$\mathscr{C}_\alpha \subseteq \mathscr{C}_\beta \,.$$

▶

Next, we generalize the concept of struct assembly to level 1 structs.

[40] For simplicity, we suppress the level index on the substruct operator $\Subset^1$ since both operands carry the corresponding index and thus no ambiguity arises.

Definition 17. Assume we are given several level 1 structs $\sigma_1^1, \sigma_2^1, \ldots, \sigma_r^1$, where $\sigma_i^1 = \langle \mathscr{C}_{\sigma_i}, \sigma_i \rangle$. If the assembly $\mathcal{A}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ is a valid struct σ, then the pair

$$\sigma^1 \;=\; \Big\langle \bigcup_{i=1}^{r} \mathscr{C}_{\sigma_i}, \sigma \Big\rangle$$

is a level 1 struct called the **assembly of level 1 structs** $\sigma_1^1, \sigma_2^1, \ldots, \sigma_r^1$, denoted

$$\sigma^1 \;=\; \mathcal{A}(\sigma_1^1, \sigma_2^1, \ldots, \sigma_r^1).$$

▶

6 Two-level class representations

In this section, we make the next step towards a generalization of the concept of single-level class and introduce the concept of two-level class representation.

We will use the following notation for a finite set whose elements are *some* selected elements of various (not necessarily distinct) classes from level 0 class setting $\mathcal{C}$:

$$\mathscr{C} \;=\; \{\, \mathfrak{c}_i \mid \mathfrak{c}_i \in \mathfrak{C}_{k_i} \in \mathcal{C},\, 1 \leq i \leq r \,\}.$$

Definition 18. For a level 0 class setting $\mathcal{C}$, a **level 1 class element link**, $\mathrm{CEL}^1\big(\mathscr{C}, \mathrm{Con}(\overline{\Pi}, \mathcal{UT})\big)$, **between** (level 0 class) **elements in a set** $\mathscr{C}$ is defined as the pair

$$\mathrm{CEL}^1\big(\mathscr{C}, \mathrm{Con}(\overline{\Pi}, \mathcal{UT})\big) \;=\; \big\langle \mathscr{C}, \mathrm{Con}(\overline{\Pi}, \mathcal{UT}) \big\rangle,$$

where

(i) for $\mathfrak{c}_i \in \mathscr{C}$, $\mathfrak{c}_i = \langle \gamma_i, \mathscr{G}_{\mathfrak{R}_i} \rangle$, assembly $\sigma = \mathcal{A}(\gamma_1, \gamma_2, \ldots, \gamma_r)$ exists

(ii) for each γ_i there exists its substruct γ_i' such that

$$\gamma_i' \;=\; \langle \Pi_i', \mathrm{SL}_i' \rangle, \qquad \overline{\Pi} \subseteq \Pi_i'$$

and

$$\gamma_i' \text{ satisfies } \mathrm{Con}(\overline{\Pi}, \mathcal{UT}).$$

(See Figure 25.) ▶

In particular, the above elements $\mathfrak{c}_i$'s in $\mathscr{C}$ must share primitives from Π_1, while γ_i''s can only differ in non-pivot primitives from the constraint.

Note that the constraint $\mathrm{Con}(\overline{\Pi}, \mathcal{UT})$ need not have any particular relation to the constraints involved in the specification of the above classes $\mathfrak{C}_{k_i}$, $\mathfrak{C}_{k_i} \in \mathcal{C}$.

As was the case with the *primitives* in $\overline{\Pi}$ and $\mathcal{UT}$ involved in the definition of a level 0 constraint (Def. 10), the concept of a level 1 constraint (Def. 19) relies on two finite sets of *class elements* $\overline{\mathscr{C}}$ and $\mathscr{C}^*$. Moreover, one should note the overall analogy between these two definitions, with the two sets of class elements playing a role somewhat similar to that played by $\overline{\Pi}$ and $\bigcup \mathbf{\Pi}_i^*$.

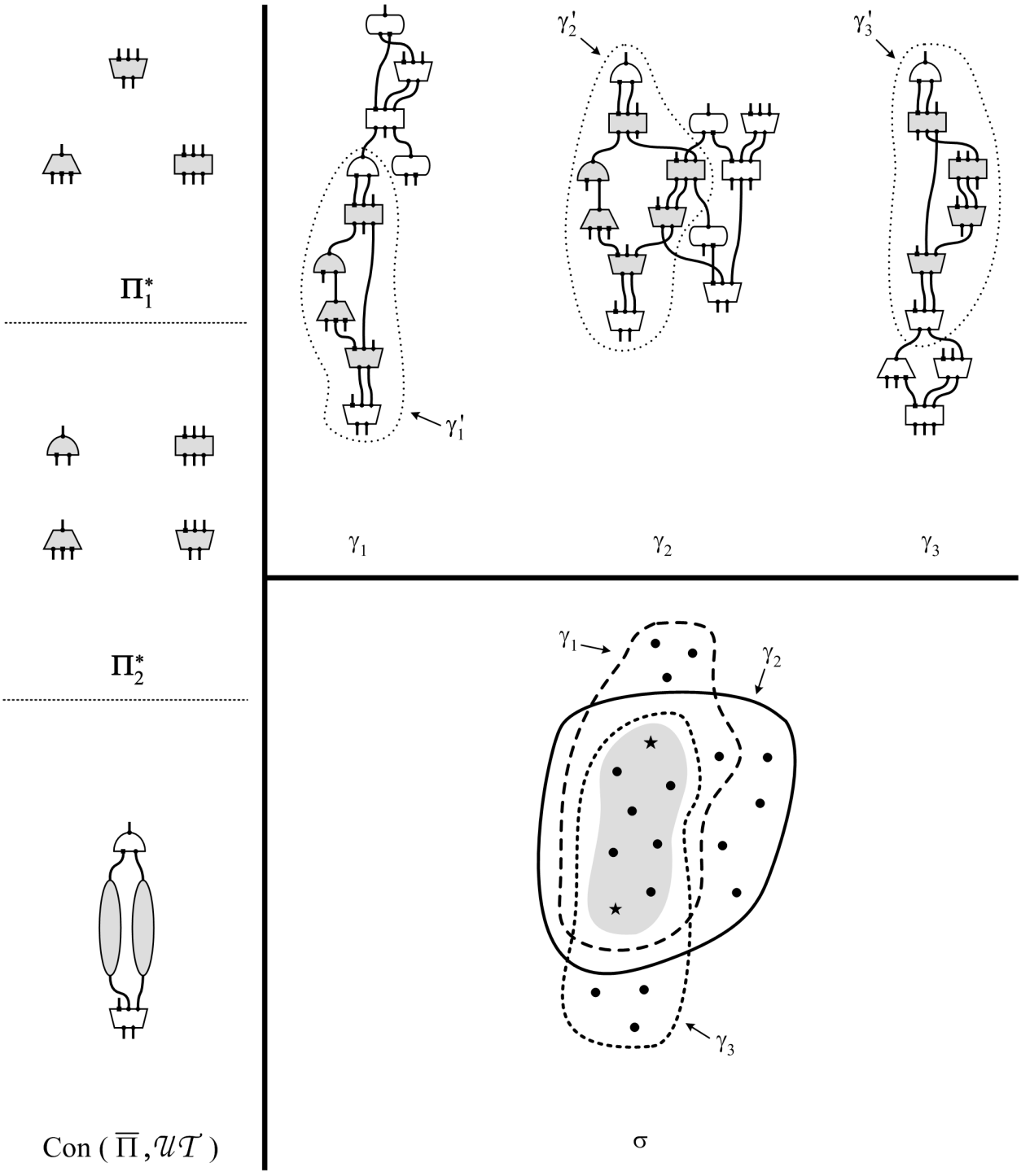


Figure 25: Illustration of a level 1 class element link. *Left*: depiction of a (level 0) constraint. *Top Right*: structs corresponding to three level 0 class elements (from $\mathscr{C}$), with the corresponding substructs γ_i^*'s satisfying constraint Con ($\overline{\Pi}$, $\mathcal{UT}$) (see Fig. 19) delineated with dotted lines; the primitives from Π_i^*'s are shaded. *Bottom Right*: the struct σ that is the assembly of the above three structs where the two primitives from $\overline{\Pi}$ are depicted as stars and γ_i^*'s are not delineated. Shading here is not related to that on the left, but rather to indicate the overlap area that will be shown in a similar way in some of the following.

Definition 19. For a level 0 class setting $\mathcal{C}$, a **level 1 constraint** $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$—involving two sets of level 0 class elements: $\overline{\mathscr{C}}$, the **pivot class elements**, and $\mathscr{C}^*$, $\overline{\mathscr{C}} \cap \mathscr{C}^* = \varnothing$—is defined as a pair

$$\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*) \;=\; \Big\langle \Big\{ \mathrm{CEL}^1\big(\mathscr{C}_j, \mathrm{Con}\,(\overline{\Pi}_j, \mathcal{UT}_j)\big) \Big\}_{1 \le j \le J}, \;\; \mathscr{C}^* \Big\rangle,$$

where $\bigcup\limits_j \mathscr{C}_j = \overline{\mathscr{C}}$ and the following condition is satisfied.

There exists:

level 1 struct

$$\sigma^1 \;=\; \langle\, \mathscr{C}_\sigma \,,\, \sigma \,\rangle\,,$$

set of class elements $\mathscr{C}'$, $\overline{\mathscr{C}} \subseteq \mathscr{C}' \subseteq \overline{\mathscr{C}} \cup \mathscr{C}^*$,
relabeling f, and bijection

$$g\colon \mathscr{C}' \to \mathscr{C}_\sigma\,,$$

such that

(i) $\forall\, \mathfrak{c}_i \in \mathscr{C}'\,, \quad \mathfrak{c}_i \in \mathfrak{C}_{k_i} \implies g(\mathfrak{c}_i) \in \mathfrak{C}_{k_i}$

(ii) $\forall\, j \quad \mathrm{CEL}^1\big(\, g(\mathscr{C}_j)\,,\, \mathrm{Con}\,(\overline{\Pi}_j\{f\}\,,\, \mathcal{UT}_j)\big)$ is a level 1 class element link between elements in $g(\mathscr{C}_j)$.

We say that the above **level 1 struct** σ^1 **satisfies level 1 constraint** $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$, and in the case when $\mathscr{C}^* = \varnothing$ we write simply $\mathrm{Con}^1(\overline{\mathscr{C}})$. In the case when $\overline{\mathscr{C}} = \mathscr{C}^* = \varnothing$ we call the constraint the **null constraint**, and denote it Θ^1. ▶

Analogous to Def. 10, the set of pivot class elements $\overline{\mathscr{C}}$ and the set of class element links $\Big\{ \mathrm{CEL}^1\big(\mathscr{C}_j, \mathrm{Con}\,(\overline{\Pi}_j, \mathcal{UT}_j)\big) \Big\}_{1 \le j \le J}$ identify the *necessary* structural components in every struct σ^1 satisfying constraint $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$, while the class elements in $\mathscr{C}^*$ are *optional* constituent elements that *may* accompany in σ^1 the necessary component of the constraint, thus allowing for a reasonable degree of freedom. Such optional constituent elements may include particular "environmental" and/or spurious class elements.

As can be seen from Fig. 26, it is *convenient* (although incomplete) to think of a level 1 constraint as a combination of previous-level constraints.

Definition 20. For a level 1 constraint $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$ a corresponding **level 1 active constraint** is defined as the following triple

$$\mathrm{ACon}^1(\,\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\,\mathrm{anc}}, \overline{\mathscr{C}}^{\,\mathrm{opn}}\,) \;=\; \big\langle\, \mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)\,, \overline{\mathscr{C}}^{\,\mathrm{anc}}, \overline{\mathscr{C}}^{\,\mathrm{opn}} \big\rangle,$$

where

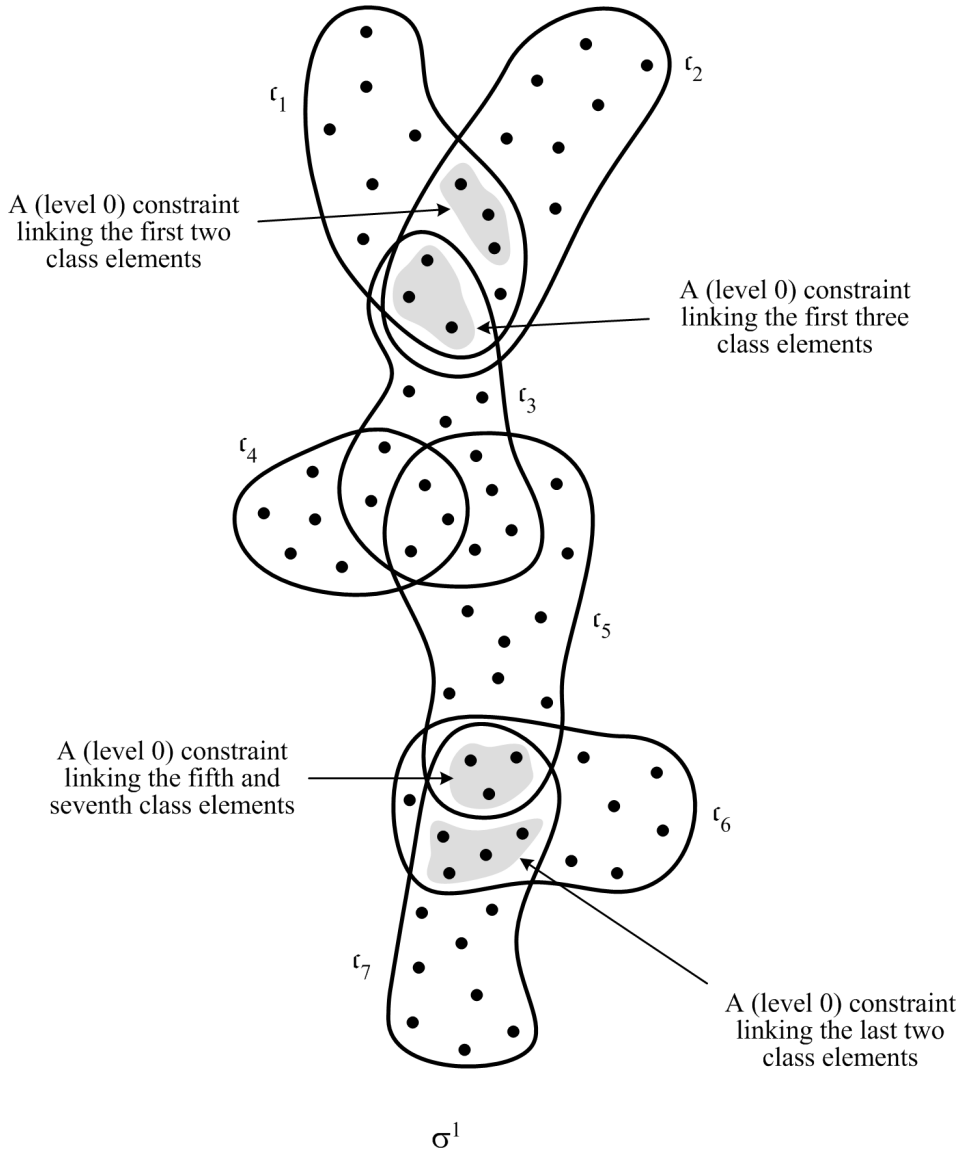


Figure 26: Pictorial illustration of a level 1 struct σ^1 that satisfies some constraint $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$. Class element $\mathfrak{c}_4$ is from set $\mathscr{C}^*$.

- the set of **anchor class element**, denoted $\overline{\mathscr{C}}^{\,\mathrm{anc}}$, is a *non-empty* proper subset of $\overline{\mathscr{C}}$ ($\overline{\mathscr{C}}^{\,\mathrm{anc}} \subset \overline{\mathscr{C}}$)
- the set of **open class elements**, denoted $\overline{\mathscr{C}}^{\,\mathrm{opn}}$, is a subset of $\overline{\mathscr{C}}$.

A **level 1 struct** α^1 **satisfying a level 1 active constraint**, or simply **active level 1 struct**, is a struct that satisfies the corresponding level 1 constraint $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*)$ and its *pivot* class elements inherit (under a class element bijection g) all anchor and open markings from the active constraint $\mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\,\mathrm{anc}}, \overline{\mathscr{C}}^{\,\mathrm{opn}})$. The **sets of anchor and open class elements in** α^1 are denoted $\overline{\mathscr{C}}_\alpha^{\,\mathrm{anc}}$ ($\overline{\mathscr{C}}_\alpha^{\,\mathrm{anc}} \neq \varnothing$) and $\overline{\mathscr{C}}_\alpha^{\,\mathrm{opn}}$, respectively.

In the case when $\mathrm{Con}^1(\overline{\mathscr{C}}, \mathscr{C}^*) = \Theta^1$, we call the active constraint the **null active level 1 constraint** and denote it Θ^1 also. ▶

Analogous to Def. 12, the next definition *restricts* the above family of level 1 structs, satisfying active constraint $\mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})$, to a particular subfamily of level 1 structs.

Definition 21. A **level 1 working struct** $\sigma^{w,1}$ is a level 1 struct some of whose class elements, subset $\overline{\mathscr{C}}^{\mathrm{opn}}_{\sigma^{w,1}}$, $\overline{\mathscr{C}}^{\mathrm{opn}}_{\sigma^{w,1}} \subseteq \mathscr{C}_{\sigma^{w,1}}$, are marked as open. For non-null active constraint $\mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})$ [41], the **set** $\mathrm{Ext}\left(\sigma^{w,1}, \mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})\right)$ **of active extensions of working struct** $\sigma^{w,1}$ **with respect to the level 1 active constraint** is defined as the set of active structs α^1 satisfying the following conditions:

(i) α^1 satisfies $\mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})$

(ii) level 1 assembly $\mathcal{A}(\sigma^{w,1}, \alpha^1)$ exists

(iii) $\overline{\mathscr{C}}^{\mathrm{anc}}_{\alpha} \subseteq \overline{\mathscr{C}}^{\mathrm{opn}}_{\sigma^{w,1}}$ [42]

(iv) $\overline{\mathscr{C}}_{\alpha} \setminus \mathscr{C}_{\sigma^{w,1}} \neq \varnothing$

In this case, the resulting level 1 assembly $\mathcal{A}(\sigma^{w,1}, \alpha^1)$, also denoted $\sigma^{w,1} \underset{\mathrm{Ext}}{\triangleleft} \alpha^1$, is defined to be a level 1 working struct with open class elements specified as follows: they are the class elements in $\left(\overline{\mathscr{C}}^{\mathrm{opn}}_{\sigma^{w,1}} \setminus \overline{\mathscr{C}}_{\alpha}\right) \cup \overline{\mathscr{C}}^{\mathrm{opn}}_{\alpha}$.

For the null active constraint, we define the set $\mathrm{Ext}\left(\sigma^{w,1}, \mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})\right)$ to be $\{\theta^1\}$. ▶

The two paragraphs of comments following Def. 12 apply here also.

We now proceed to the definition of a two-level class representation, remembering that class setting $\mathcal{C}$ is fixed. (This definition is practically a copy of Definition 13.)

Definition 22. A **two-level class representation** $\mathfrak{R}^1$ is a triple

$$\mathfrak{R}^1 = \left\langle \overline{\mathcal{C}}_{\mathfrak{R}}, \mathcal{C}^*_{\mathfrak{R}}, \mathscr{G}^1_{\mathfrak{R}} \right\rangle,$$

where $\overline{\mathcal{C}}_{\mathfrak{R}}$ ($\overline{\mathcal{C}}_{\mathfrak{R}} \subseteq \mathcal{C}$) is a set of **constituent pivot level 0 classes**, $\mathcal{C}^*_{\mathfrak{R}}$ ($\mathcal{C}^*_{\mathfrak{R}} \subseteq \mathcal{C}$) is a set of **constituent non-pivot level 0 classes**, and $\mathscr{G}^1_{\mathfrak{R}}$ is a partial specification of a **level 1 class generating system**:

$$\mathscr{G}^1_{\mathfrak{R}} = \left\langle \left\{ \mathrm{ACon}^1_{1,i}(\overline{\mathscr{C}}_{1,i}, \mathscr{C}^*_{1,i}, \overline{\mathscr{C}}^{\mathrm{anc}}_{1,i}, \overline{\mathscr{C}}^{\mathrm{opn}}_{1,i}) \right\}_{i \in I^1_1}, \left\{ \mathrm{ACon}^1_{2,i}(\overline{\mathscr{C}}_{2,i}, \mathscr{C}^*_{2,i}, \overline{\mathscr{C}}^{\mathrm{anc}}_{2,i}, \overline{\mathscr{C}}^{\mathrm{opn}}_{2,i}) \right\}_{i \in I^1_2},$$

[41] σ^1, w has no relation to the active constraint.

[42] Note that when $\overline{\mathscr{C}}^{\mathrm{opn}}_{\sigma^{w,1}} = \varnothing$, this condition is violated (since by definition $\overline{\mathscr{C}}^{\mathrm{anc}}_{\alpha} \neq \varnothing$) and hence $\mathrm{Ext}\left(\sigma^{w,1}, \mathrm{ACon}^1(\overline{\mathscr{C}}, \mathscr{C}^*, \overline{\mathscr{C}}^{\mathrm{anc}}, \overline{\mathscr{C}}^{\mathrm{opn}})\right) = \varnothing$.

$$\ldots, \left\{ \mathrm{ACon}^1_{t,i}(\overline{\mathscr{C}}_{t,i}, \mathscr{C}^*_{t,i}, \overline{\mathscr{C}}^{\mathrm{anc}}_{t,i}, \overline{\mathscr{C}}^{\mathrm{opn}}_{t,i}) \right\}_{i \in I^1_t} \Big\rangle ,$$

where each $\mathfrak{c} \in \overline{\mathscr{C}}_{j,i}$, $\mathfrak{c} \in \mathfrak{C}_k \in \overline{\mathcal{C}}_{\mathfrak{R}}$ and $\mathfrak{c} \in \mathscr{C}^*_{j,i}$, $\mathfrak{c} \in \mathfrak{C}_l \in \mathcal{C}^*_{\mathfrak{R}}$. Moreover, each of the above sets of active constraints specifies the corresponding step by $\mathscr{G}^1_{\mathfrak{R}}$. The *actions* of this generating system are described next.

Each step in the operation of the generating system $\mathscr{G}^1_{\mathfrak{R}}$ *follows* a "step" in the action of the **level 1 environment** $\mathfrak{E}^1$. Such environment could be thought of as comprised of a finite set of level 1 classes that can immediately interact with $\mathscr{G}^1_{\mathfrak{R}}$. Each such "step" by $\mathfrak{E}^1$ is specified by its own sets of active constraints. This interaction typically manifests itself in the environmental structs being attached to the current incomplete class element *in between* the steps of the generating system. Specifically, at step j, the system "responds" to the following j$^{\text{th}}$ step by $\mathfrak{E}^1$ in a manner specified below:

- step j by $\mathfrak{E}^1$: the environment $\mathfrak{E}^1$ may assemble several level 1 structs (each taken from a corresponding class) to the *previous level 1 working struct* $\sigma^1_{2(j-1)}$ to produce the *current level 1 working struct* σ^1_{2j-1}, whose pivot class element *markings are unchanged*; such step by $\mathfrak{E}^1$ could in fact be comprised of several "actual" steps (depending on the nature of the level 1 classes in $\mathfrak{E}^1$ participating in this step);

- step j by $\mathscr{G}^1_{\mathfrak{R}}$: in turn[43], denoting by γ^1_{j-1}, $\gamma^1_{j-1} \Subset \sigma^1_{2j-1}$, the $j-1$ **level 1 working class element**, i.e., the substruct of the current level 1 working struct formed by the class elements that have been contributed so far by $\mathscr{G}^1_{\mathfrak{R}}$ only, the class generating system

  - first nondeterministically/probabilistically generates [44] level 1 struct β^1_j from one of the non-empty sets of structs $\left\{ \mathrm{Ext}\left(\gamma^1_{j-1}, \mathrm{ACon}^1_{j,i}(\overline{\mathscr{C}}_{j,i}, \mathscr{C}^*_{j,i}, \overline{\mathscr{C}}^{\mathrm{anc}}_{j,i}, \overline{\mathscr{C}}^{\mathrm{opn}}_{j,i})\right) \right\}_{i \in I^1_j}$ and also satisfying:
    - $\mathscr{C}_{\beta^1_j} \setminus \overline{\mathscr{C}}_{\beta^1_j} \subseteq \mathscr{C}_{\sigma^1_{2j-1}}$
    - $\mathcal{A}(\beta^1_j, \sigma^1_{2j-1})$ exists
  - assembles β^1_j to σ^1_{2j-1} to produce the next level 1 working struct σ^1_{2j}; moreover, it is not difficult to see that performing the latter assembly also accomplishes the following assembly (since $\gamma^1_{j-1} \Subset \sigma^1_{2j-1}$) producing the next working class element $\gamma^1_j = \gamma^1_{j-1} \underset{\mathrm{Ext}}{\lhd} \beta^1_j$ (to appropriately allocate "open" markings). See Fig. 27.

Note that the initial step is a simplified version of the above generic step, in which the very first step by the environment produces not only (the current working) struct σ^1_1, but also its substruct, the initial working class element γ^1_0 with the appropriate markings.

[43] The class generating system acts without reference to the structure of $\mathfrak{E}^1$, i.e. it does not know the structure of the classes that affected the working struct.

[44] We draw attention to the process of generation (of β^1_j), which relies on the relevant part of the working struct as well as on the active constraint as it instantiates—usually in parallel—the appropriate level 0 class elements (via the corresponding level 0 generating systems).

The generating system $\mathscr{G}^1_{\mathfrak{R}}$ contains its own terminating condition (not presently specified) for completing struct γ^1, where $\gamma^1 = \mathcal{A}(\beta^1_1, \beta^1_2, \ldots, \beta^1_t)$, $\gamma^1 \Subset \sigma^1_{2t}$ (σ^1_{2t} is the final working struct). For any such struct γ^1, the pair

$$\mathfrak{c}^1 \;=\; \langle\, \gamma^1\,,\, \mathscr{G}^1_{\mathfrak{R}} \,\rangle$$

is called a **class element** of the **two-level class** $\mathfrak{C}(\mathfrak{R}^1, \mathfrak{E}^1)$ **induced by** $\mathfrak{R}^1$ **in environment** $\mathfrak{E}^1$ (see Fig. 28). When no confusion arises, this class is also denoted $\mathfrak{C}^1$. ▶

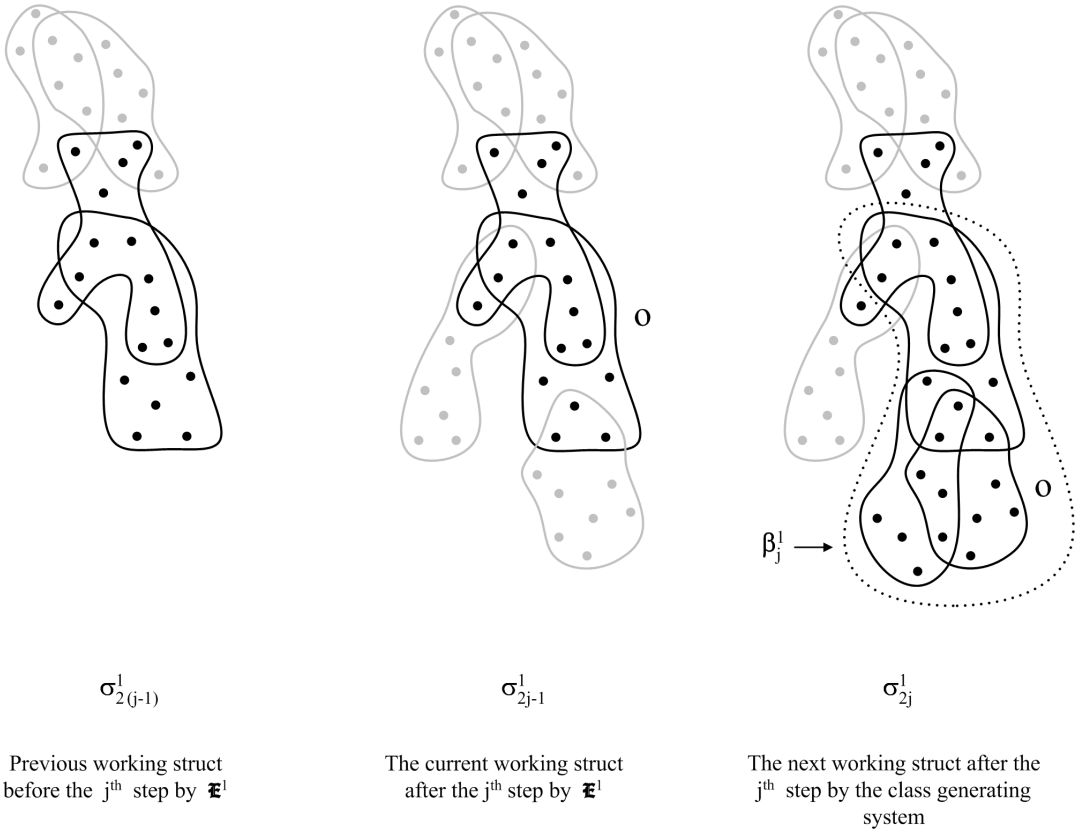


Figure 27: Pictorial representation of two steps—a step by the environment $\mathfrak{E}^1$ and the corresponding step by level 1 class generating system $\mathscr{G}^1_{\mathfrak{R}}$—in the construction of class element $\mathfrak{c}^1$, $\mathfrak{c}^1 \in \mathfrak{C}^1$. Dots stand for primitives, and lines delineate level 0 class elements. Grey class elements are those added by the environment, and the dotted line delineates the level 1 active struct assembled to σ^1_{2j+1} (and satisfying the corresponding level 1 active constraint). Elements that remain shaded at the end of the generating process are not part of $\mathfrak{c}^1$. Note that because the bottom grey class element in σ^1_{2j+1} happened to be in β^1_j, it *became* part of the working class element in σ^1_{2j+2}.

All comments following Def. 13 apply here also.

We draw your attention to the following *synonymous terms*: two-level class (element) and level 1 class (element).

As was done in the previous section for level 0 classes, we are now ready to define a class setting for level 1.

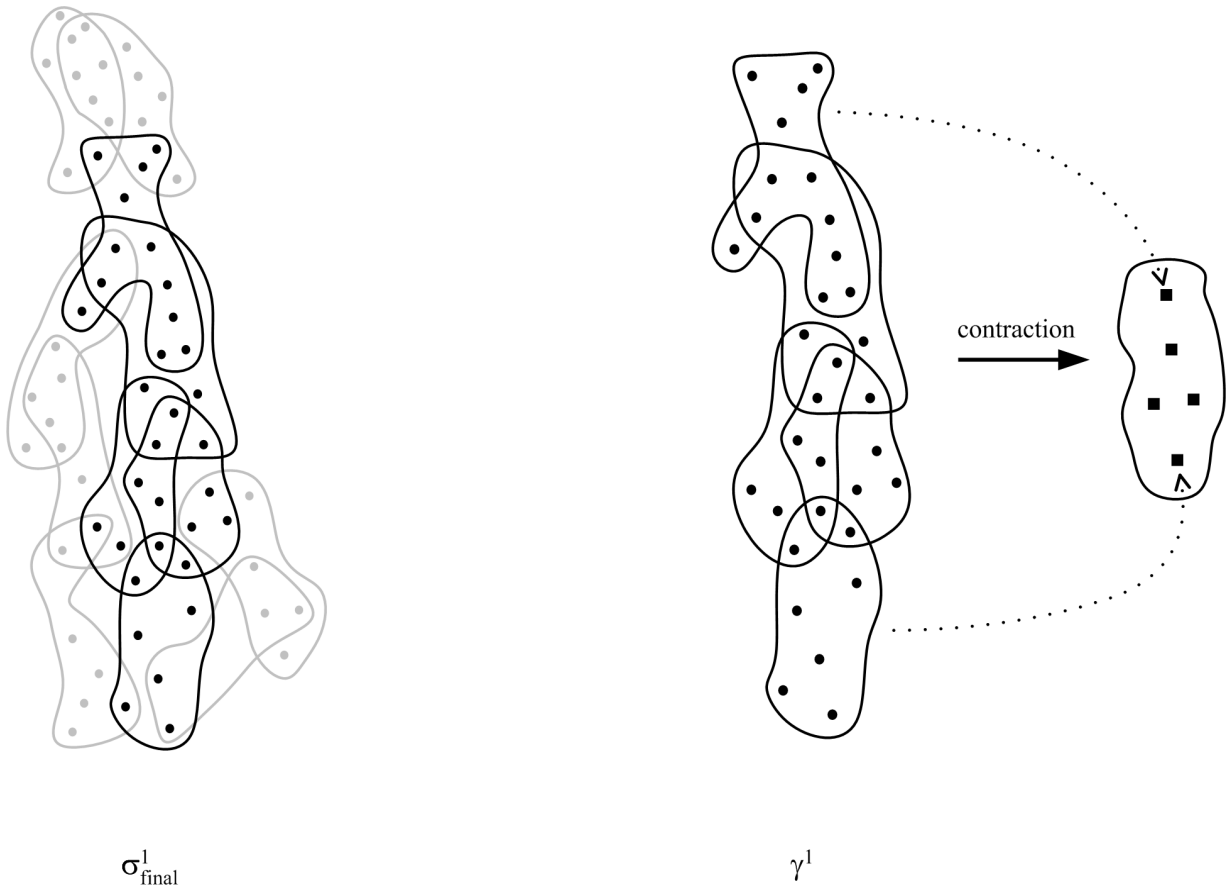


Figure 28: *Left*: Pictorial representation of the final output of the class generating system, the level 1 working struct σ^1_{final} (which is a "completion" of the last struct shown in Fig. 27). *Right*: The final working class element γ^1, which is a substruct of struct σ^1_{final}, where γ^1 is the struct of the class element $\mathfrak{c}^1$, $\mathfrak{c}^1 = \langle \gamma^1 , \mathscr{G}^1_{\mathfrak{R}} \rangle$. On the far right, we show a depiction of $\mathfrak{c}^1$ as a (potential) component of a level 2 struct: squares stand for contracted class elements.

Definition 23. We introduce a finite set of two-level classes

$$\mathcal{C}^1 \;=\; \{\, \mathfrak{C}^1_1 , \mathfrak{C}^1_2 , \ldots , \mathfrak{C}^1_{l_1} \,\} ,$$

l_1 is the number of two-level classes, which we call a **level 1 class setting**. ▶

The evolution of the universe suggests that the following inequalities between the corresponding cardinalities emerge with time:

$$m < l_0 < l_1 .$$

Indeed, these inequalities are quite natural since, for example, the number of classes is much larger than the number of primitives.

Level 2 analogues of Definitions 15–17 (including **level 2 struct**, **substruct**, and **assembly of level 2 structs**) follow immediately. For convenience, we present an analogue of Definition 15 only.

Definition 24. Having fixed class setting $\mathcal{C}^1$, a **level 2 struct** σ^2 is defined as a pair

$$\sigma^2 \;=\; \langle\, \mathscr{C}^1_{\sigma^1}\,,\, \sigma^1 \,\rangle\,,$$

where the **level 1 class-based representation of level 1 struct** σ^1

$$\mathscr{C}^1_{\sigma^1} \;=\; \{\, \mathfrak{c}^1_1\,,\, \mathfrak{c}^1_2\,,\, \ldots\,,\, \mathfrak{c}^1_{s_1} \,\}$$

is defined in such a way that $\mathfrak{c}^1_i \in \mathfrak{C}^1_j$, $\mathfrak{c}^1_i \;=\; \langle\, \gamma^1_i\,,\, \mathscr{G}^1_{\mathfrak{R}_i} \,\rangle$, and $\sigma^1 \;=\; \mathcal{A}\big(\, \gamma^1_1\,,\, \gamma^1_2\,,\, \ldots\,,\, \gamma^1_{s_1} \,\big)$. We refer to $\mathfrak{c}^1_i$ as a **constituent element of** σ^2.

The set of all level 2 structs will be denoted Σ^2, and when $\mathscr{C}^1_{\sigma^1} \;=\; \varnothing$, struct σ^2 will be called the **null level 2 struct**, denoted θ^2. ▶

[Important] **Remark 6.** We draw the reader's attention to the following incompleteness in our presentation: we have not yet *finalized* the generalization of the concept of primitive attachment (and therefore the relation SL) to class elements in a higher-level struct in such a way that would describe the interrelationships between the various constituent class elements. ◗

7 Higher-level class representations

Finally, we are going to discuss the general form of class representation, which is a straightforward adaptation of a two-level class representation. To avoid a bacchanalia of level-related indices in the class representation for a general level k—which is *absolutely* analogous to the one presented below and in Def. 22—it is sufficient to restate the level 1 (2) definitions for level 2 (3), particularly in view of the fact that the number of levels is always expected to be very small.

We proceed to the definition of a three-level class representation, where analogues of Defs. 18, 19, 20 and 21 are omitted (since they are trivial modifications of the latter definitions). The corresponding notations are: $\mathrm{CEL}^2\big(\,\mathscr{C}^1,\, \mathrm{Con}^1(\overline{\mathscr{C}},\, \mathscr{C}^*)\,\big)$, $\mathrm{Con}^2(\overline{\mathscr{C}}^1,\, \mathscr{C}^{*,1})$, $\mathrm{ACon}^2(\,\overline{\mathscr{C}}^1,\, \mathscr{C}^{*,1},\, \overline{\mathscr{C}}^{\,\mathrm{anc},1},\, \overline{\mathscr{C}}^{\,\mathrm{opn},1}\,)$, and $\mathrm{Ext}\,\big(\sigma^{w,2},\, \mathrm{ACon}^2(\,\overline{\mathscr{C}}^1,\, \mathscr{C}^{*,1},\, \overline{\mathscr{C}}^{\,\mathrm{anc},1},\, \overline{\mathscr{C}}^{\,\mathrm{opn},1}\,)\big)$.

Definition 25. A **three-level class representation** $\mathfrak{R}^2$ is a triple

$$\mathfrak{R}^2 \;=\; \big\langle\, \overline{\mathcal{C}}^1_{\mathfrak{R}}\,,\, \mathcal{C}^{*,1}_{\mathfrak{R}}\,,\, \mathscr{G}^2_{\mathfrak{R}} \,\big\rangle\,,$$

where $\overline{\mathcal{C}}^1_{\mathfrak{R}}$ $(\,\overline{\mathcal{C}}^1_{\mathfrak{R}} \subseteq \mathcal{C}^1\,)$ is a set of **constituent pivot level 1 classes**, $\mathcal{C}^{*,1}_{\mathfrak{R}}$ $(\,\mathcal{C}^{*,1}_{\mathfrak{R}} \subseteq \mathcal{C}^1\,)$ is a set of **constituent non-pivot level 1 classes**, and $\mathscr{G}^2_{\mathfrak{R}}$ is a partial specification of a **level 2 class generating system**:

$$\mathscr{G}^2_{\mathfrak{R}} = \Big\langle\, \big\{\, \mathrm{ACon}^2_{1,i}(\,\overline{\mathscr{C}}^1_{1,i},\, \mathscr{C}^{*,1}_{1,i},\, \overline{\mathscr{C}}^{\,\mathrm{anc},1}_{1,i},\, \overline{\mathscr{C}}^{\,\mathrm{opn},1}_{1,i}\,)\,\big\}_{i\in I^2_1}\,,\, \ldots$$

$$\ldots\,,\, \big\{\, \mathrm{ACon}^2_{t,i}(\,\overline{\mathscr{C}}^1_{t,i},\, \mathscr{C}^{*,1}_{t,i},\, \overline{\mathscr{C}}^{\,\mathrm{anc},1}_{t,i},\, \overline{\mathscr{C}}^{\,\mathrm{opn},1}_{t,i}\,)\,\big\}_{i\in I^2_t} \,\Big\rangle\,,$$

where each $\mathfrak{c}^1 \in \overline{\mathscr{C}}^1_{j,i}$, $\mathfrak{c}^1 \in \mathfrak{C}^1_k \in \overline{\mathcal{C}}^1_{\mathfrak{R}}$ and $\mathfrak{c} \in \mathscr{C}^{*,1}_{j,i}$, $\mathfrak{c}^1 \in \mathfrak{C}^1_l \in \mathcal{C}^{*,1}_{\mathfrak{R}}$. Moreover, each of the above sets of active constraints specifies the corresponding step by $\mathscr{G}^2_{\mathfrak{R}}$. The *actions* of this generating system are described next.

Each step in the operation of the generating system $\mathscr{G}^2_{\mathfrak{R}}$ *follows* a "step" in the action of the **level 2 environment** $\mathfrak{E}^2$. Such environment could be thought of as comprised of a finite set of level 2 classes that can immediately interact with $\mathscr{G}^2_{\mathfrak{R}}$. Each such "step" by $\mathfrak{E}^2$ is specified by its own sets of active constraints. This interaction typically manifests itself in the environmental structs being attached to the current incomplete class element *in between* the steps of the generating system. Specifically, at step j, the system "responds" to the following j$^{\text{th}}$ step by $\mathfrak{E}^2$ in a manner specified below:

- step j by $\mathfrak{E}^2$: the environment $\mathfrak{E}^2$ may assemble several level 2 structs (each taken from a corresponding class) to the *previous level 2 working struct* $\sigma^2_{2(j-1)}$ to produce the *current level 2 working struct* σ^2_{2j-1}, whose pivot class element *markings are unchanged*; such step by $\mathfrak{E}^2$ could in fact be comprised of several "actual" steps (depending on the nature of the level 2 classes in $\mathfrak{E}^2$ participating in this step);

- step j by $\mathscr{G}^2_{\mathfrak{R}}$: in turn[45], denoting by γ^2_{j-1}, $\gamma^2_{j-1} \Subset \sigma^2_{2j-1}$, the $j-1$ **level 2 working class element**, i.e., the substruct of the current level 2 working struct formed by the class elements that have been contributed so far by $\mathscr{G}^2_{\mathfrak{R}}$ only, the class generating system

  - first nondeterministically/probabilistically generates[46] level 2 struct β^2_j from one of the non-empty sets of structs $\left\{ \text{Ext}\left(\gamma^2_{j-1}, \text{ACon}^2_{j,i}(\overline{\mathscr{C}}^1_{j,i}, \mathscr{C}^{*,1}_{j,i}, \overline{\mathscr{C}}^{\text{anc},1}_{j,i}, \overline{\mathscr{C}}^{\text{opn},1}_{j,i})\right)\right\}_{i \in I^2_j}$ and also satisfying:
    - $\mathscr{C}^1_{\beta^2_j} \setminus \overline{\mathscr{C}}^1_{\beta^2_j} \subseteq \mathscr{C}^1_{\sigma^2_{2j-1}}$
    - $\mathcal{A}(\beta^2_j, \sigma^2_{2j-1})$ exists
  - assembles β^2_j to σ^2_{2j-1} to produce the next level 2 working struct σ^2_{2j}; moreover, it is not difficult to see that performing the latter assembly also accomplishes the following assembly (since $\gamma^2_{j-1} \Subset \sigma^2_{2j-1}$) producing the next working class element $\gamma^2_j = \gamma^2_{j-1} \underset{\text{Ext}}{\lhd} \beta^2_j$ (to appropriately allocate "open" markings).

Note that the initial step is a simplified version of the above generic step, in which the very first step by the environment produces not only (the current working) struct σ^2_1, but also its substruct, the initial working class element γ^2_0 with the appropriate markings.

The generating system $\mathscr{G}^2_{\mathfrak{R}}$ contains its own terminating condition (not presently specified) for completing struct γ^2, where $\gamma^2 = \mathcal{A}(\beta^2_1, \beta^2_2, \ldots, \beta^2_t)$, $\gamma^2 \Subset \sigma^2_{2t}$ (σ^2_{2t} is the final

[45] The class generating system acts without reference to the structure of $\mathfrak{E}^2$, i.e. it does not know the structure of the classes that affected the working struct.

[46] We draw attention to the process of generation (of β^2_j), which relies on the relevant part of the working struct as well as on the active constraint as it instantiates—usually in parallel—the appropriate level 1 class elements (via the corresponding level 1 generating systems).

working struct). For any such struct γ^2, the pair

$$\mathfrak{c}^2 \;=\; \langle\, \gamma^2\,,\, \mathscr{G}^2_{\mathfrak{R}} \,\rangle$$

is called a **class element** of the **three-level class** $\mathfrak{C}(\mathfrak{R}^2, \mathfrak{E}^2)$ **induced by** $\mathfrak{R}^2$ **in environment** $\mathfrak{E}^2$. When no confusion arises, this class is also denoted $\mathfrak{C}^2$. ▶

Obviously, for some chosen application, for a particular level, the effect of the environment on a class may manifest itself to a "larger or lesser extent" as compared to that on a class at a different level, i.e. at a particular level, there may either be more or less environmental classes interacting with the class in question.

Next, level 2 analogues of **level 2 class setting** and $\mathfrak{C}^2_i$ **w.r.t.** $\mathcal{C}^2$, and level 3 analogues of **level 3 struct**, its **substruct**, and **assembly of level 3 structs**, follow immediately.

Speculative remark. One of the anticipated consequences of the above class representation definition is that it should facilitate the introduction of an analogue of the concept of a *topology* for a class (i.e. how "close") and also facilitates the emergence of a "class topography" (i.e. how "typical"). Moreover, it is natural to assume that a fixed set of classes (at all previous levels) do not uniquely induce the topology of a next-level class: it is the overall structure, or "structural features", of the generating system that should ensure uniqueness. ◗

For a very early attempt to implement the concept of structural generativity via Markov stochastic processes in the ETS formalism, see [1], [42], [43] (where this was accomplished in a more conventional manner by attaching numeric weights to transforms, where the latter concept is introduced in the next section).

8 An illustrative example: the class of "Bubble Men"

In this section we present, in diagrammatic form, a relatively simple but in some sense characteristic example of a 3-level class and its generating system, the **Bubble Man** class. In the choice, we were motivated by considerations coming from both developmental biology and ETS. We want, at least to some extent, to "hit two targets with one stone": to illustrate, one, the concept of the ETS generating system (as it might apply, for example, to shape representation) as well as, two, the emerging *overall* view of biological developmental processes [47]. The example is presented in a pictorial form and the following figures should be read sequentially.

As far as the biological analogy of our Bubble Man example is concerned, as one can readily see, some level 0 classes correspond to different biological developmental stages, i.e., to local groups of (biological) cells appearing at different times during development. This is because a horizontal slice through a level 0 struct corresponds to a particular time instant in the development (see Fig. 40). Thus, one can draw analogy between our bubble-man

[47] Obviously, the *concrete* details of the example should *in no way* be interpreted as related to the actual developmental processes.

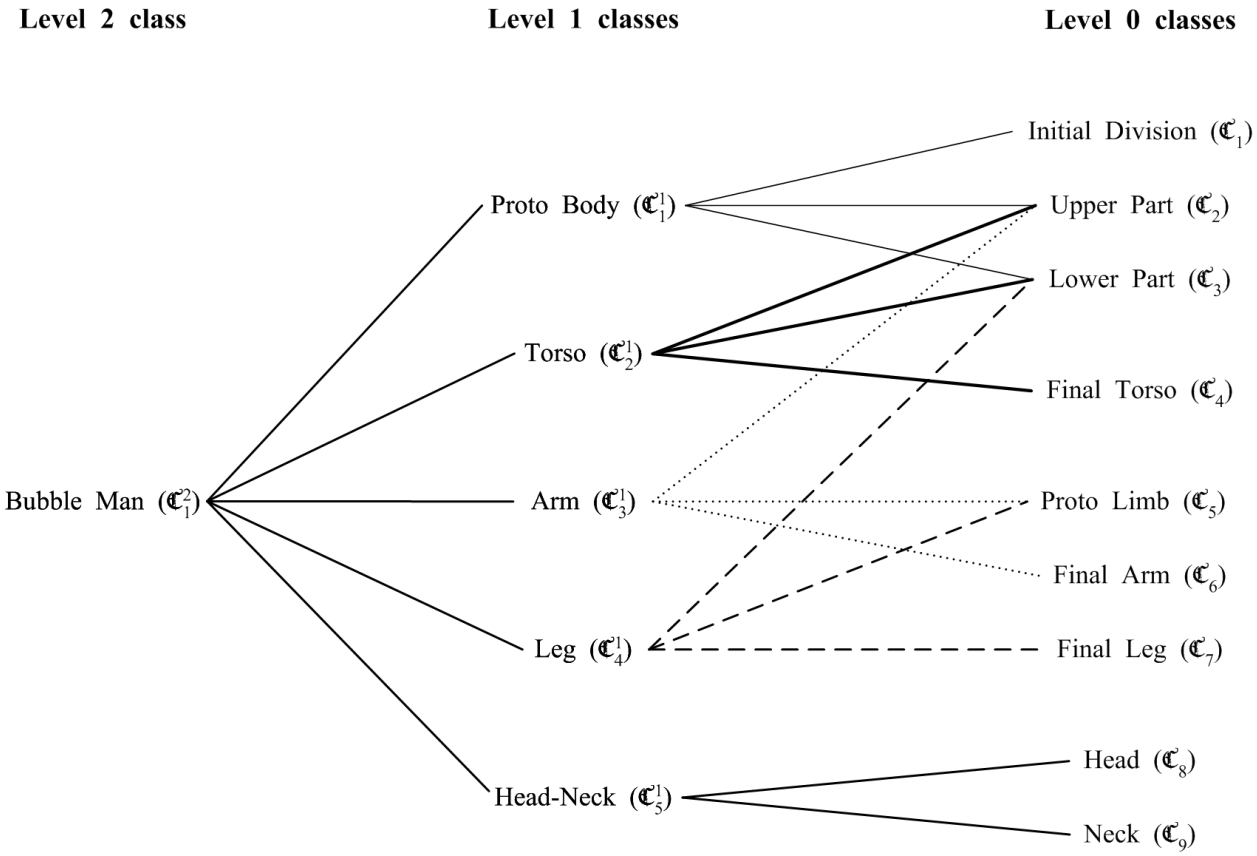


Figure 29: Names of various classes in this example (at three levels). Lines point to the constituent classes.

ovals and biological cells: analogous to cells, our ovals also become more "specialized" as time progresses. We have "cells" at the beginning, and we have (many more specialized) "cells" at the end, but the representational language is the same for all stages. However, it is important to keep in mind that each of our primitives (in the next figure) correspond not a cell but to the appropriate event transforming one or several cells.

We should note that all depicted level 0 structs are abstract structs (see Def. 7), since we do not show the labels for the primal processes, i.e. for the elements of the primal "sub-classes" involved (see Appendix). Also, the considerations of symmetry dictate that—when generating concrete left/right proto limbs, final arms and final legs—the corresponding generating system should make *identical choices* (except for left/right variations in the primal "sub-classes") of the constraints at each of its steps.

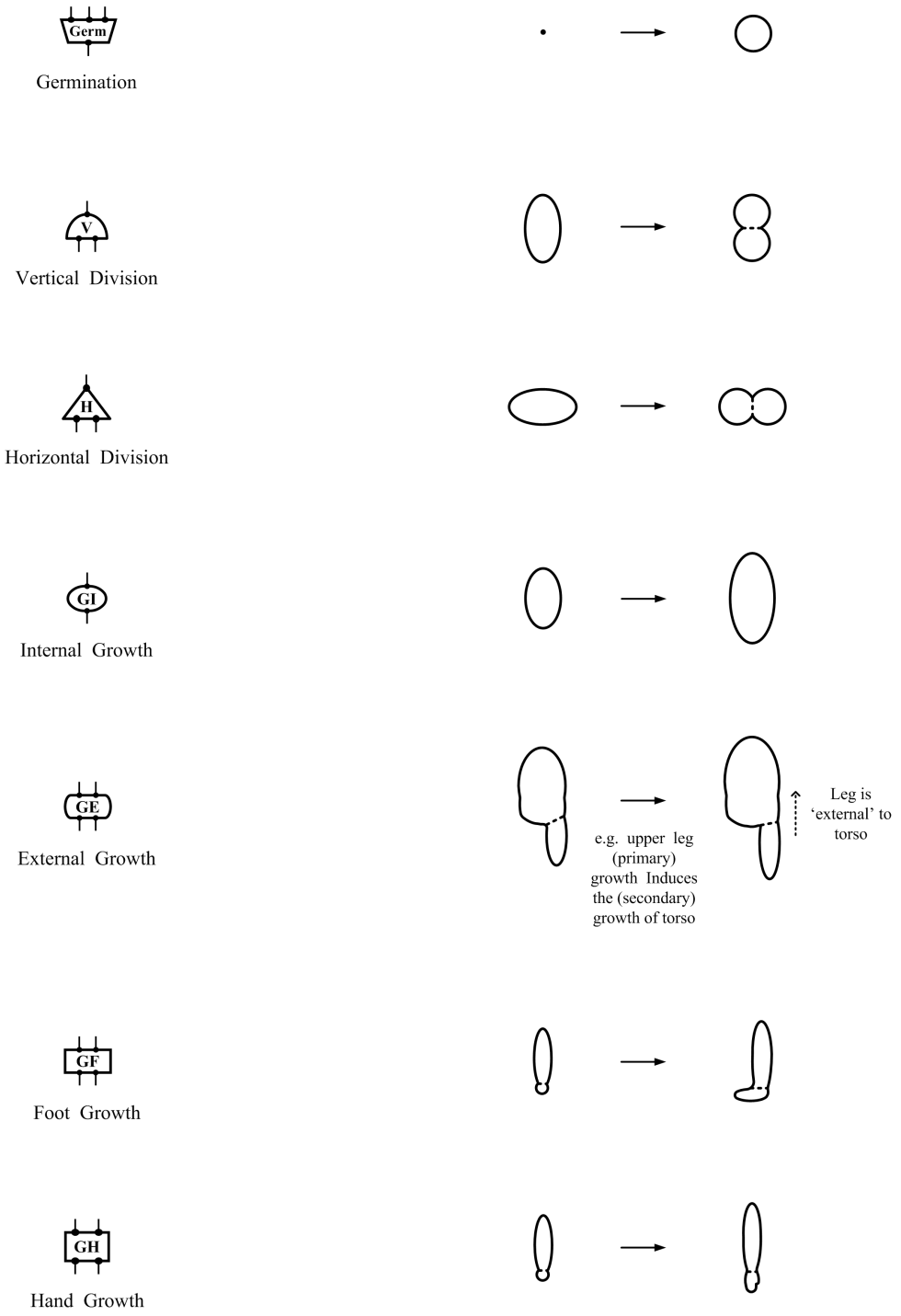


Figure 30: *Left*: The primitive-classes (see Appendix) used in this example. All primal processes (except the initials for germination) belong to the same class of oval-like "cells", i.e., there is only one primal class with its "sub-classes" shown in the next two figures. The last three primitives have similar structure: the left initial is associated with a single cell and the right initial is associated with one of its neighbours, while the left terminal is associated with the enlarged original cell and the right, with the appropriately modified neighbouring cell. So, the three events each produce the modification of the original cell and also of its neighbour. (Obviously, we could have introduced similar primitives with more initial and terminal sites, responsible for the modification of several neighbouring cells, and in general, we could have split each of these primitive into several.) *Right*: "Geometric" encapsulations of the corresponding primitive events.

Abbreviations for the primal "Sub-Classes" of the Basic Cell primal class

g — germ cell	Lur — upper right Limb cell	Lul — upper left Limb cell
ub — upper body cell	Llr — lower right Limb cell	Lll — lower left Limb cell
lb — lower body cell	LGur — upper right Leg cell	LGul — upper left Leg cell
hn — head-neck cell	LGlr — lower right Leg cell	LGll — lower left Leg cell
t — torso cell	Aur — upper right Arm cell	Aul — upper left Arm cell
h — head cell	Alr — lower right Arm cell	All — lower left Arm cell
n — neck cell	flr — right foreleg cell	fll — left foreleg cell
Lu — upper Limb cell	far — right forearm cell	fal — left forearm cell
Ll — lower Limb cell	fr — right foot cell	fl — left foot cell
	hr — right hand cell	hl — left hand cell

Figure 31: *All* postulated primal "sub-classes" of the single primal class that we call Basic Cell. In the abbreviations capitals stand for: L for "limb", LG for "leg", and A for "arm".

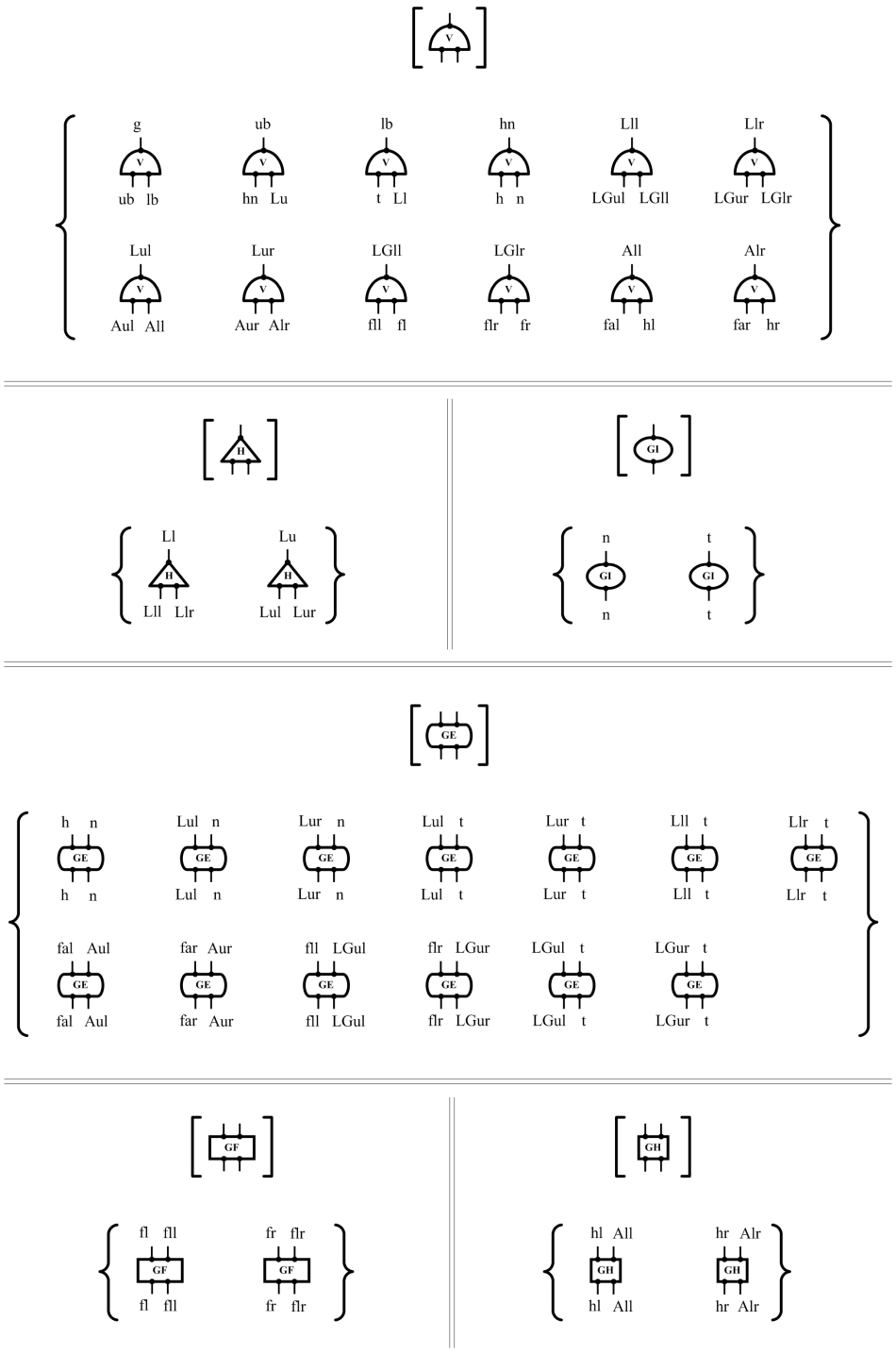
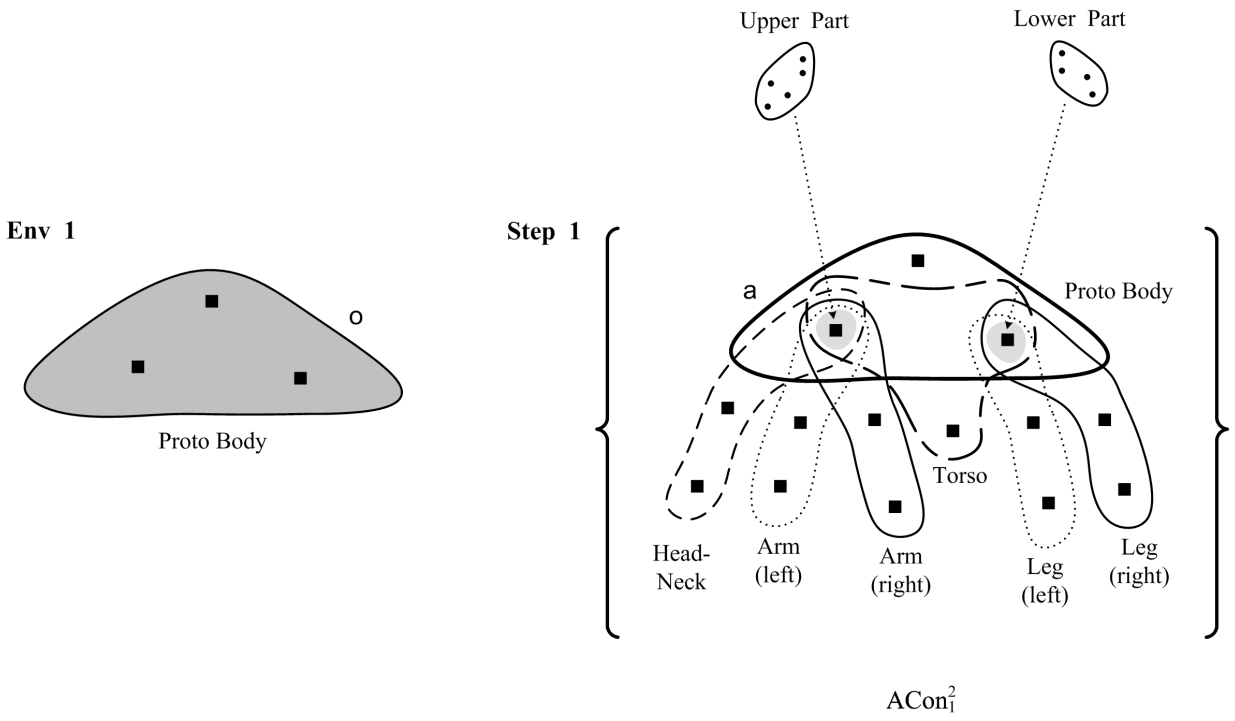


Figure 32: Primitive-Classes and their elements, except for the germination primitive-class (since the specification of the last one not critical). See Appendix for the concept of primtive-class.



"Bubble Man" Class Generating System

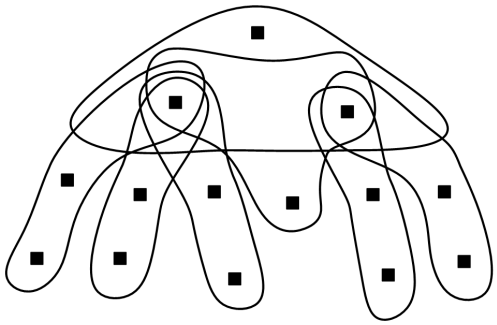
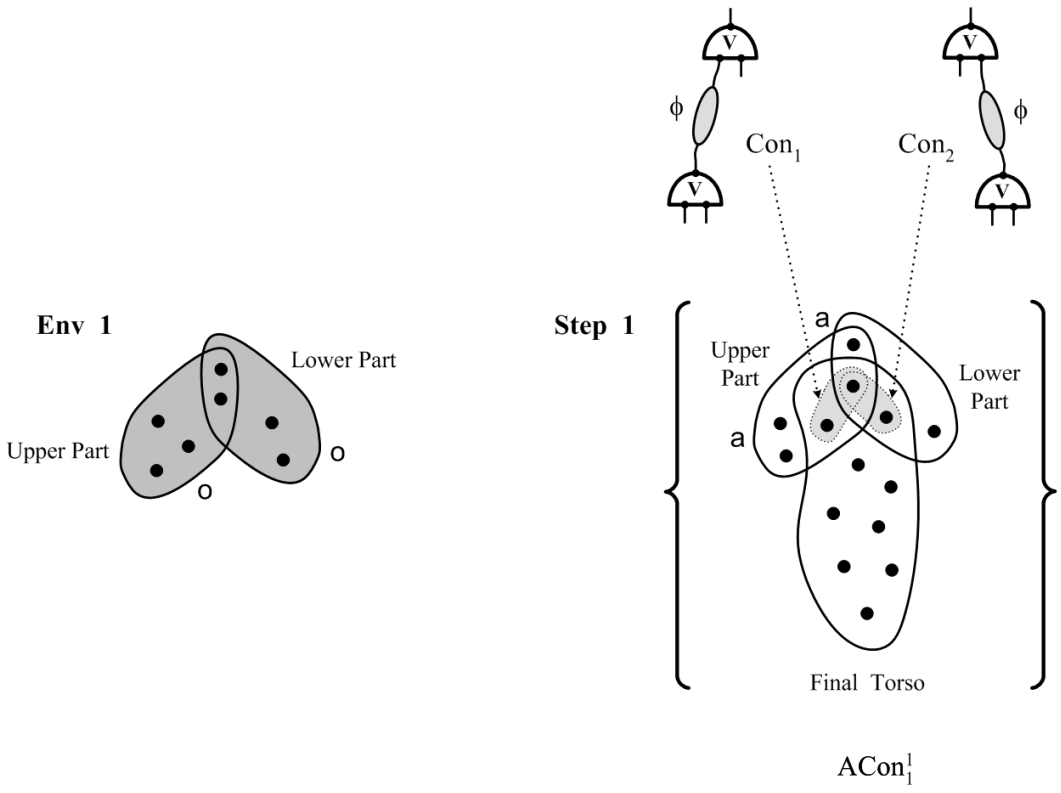
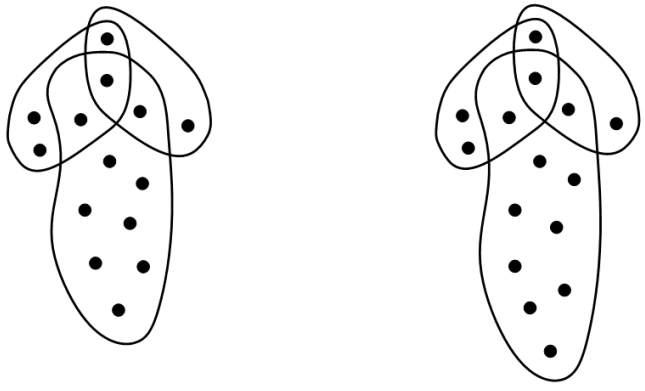


Figure 33: Pictorial description of level 2 (the highest level) for the **Bubble Man** generating system. This descriptions is comprised of two steps only, one by the environment (left) and one by the generating system itself. (Reminder: *The above shapes should be interpreted temporally.*) *Top Right*: the singleton set (large braces) consisting of the level 2 active constraint specifying the only step by the generating system. Various lines delineate level 1 class elements with the class names identified, while the dark squares stand for their constituent level 0 class elements (expanded in the following figures). Above the constraint, two shaded regions—each of which is the overlap of several level 1 structs involved in a level 2 class element link—are expanded to show the corresponding level 1 structs $\gamma_1'^{,1}$ and $\gamma_2'^{,1}$ (without the associated level 1 constraints, analogous to Fig. 25). In this case, each $\gamma_i'^{,1}$ happens to consist of a single constituent level 0 class element. *Bottom*: The only level 2 struct.



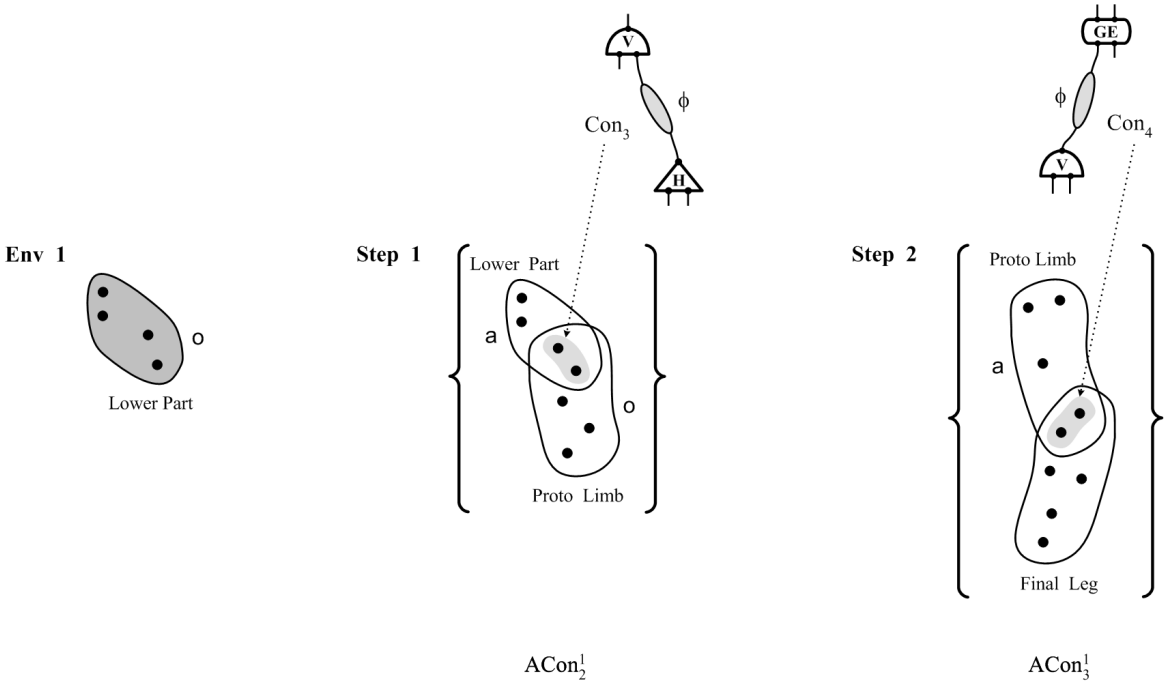
"Torso" Class Generating System



a torso

a wider torso

Figure 34: Pictorial description of level 1 generating system for class **Torso**. This description is comprised of two steps, one by the environment and one by the generating system itself. Various lines delineate the constituent level 0 class elements with the class names identified, while the dots stand for their constituent primitives. *Top*: The above two steps: the step by the generating system is depicted as a singleton set containing a level 1 active constraint with the corresponding level 0 constraints shown above. *Bottom*: Two examples of level 1 structs generated by this generating system.



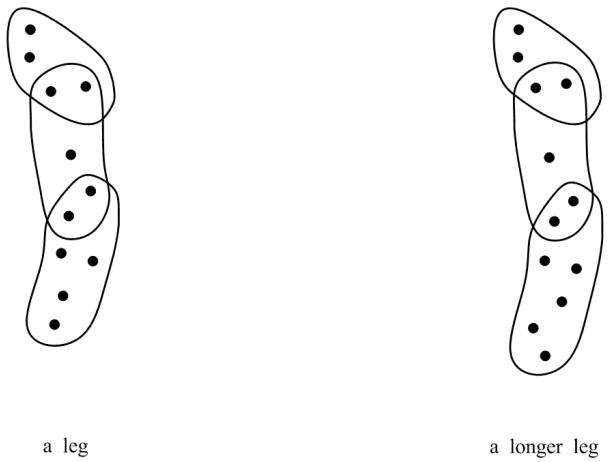


Figure 35: Pictorial description of level 1 generating system for class **Leg**. This description is comprised of three steps, one by the environment and two by the generating system itself. Various lines delineate the constituent level 0 class elements with the class names identified, while the dots stand for their constituent primitives *Top*: The above three steps: the two steps by the generating system are depicted by singleton sets of level 1 active constraints with the corresponding level 0 constraints shown above. *Bottom*: Two examples of level 1 structs generated by this generating system.

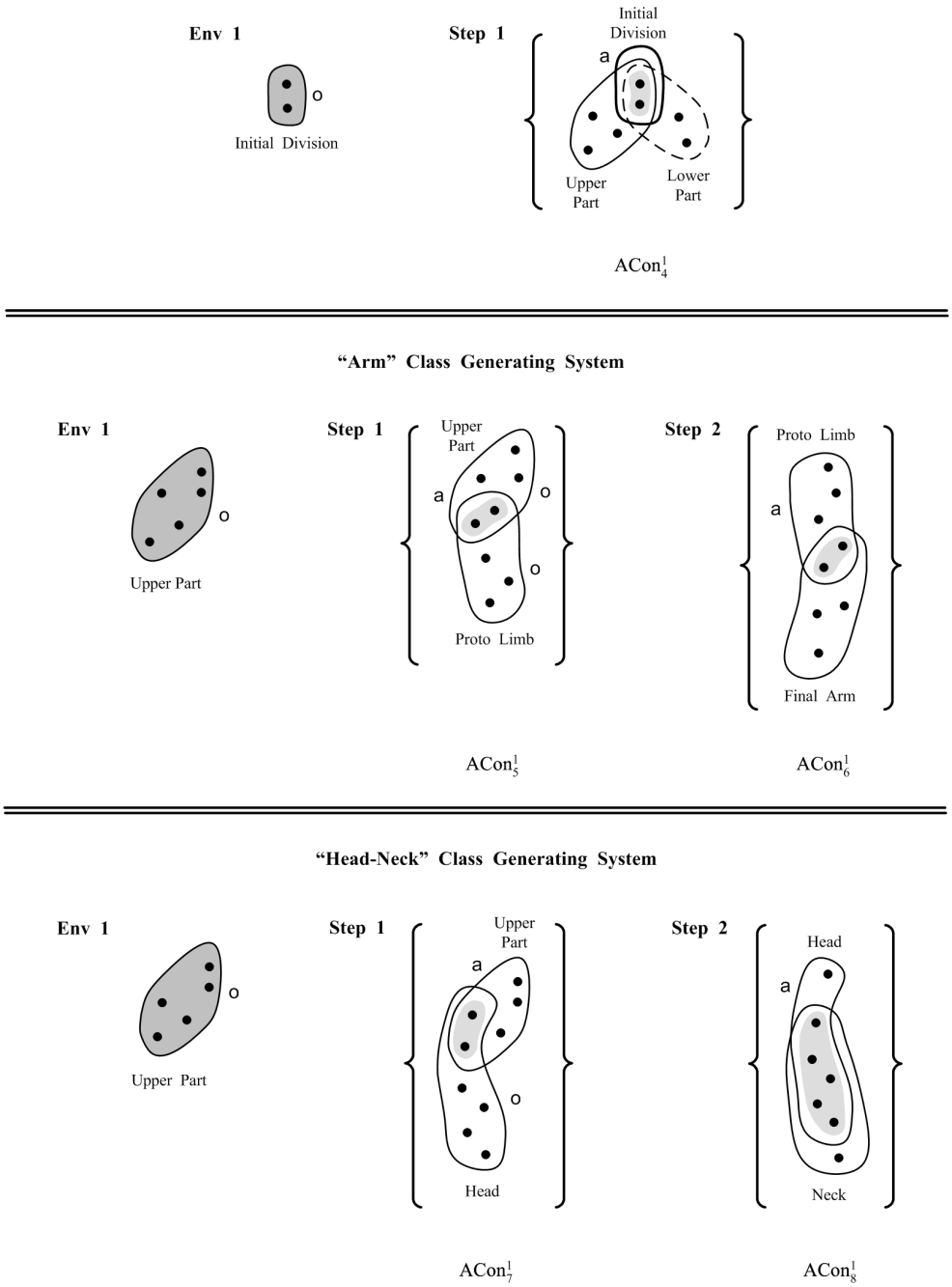


Figure 36: Stepwise specification (via sets of active constraints) of the remaining three level 1 class generating systems. The corresponding level 0 constraints are not shown.

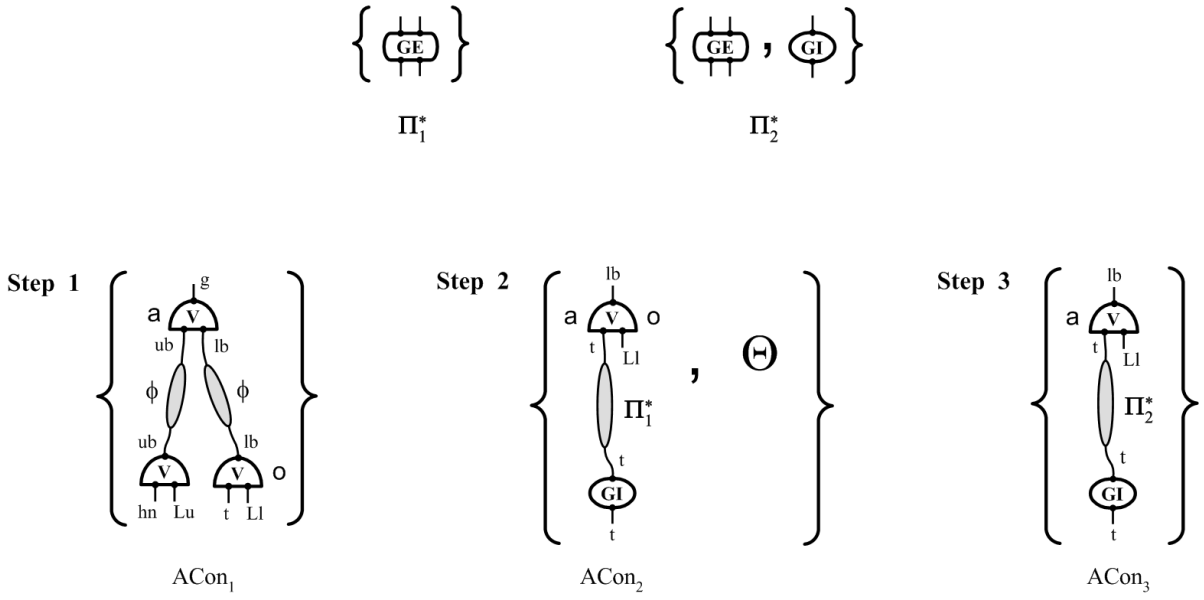


Figure 37: Pictorial description of level 0 generating system for class **Final Torso**. This description is comprised of three steps by the generating system, where for simplicity, the following steps by the environment are omitted. The first environmental step results in the initial Vertical Division primitive and the later steps result in the addition of one or several External Growth primitives, which are contributed by the various limb generating systems. The shown three steps are those by the generating system: each step is specified by a set containing one or two level 0 active constraints. (For this and other level 0 class representations in this example: in *all* unit-constraints *all* formation rules FR do not impose any restrictions on the set of admissible pivot primitives.)

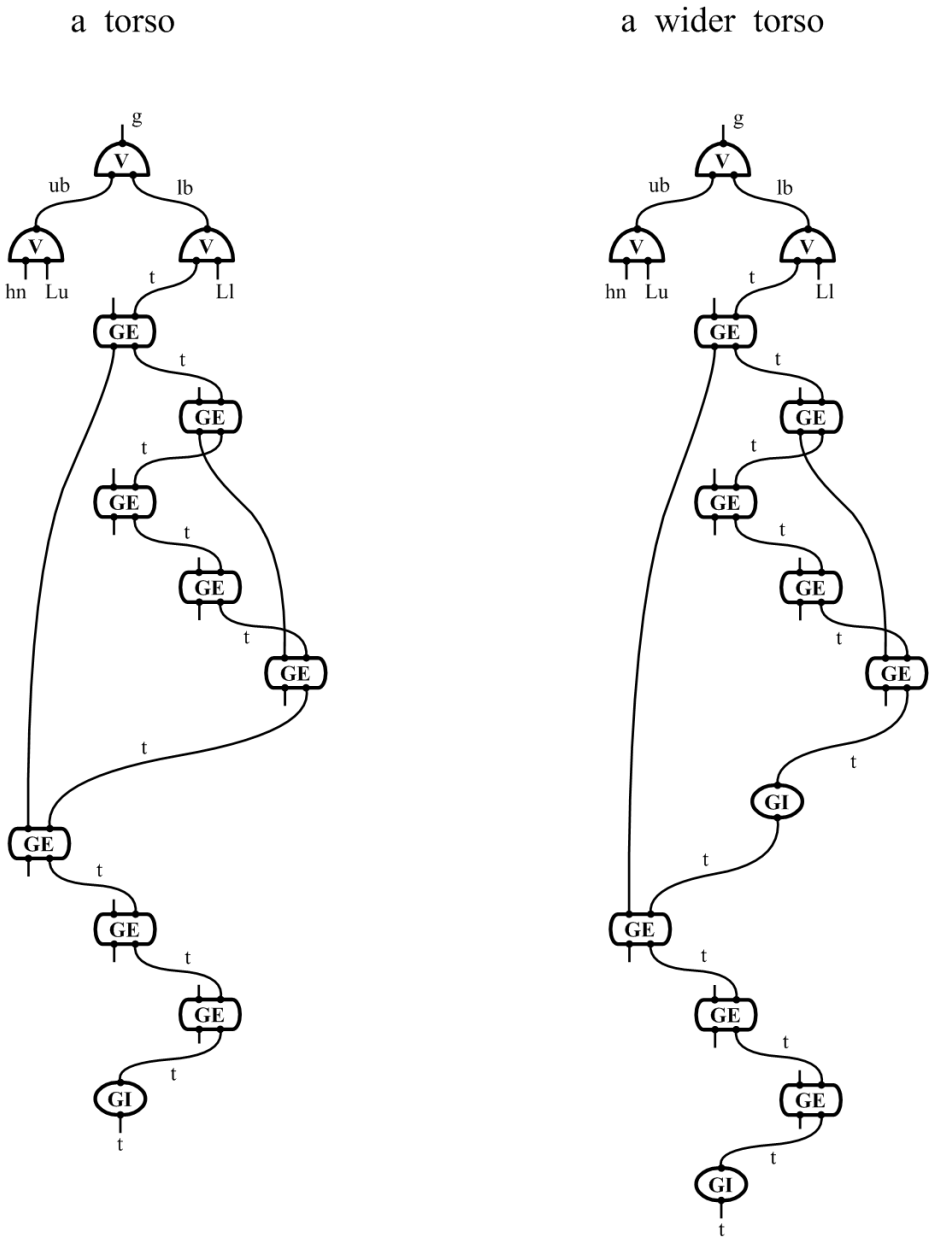


Figure 37 (continued): Two examples of level 0 structs generated by the above generating system. Some primal "sub-class" labels are omitted. Note the substantial contribution of the environment to the elements of this class.

"Final Leg" Class Generating System

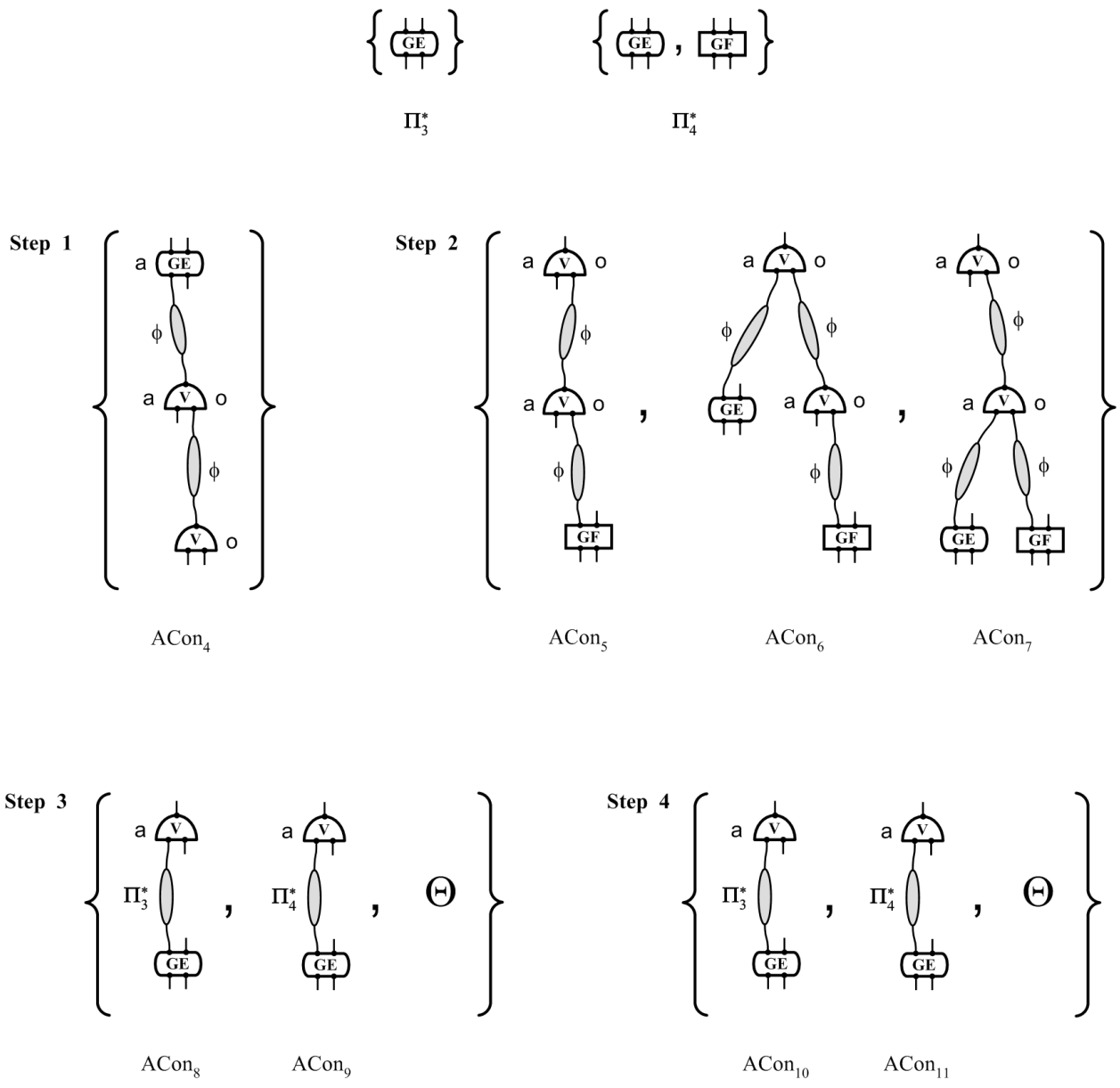


Figure 38: Pictorial description of level 0 generating system for class **Final Leg**. This description is comprised of four steps by the generating system. For simplicity, the steps by the environment are not modeled. Each step is specified by the depicted set of level 0 active constraints. If a primitive doesn't have labels on some of its primal processes, it denotes a *family* of all (admissible) primitives containing only those primitives whose primal processes with *shown* labels are fixed while other primal processes can vary. In this manner one specifies the subset of admissible primitives from the corresponding primitive class (see Appendix).

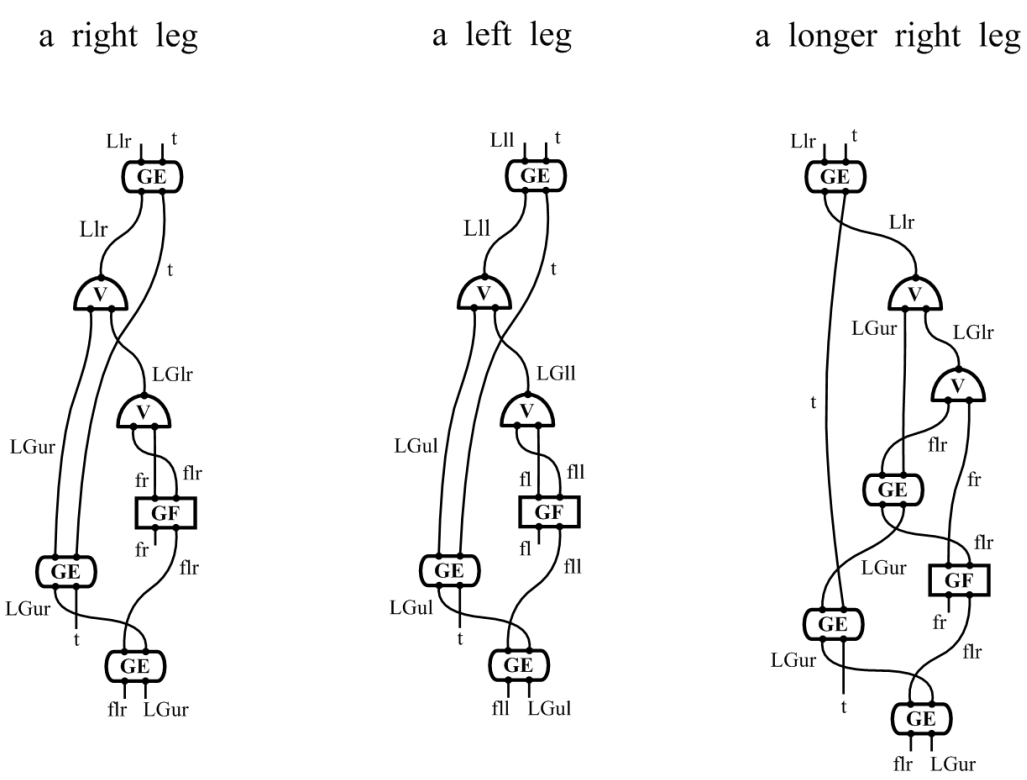
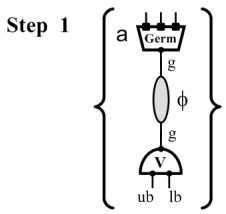


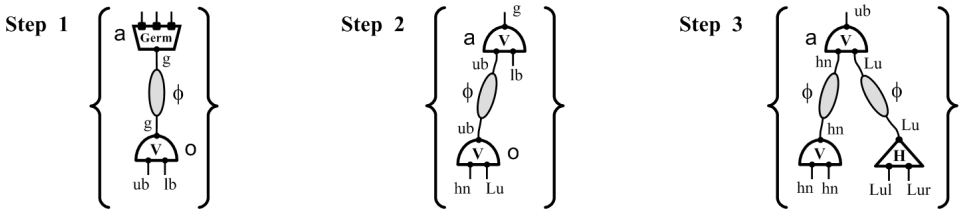
Figure 38 (continued): Two examples of level 0 structs generated by the above generating system.

"Initial Division" Class Generating System



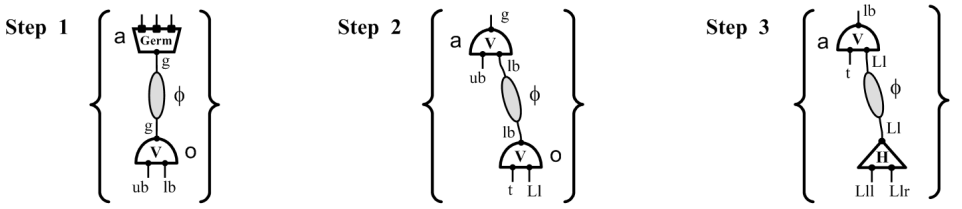
ACon_{12}

"Upper Part" Class Generating System



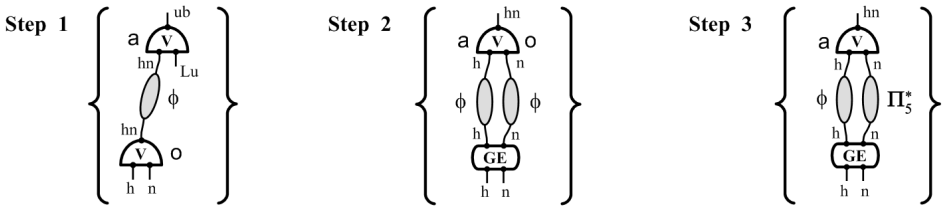
ACon_{13} ACon_{14} ACon_{15}

"Lower Part" Class Generating System



ACon_{16} ACon_{17} ACon_{18}

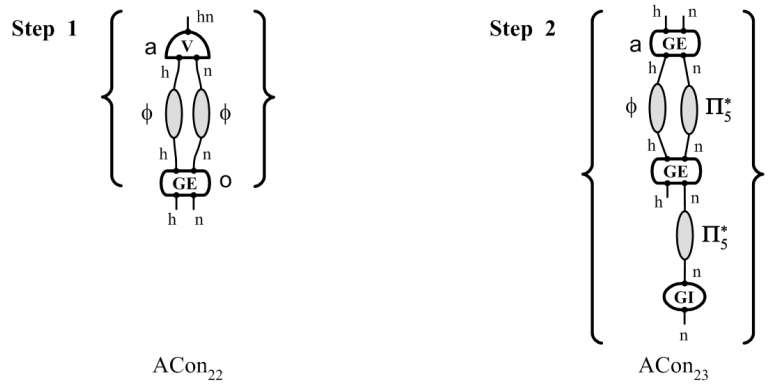
"Head" Class Generating System



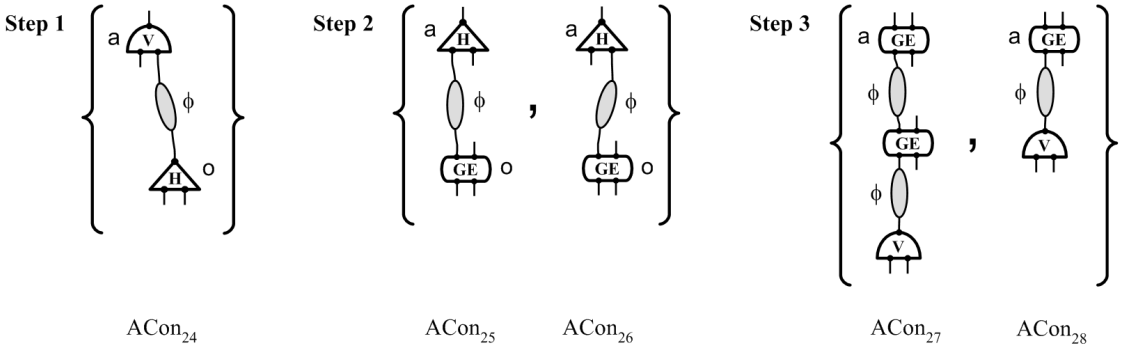
ACon_{19} ACon_{20} ACon_{21}

Figure 39: Stepwise specification (via sets of active constraints) of the remaining six level 0 class generating systems. The set $\Pi_5^* = \{\,\mathrm{GE}\,\}$. Steps by the environment are omitted.

"Neck" Class Generating System



"Proto Limb" Class Generating System



"Final Arm" Class Generating System

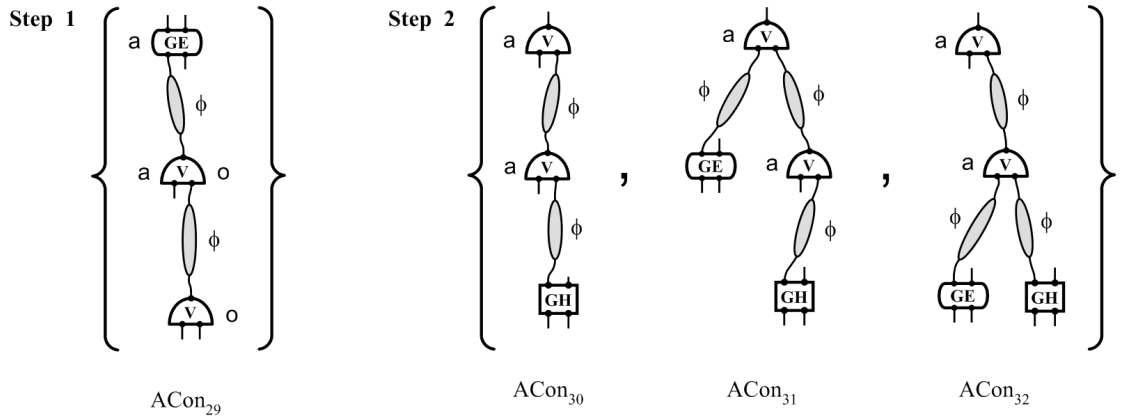


Figure 41 (continued)

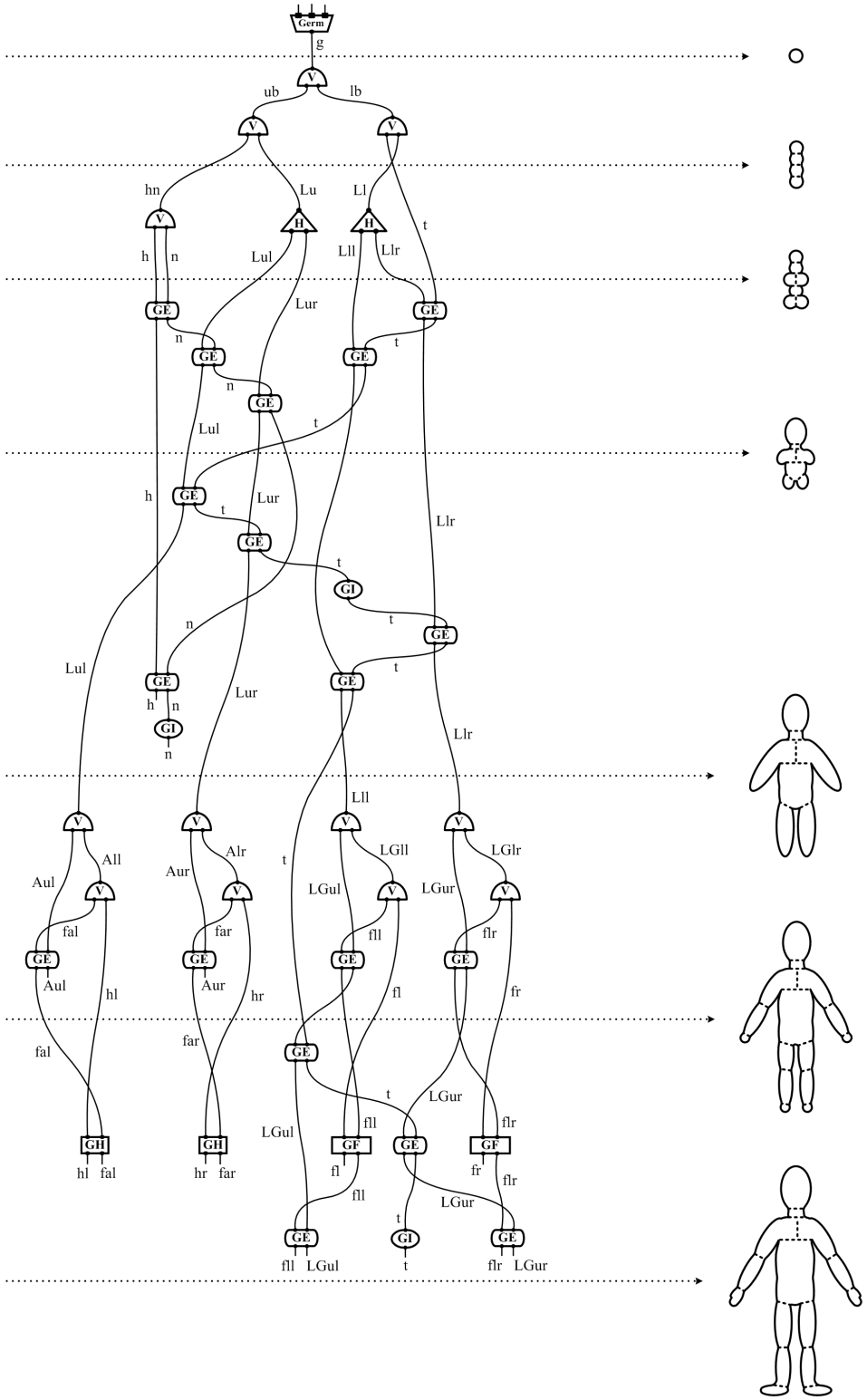


Figure 40: *Left*: A level 0 abstract struct from the Bubble Man class. All primal processes are labeled by the corresponding "sub-class" names. *Right*: A sequence of *stylized spatial* instantiations of the indicated (by dotted lines) level 0 substructs of the struct on the left (in order to illustrate the translation of the temporal event-based representation into a spatial one).

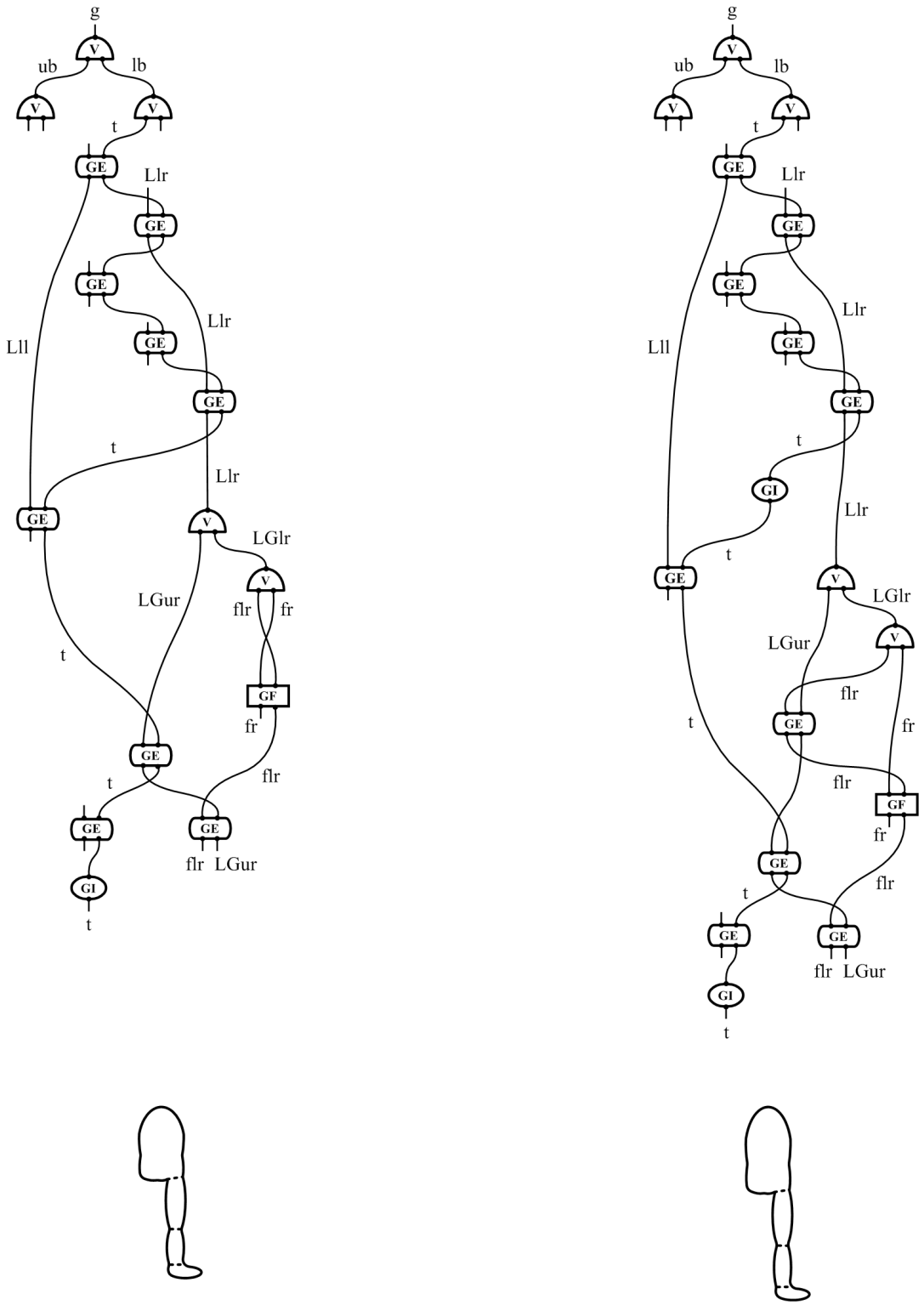


Figure 41: Two substructs of two different level 0 abstract structs from the Bubble Man class: one substruct represents a torso together with the leg and the other a larger torso with a larger leg.

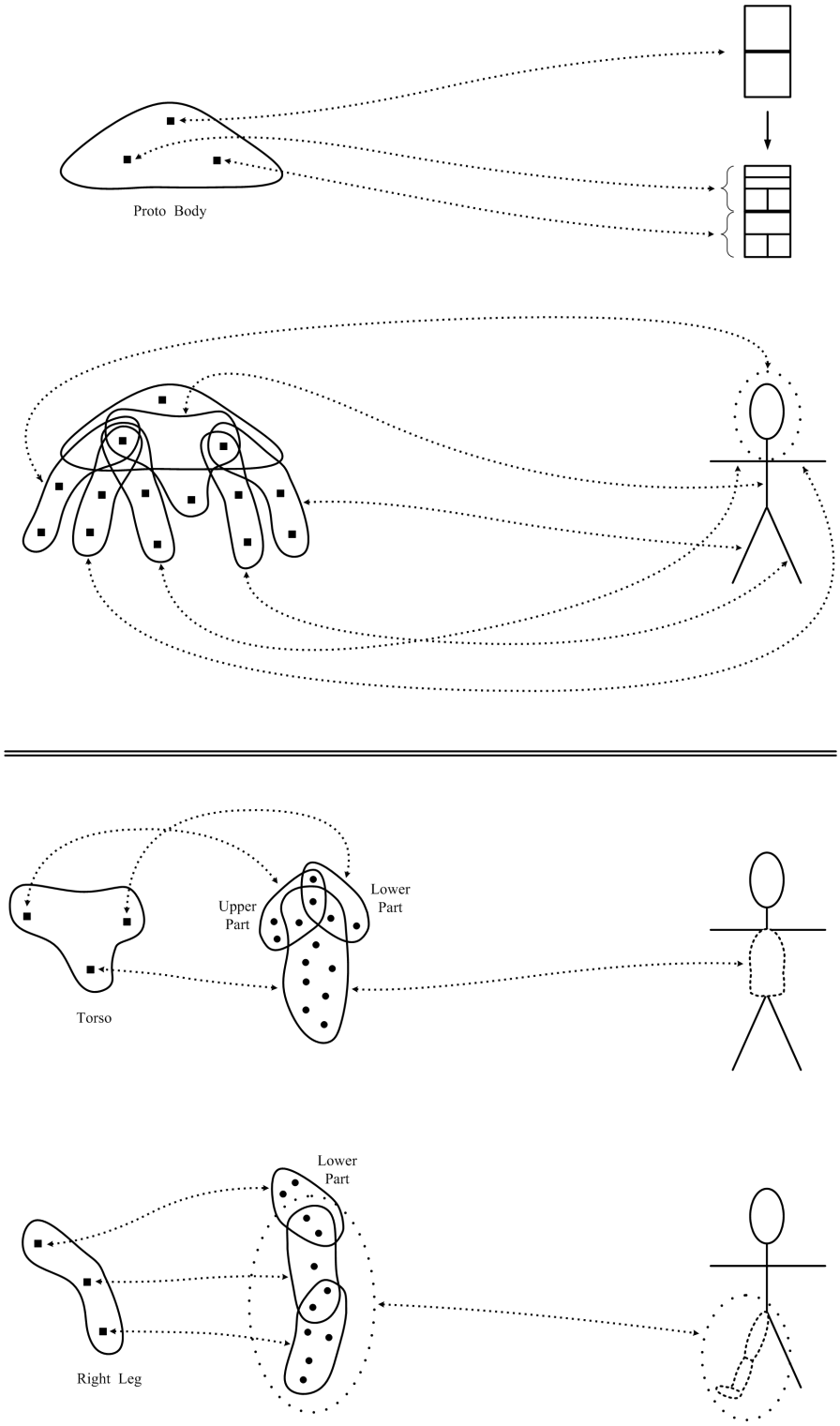


Figure 42: Illustration of the top two levels of abstraction in the Bubble Man class generating system. This particular form of hierarchical representation, novel to Biology, hints at the possibility of a similar (obviously more sophisticated) biological system *actually guiding* biological developmental processes.

Some general remarks on the organization of actual biological and physical processes are in order. First, in a biological situation, there would be many more level 0 classes, especially at later points in development. Also, in a biological or physical generating system, one should be more careful when deciding to skip a step (our null constraint Θ), since this particular algorithmic view is not quite "realistic".

In our example, one can observe the general idea that higher levels are more protected, or, more "rigid", and hence less amenable to change, compared to the lower levels which are "cheaper" for the generating system to modify.

Finally, it is instructive to compare a "biological" generating system with a typical non-biological one: the former is expected to output quickly widening structs, while the latter outputs structs whose width is more or less constant (e.g., the hydrogen or lithium structs shown in Introduction). Thus, physical processes tend to be relatively simple and periodic, while biological processes have more levels, since they need to *organize* the underlying physical processes in complex ways.

Part IV
Transformations and stages

> Our understanding of the world is built up of innumerable layers. Each layer is worth exploring, as long as we do not forget that it is one of many. Knowing all there is to know about one layer—a most unlikely event—would not teach us much about the rest.
>
> E. Chargaff, Heraclitean Fire: Sketches from a Life before Nature, 1978

9 Transformations

In this section, we introduce several additional central concepts, i.e. those of abstract and concrete transformations, which, in particular, are supposed to clarify those of abstract and concrete primitives. As was mentioned in the Introduction, by a 'transformation' we mean a macroevent that encapsulates the metamorphosis of one set of interacting structural processes into another set of processes. This macroevent signals the disruption of the previously stable flow of events associated with the corresponding structural processes, due to *new*, "disruptive" interactions of the latter processes, i.e. those not *already* a part of their regular flows. Thus, a transformation entrenches the nature of a particular (and possibly recurring) *pattern of interactions for the 'initial' structural processes which results in modified 'terminal' structural processes*: we say that the result of the interaction of the initial class generating systems is a set of terminal class generating systems. An abstract transform, then, can be viewed as an apparatus responsible for generating/modifing classes out of existing ones. In view of the unresolved complexity of the transform concept, the exposition in this section is more sketchy.

It might be useful to note that the concept of a transformation can also be considered as a far-reaching generalization of the concept of an element of an (algebraic) transformation group, where such an element is supposed to represent one of the transformations preserving object invariance, and the whole transformation group is supposed to capture the various symmetries of the object.

Definition 26. In what follows we assume that, at this **(basic or $0^{\mathbf{th}}$) stage of representation**, we have constructed classes of up to level r, culminating in the following **(basic) stage class setting**

$$\boldsymbol{\mathcal{C}} \;=\; \bigcup_{i=1}^{r} \mathcal{C}^i \,.$$

We set $m' = |\boldsymbol{\mathcal{C}}|$ and, for simplicity, when referring to a class $\mathfrak{C}^i_j$ from $\boldsymbol{\mathcal{C}}$, we will drop its level index i (keeping in mind that classes, their elements, etc. have implicit indices referring to levels). ▶

Before introducing transformations, we need the following notations.

Notational convention 3. A p-tuple of classes selected from $\boldsymbol{\mathcal{C}}$

$$\langle\, \mathfrak{C}_{i_1}\,,\, \mathfrak{C}_{i_2}\,,\, \ldots\,,\, \mathfrak{C}_{i_p}\,\rangle$$

is called a **tuple of classes** (TC). Moreover, a p-tuple

$$\langle\, \mathfrak{c}_{i_1}\,,\, \mathfrak{c}_{i_2}\,,\, \ldots\,,\, \mathfrak{c}_{i_p}\,\rangle\,,$$

such that

$$\mathfrak{c}_{i_j} \in \mathfrak{C}_{i_j} \quad \text{and} \quad \mathfrak{c}_{i_j} \neq \mathfrak{c}_{i_k} \;\text{ for }\; j \neq k\,,$$

is called a **tuple of class elements** (TCE) associated with TC.

We will need to distinguish between *two kinds of tuples* of classes, which we refer to as an **initial tuple of classes** (ITC) and **terminal tuple of classes** (TTC), and the corresponding tuples of class elements (referred to as ITCE and TTCE, respectively).

Given a TC, the corresponding **set of all tuples of class elements** is denoted $\mathbf{TCE}_{\,\mathrm{TC}}$, $\mathbf{TCE}_{\,\mathrm{TC}} \subseteq \mathfrak{C}_{i_1} \times \ldots \times \mathfrak{C}_{i_p}$. ◗

Moving on to the central definition in this section and relying on the terminology of Notational Convention 3, we note that the following concept of a transformation in fact *designates* a particular kind of interaction[48] *between* the constituent processes of ITC, resulting in the generation of the terminal tuple of processes TTC.

Definition 27. For a given **abstract transformation name** $\widehat{\boldsymbol{\tau}}$, the corresponding **abstract transformation** of classes (from $\boldsymbol{\mathcal{C}}$), or simply **abstract transform**, is an enumerable set $\boldsymbol{\tau}$ [49]

$$\boldsymbol{\tau} \;=\; \Big\{\, \boldsymbol{\tau}(\mathrm{ITCE}) \;\Big|\; \mathrm{ITCE} \in \mathbf{TCE}_{\,\mathrm{ITC}(\widehat{\boldsymbol{\tau}})} \,\Big\}$$

whose generic element $\boldsymbol{\tau}(\mathrm{ITCE})$, called a *corresponding* (concrete) **transform**, is defined as follows:[50]

for each ITCE from $\mathbf{TCE}_{\,\mathrm{ITC}(\widehat{\boldsymbol{\tau}})}$ the corresponding transform is defined as[51]

$$\boldsymbol{\tau}(\mathrm{ITCE}) \;=\; \tau_{\mathrm{ITCE}} \;\stackrel{\mathrm{def}}{=}\; \big\langle\, \widehat{\boldsymbol{\tau}}\,,\, \mathrm{ITC}(\widehat{\boldsymbol{\tau}})\,,\, \mathrm{TTC}(\widehat{\boldsymbol{\tau}})\,,\, \mathrm{ITCE}\,\big\rangle\,,$$

where

[48] At present, given our limited applied experience, we do not propose any classification of these interactions (into concrete categories).

[49] Note that, here, the set $\mathbf{TCE}_{\,\mathrm{ITC}(\boldsymbol{\tau})}$ plays a role analogous to that of the set of labels $\mathcal{L}$ associated with a primitive $\boldsymbol{\pi}$ (see Def. 1). The latter important analogy will be exploited in the next section, when transforms are replaced by next-level primitives (see mapping $tabst_{\,i\text{-labl}}$ in Def. 29).

[50] As in Def. 1, both of the following notations, i.e. $\boldsymbol{\tau}(\mathrm{ITCE})$ and τ_{ITCE}, will be used.

[51] For simplicity, the hat accent on $\widehat{\boldsymbol{\tau}}$ is dropped when it is used as a lower index.

$\mathrm{ITC}(\widehat{\boldsymbol{\tau}}) = \langle\, \mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau},1}\,, \ldots, \mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau},p}\,\rangle$	$\mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau},i}$ is the (given) $i^{\,\mathbf{th}}$ **initial class for** $\widehat{\boldsymbol{\tau}}$, $p>0$
$\mathrm{ITCE} = \langle\, \overline{\mathfrak{c}_1}\,, \ldots, \overline{\mathfrak{c}_p}\,\rangle$	$\overline{\mathfrak{c}_i}$ is the $i^{\,\mathbf{th}}$ **initial class element** for $\boldsymbol{\tau}(\mathrm{ITCE})$, $\overline{\mathfrak{c}_i} \in \mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau},i}$
$\mathrm{TTC}(\widehat{\boldsymbol{\tau}}) = \langle\, \mathfrak{C}^{\mathrm{Term}}_{\boldsymbol{\tau},1}\,, \ldots, \mathfrak{C}^{\mathrm{Term}}_{\boldsymbol{\tau},q}\,\rangle$	$\mathfrak{C}^{\mathrm{Term}}_{\boldsymbol{\tau},j}$ is the (given) $j^{\,\mathbf{th}}$ **terminal class for** $\widehat{\boldsymbol{\tau}}$.

We denote by $\mathbf{T}_{\boldsymbol{\mathcal{C}}}$, or simply $\mathbf{T}$, the set of all abstract transforms associated with $\boldsymbol{\mathcal{C}}$, and by $\mathrm{T}_{\boldsymbol{\mathcal{C}}}$, or simply T, the set of all corresponding (concrete) transforms. ▶

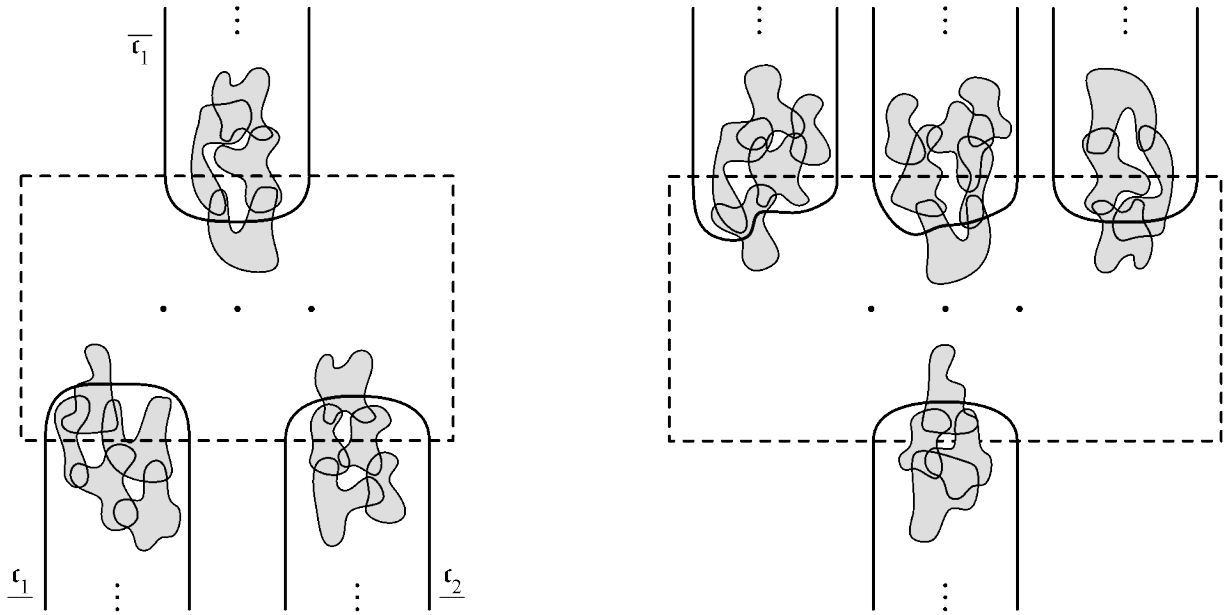


Figure 43: An illustration of two concrete transforms in which the structural processes are only partially shown.

It is useful to note that a convenient view of the concrete transform is one that takes initial *class system instances* and outputs terminal *class system instances*.

Remark 7 (on the relationship between transforms and the classes involved). When observing two transforms directly "connected" by some class element, it is useful to note that, if a particular initial class element of the top transform is identical to the connecting class element, this means that the top transform simply "regenerated" the corresponding process. Note that, if an initial process is observed "passing through" a transform, i.e. non-interacting with other initial processes, then, when identifying the transform, this process *should not be treated as part of the transform.* ◗

We are now ready to introduce another (radical and unique) feature of the proposed formalism, the *natural* emergence of the next stage of representation. Which of the structural elements of the ETS formalism allows this? As suggested in this section, *processes produced by various class generating systems can be subject to various transformations*, thus leading

naturally to a more global hierarchy, introduced next, of structural processes and transforms. The latter hierarchy is supposed to correspond to the more conventional hierarchy of processes in nature.

10 Multi-stage inductive structures

In this section, we shall see why the ETS formalism allows a natural transition to the next *stage* of representation. Such a transition consists of the construction of a new (next-stage) set of primitives, which can then be used to introduce next-stage structs, transforms, etc. in a manner similar to that outlined above in this paper.

As was mentioned in the introduction to Part III, one should not confuse levels with the stages introduced in this Section.

Stage ascension principle. The pattern of class interactions encapsulated by an abstract transform may be adequately captured at the next representational stage by a new abstract primitive whose initials and terminals are obtained by suppressing the structure of the transform's initial and terminal classes (in the manner described below, in Def. 29). ▶

Notational convention 4. From this section onwards, notations associated with the next-stage will be marked by the addition of a prime ($'$) to the corresponding present-stage notation. Moreover, the *lower (subscript) index of a mapping name* refers to the codomain of the mapping. ◗

We now consider the situation in which the agent has already identified (i.e. constructed) a set of transformations relevant to the present observations.

Definition 28. Having identified a set $\boldsymbol{\mathcal{C}} = \{\mathfrak{C}_1\,,\,\mathfrak{C}_2\,,\,\ldots\,,\,\mathfrak{C}_{m'}\}$ of classes in Def. 26, we further assume that a set $\mathbb{TS}_{\boldsymbol{\mathcal{C}}}$, or simply $\mathbb{TS}$, of abstract transforms,

$$\mathbb{TS}_{\boldsymbol{\mathcal{C}}} \;=\; \{\,\boldsymbol{\tau}_1,\,\boldsymbol{\tau}_2,\,\ldots,\,\boldsymbol{\tau}_{n'}\,\}\,,$$

called a **transformation system**, is also identified. Recall that a generic element $\boldsymbol{\tau}_i(\mathrm{ITCE})$ of $\boldsymbol{\tau}_i$ is

$$\boldsymbol{\tau}_i(\mathrm{ITCE}) \;=\; \big\langle\, \widehat{\boldsymbol{\tau}}_i\,,\,\mathrm{ITC}(\widehat{\boldsymbol{\tau}}_i)\,,\,\mathrm{TTC}(\widehat{\boldsymbol{\tau}}_i)\,,\,\mathrm{ITCE}\,\big\rangle\,,$$

where

$$\begin{aligned}\mathrm{ITC}(\widehat{\boldsymbol{\tau}}_i) &= \langle\, \mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau}_i,1}\,,\,\ldots\,,\,\mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau}_i,p(i)}\,\rangle\\ \mathrm{TTC}(\widehat{\boldsymbol{\tau}}_i) &= \langle\, \mathfrak{C}^{\mathrm{Term}}_{\boldsymbol{\tau}_i,1}\,,\,\ldots\,,\,\mathfrak{C}^{\mathrm{Term}}_{\boldsymbol{\tau}_i,q(i)}\,\rangle\,.\end{aligned}$$

▶

The following definition introduces the concept of next-stage primitive transformation, both abstract and concrete. (Since the structure of the next definition follows that of Def. 1, one might find it useful to review it now.)

Definition 29. For the above transformation system $\mathbb{TS}_{\boldsymbol{\mathcal{C}}}$, the set $\mathbb{C}'$ of **next-stage classes** is defined as follows:

$$\mathbb{C}' \stackrel{\text{def}}{=} \left\{ C'_1, C'_2, \ldots, C'_{m'} \right\},$$

where $|C'_i| = |\mathfrak{C}_i|$, $1 \leq i \leq m'$, i.e. we have a bijection

$$abst_{\,i\text{-class}} \colon \mathfrak{C}_i \to C'_i$$

which abstracts away the *structure* of each element in $\mathfrak{C}_i$. The set of **names of next-stage primitives** is:

$$\widehat{\boldsymbol{\Pi}'} \stackrel{\text{def}}{=} \left\{ \widehat{\boldsymbol{\tau}_1}, \widehat{\boldsymbol{\tau}_2}, \ldots, \widehat{\boldsymbol{\tau}_{m'}} \right\},$$

i.e. $\widehat{\boldsymbol{\pi}'_i} = \widehat{\boldsymbol{\tau}_i}$. As in Def. 1, before defining $\boldsymbol{\pi}'_i$, we need to introduce the following three concepts. For each $\widehat{\boldsymbol{\pi}'_i}$, define the **tuple of initial next-stage classes**

$$\mathrm{Init}(\widehat{\boldsymbol{\pi}'_i}) \stackrel{\text{def}}{=} \left\langle C'_{j_1}, C'_{j_2}, \ldots, C'_{j_{p(i)}} \right\rangle,$$

where

$$C'_{j_k} = abst_{\,i\text{-class}}\left(\mathfrak{C}^{\mathrm{Init}}_{\boldsymbol{\tau}_i, j_k} \right) \qquad 1 \leq k \leq p(i).$$

Then, the **set of next-stage labels associated with** $\boldsymbol{\pi}'_i$ is

$$\mathcal{L}'_i \stackrel{\text{def}}{=} C'_{j_1} \times C'_{j_2} \times \ldots \times C'_{j_{p(i)}}$$

and we have a bijection

$$tabst_{\,i\text{-labl}} \colon \mathbf{TCE}_{\,\mathrm{ITC}(\widehat{\boldsymbol{\tau}_i})} \to \mathcal{L}'_i$$
$$tabst_{\,i\text{-labl}} \stackrel{\text{def}}{=} abst_{\,j_1\text{-class}} \times abst_{\,j_2\text{-class}} \times \ldots \times abst_{\,j_{p(i)}\text{-class}},$$

hence the name "tuple abstraction" (see Notational Convention 3 and Def. 27), which is defined as a product of mappings. The concept of $\mathrm{Term}(\widehat{\boldsymbol{\pi}'_i})$ is defined in a manner similar to $\mathrm{Init}(\widehat{\boldsymbol{\pi}'_i})$.

Define $\boldsymbol{\pi}'_i$ as a set

$$\boldsymbol{\pi}'_i = \left\{ \boldsymbol{\pi}'_i(\boldsymbol{a}') \mid \boldsymbol{a}' \in \mathcal{L}'_i \right\}$$

whose generic element $\boldsymbol{\pi}'_i(\boldsymbol{a}')$ is:[52]

$$\forall\, \boldsymbol{a}' \in \mathcal{L}'_i \qquad \boldsymbol{\pi}'_i(\boldsymbol{a}') = \pi'_{i\boldsymbol{a}'} \stackrel{\text{def}}{=} \left\langle \widehat{\boldsymbol{\pi}'_i}, \mathrm{Init}(\widehat{\boldsymbol{\pi}'_i}), \mathrm{Term}(\widehat{\boldsymbol{\pi}'_i}), \boldsymbol{a}' \right\rangle.$$

see Fig. 44. Set $\boldsymbol{\pi}'_i$ is called a **next-stage abstract primitive transformation**, or simply **next-stage abstract primitive**, and its generic element $\pi'_{i\boldsymbol{a}'}$ is called a **corresponding** (concrete) **next-stage primitive**. We denote by $\boldsymbol{\Pi}'$ the finite set of all next-stage abstract primitives $\boldsymbol{\pi}'_i$, $1 \leq i \leq n$ and by Π' the set of all (concrete) next-stage primitives. ▶

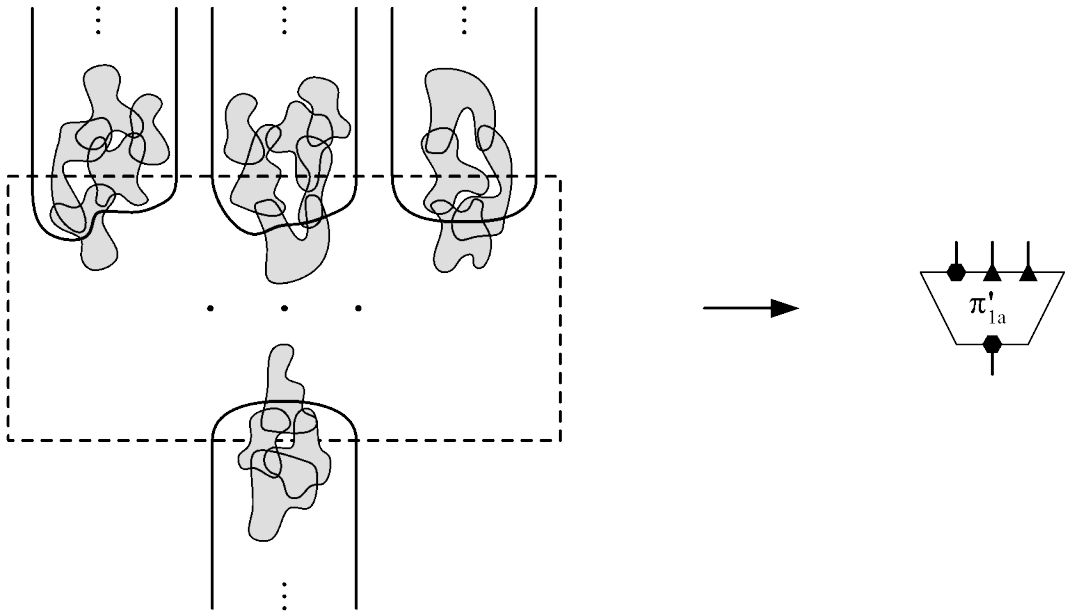


Figure 44: A transform (left) and the corresponding next-stage primitive (right).

Since the structure of a next-stage primitive is absolutely identical to that of the initial-stage primitive, all remaining initial-stage concepts are immediately applicable.

Definition 30. Next-stage analogues of definitions 2 through 27 follow immediately. ▶

We are now ready to state a formal version of the above stage ascension principle, which might be viewed as a convenient encapsulation of the main concepts in Def. 29.

Definition 31. For the above transformation system $\mathbb{TS}_{\boldsymbol{\mathcal{C}}}$,

$$\mathbb{TS}_{\boldsymbol{\mathcal{C}}} \;=\; \{\, \boldsymbol{\tau}_1,\, \boldsymbol{\tau}_2,\, \ldots,\, \boldsymbol{\tau}_{n'} \,\},$$

the corresponding **stage transition mappings** (between consecutive stages) are the following three bijective mappings:

(i)
$$\boldsymbol{stage}_{\,i\text{-class}} \colon \mathfrak{C}_i \to C'_i \qquad\qquad 1 \le i \le m'$$
$$\boldsymbol{stage}_{\,i\text{-class}} \stackrel{\text{def}}{=} \mathit{abst}_{\,i\text{-class}}\,,$$

(ii)
$$\boldsymbol{stage}_{\text{a-prim}} \colon \mathbb{TS}_{\boldsymbol{\mathcal{C}}} \to \boldsymbol{\Pi}'$$
$$\boldsymbol{stage}_{\text{a-prim}}\big(\,\boldsymbol{\tau}_i\,\big) \stackrel{\text{def}}{=} \boldsymbol{\pi}'_i\,,$$

(iii)
$$\boldsymbol{stage}_{\,i\text{-lab}} \colon \mathbf{TCE}_{\,\mathrm{ITC}(\widehat{\boldsymbol{\tau}_i})} \to \mathcal{L}'_i$$
$$\boldsymbol{stage}_{\,i\text{-lab}} \stackrel{\text{def}}{=} \mathit{tabst}_{\,i\text{-labl}}\,.$$

[52] As in Def. 1, both of the following notations, i.e. $\boldsymbol{\pi}'_i(a')$ and $\pi'_{ia'}$, will be used.

▶

It is not difficult to see how the above consecutive stage correspondence can be extended to include the following mapping from concrete transforms to concrete primitives:

$$\begin{aligned} \boldsymbol{stage}_{\text{c-prim}} &: \mathrm{T} \to \Pi' \\ \boldsymbol{stage}_{\text{c-prim}}\big(\,\boldsymbol{\tau}_i(\mathrm{ITCE})\,\big) &= \boldsymbol{\pi}'_i(a')\,, \end{aligned}$$

where

$$\boldsymbol{stage}_{\text{a-prim}}\big(\,\boldsymbol{\tau}_i\,\big) = \boldsymbol{\pi}'_i \qquad\qquad \boldsymbol{stage}_{\,i\text{-lab}}\big(\,\mathrm{ITCE}\,\big) = a'\,.$$

Finally, we can encapsulate the entire developed mathematical structure as a single concept in the following definition. Note that we substitute the arrow $\nearrow$ for the various ***stage***-mappings.

Definition 32. A **(single-stage) inductive structure** is a pair

$$\langle\, \boldsymbol{\Pi}\,,\, \mathbb{TS}_{\boldsymbol{\mathcal{C}}}\,\rangle\,,$$

where $\boldsymbol{\Pi}$ is a set of abstract primitives and $\mathbb{TS}_{\boldsymbol{\mathcal{C}}}$ is a transformation system. However this pair will also encompass *all relevant concepts*, such as structs and classes at various levels, transforms, etc.

A **multi-stage**, or more precisely an l-stage, **inductive structure** $\mathcal{MIS}$ is an l-tuple[53]

$$\mathcal{MIS} \;=\; \big\langle\, \langle\, \boldsymbol{\Pi}\,,\, \mathbb{TS}\,\rangle\,,\, \langle\, \boldsymbol{\Pi}'\,,\, \mathbb{TS}'\,\rangle\,,\, \ldots\,,\, \langle\, \boldsymbol{\Pi}^{(l-1)}\,,\, \mathbb{TS}^{(l-1)}\,\rangle\,\big\rangle\,,$$

where $\mathbb{TS}^{(l-1)} = \varnothing$, $\langle\, \boldsymbol{\Pi}^{(k)}\,,\, \mathbb{TS}^{(k)}\,\rangle$ is the k^{th} stage inductive structure, and the transition between two consecutive inductive structures is accomplished in the manner outlined above in this section (see Fig. 45). For the k^{th} stage inductive structure in $\mathcal{MIS}$ we will use the notation

$$\begin{aligned} \mathcal{MIS}(k) \;&=\; \big\langle\, \boldsymbol{\Pi}^{(k)}\,,\, \mathbb{TS}^{(k)}\,\big\rangle && k = 0, 1 \ldots, l-1 \\ \mathfrak{C}_i^{(k)} \;&\nearrow\; C_i^{(k+1)} && k = 0, 1 \ldots, l-2 \\ \boldsymbol{\tau}^{(k)} \;&\nearrow\; \boldsymbol{\pi}^{(k+1)} && k = 0, 1 \ldots, l-2 \\ \mathrm{ITCE}^{(k)} \;&\nearrow\; a^{(k+1)} && k = 0, 1 \ldots, l-2 \\ \boldsymbol{\tau}^{(k)}(\mathrm{ITCE}^{(k)}) \;\nearrow\; \boldsymbol{\pi}^{(k+1)}(a^{(k+1)}) \qquad \text{or} \qquad \tau^{(k)}_{\mathrm{ITCE}^{(k)}} \;&\nearrow\; \pi^{(k+1)}_{a^{(k+1)}} && k = 0, 1 \ldots, l-2\,, \end{aligned}$$

where the arrow $\nearrow$ stands for the appropriate ***stage***-mapping. ▶

[53] For simplicity, we drop the index $\boldsymbol{\mathcal{C}}^{(k)}$ from $\mathbb{TS}^{(k)}_{\boldsymbol{\mathcal{C}}^{(k)}}$.

In general, we expect that as stages emerge, the lower stages gradually rigidify, i.e. stabilize, and do so at a faster pace than the upper stages.

Finally, it is important to note that, for example, a class identified at stage 3 is not readily identified at stage 1 even though it also "exists" at that stage, hence the main role of stages.

[Important] **Remark 8.** As to the choice of the basic representational stage (stage 0)—and thus the stage 0 primitives—it is useful to make sure that most stage 0 classes persist, i.e. *can be reliably observed between transforms*, for a reasonable period of time. Otherwise, the constructed multi-stage inductive structure is unreliable. This consideration should guide the evaluation of the quality of the initial representation stage. ◗

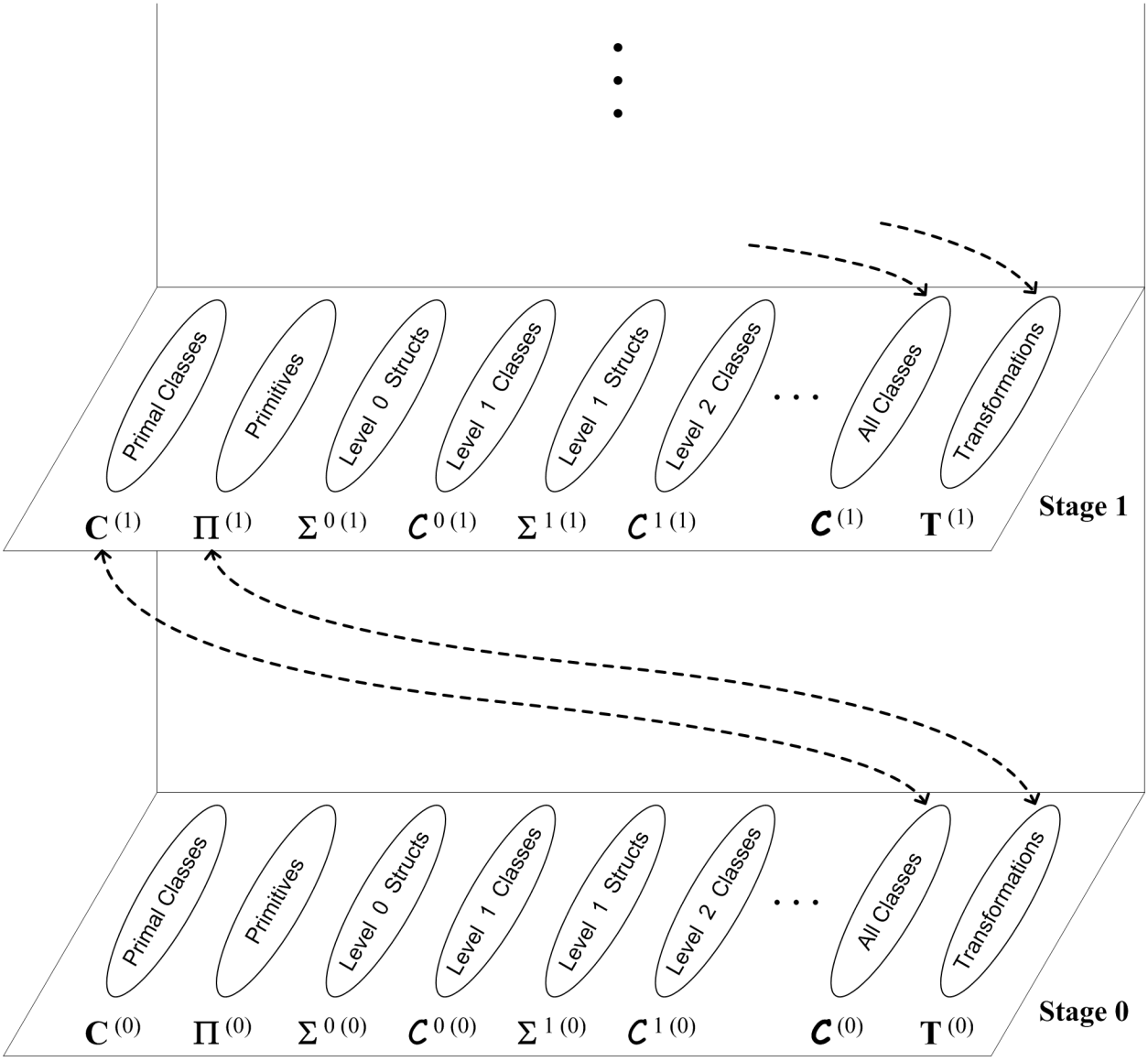


Figure 45: Schematic representation of a multi-stage inductive structure.

Part V
Conclusion

> I do admit that at any moment we are prisoners caught in the framework of our theories; our expectations; our past experiences; our language. But we are prisoners in a Pickwickian sense: if we try we can break out of our framework at any time. Admittedly, we shall find ourselves again in a framework, but it will be a better and roomier one; and we can at any moment break out of it again.
>
> .....
>
> The Myth of the Framework is, in our time, the central bulwark of irrationalism. My counter-thesis is that it simply exaggerates a difficulty into an impossibility. The difficulty of discussion between people brought up in different frameworks is to be admitted. *But nothing is more fruitful than such a discussion; than the culture clash which has stimulated some of the greatest intellectual revolutions.*
>
> Karl Popper, Normal science and its dangers, 1970 (our emphasis)

11 Preliminary general thoughts on learning

In this section, we briefly discuss our preliminary ideas regarding (inductive) learning. In general, we strongly believe that, from an applied point of view and for the near future at least, the emphasis in learning should be less on the statistical activity *called* learning and more on the (structural) representational issues, as follows from Parts III and IV. In the proposed framework, it should not come as a surprise that learning is, in fact, learning of classes and transforms.

As far as the learning of classes is concerned, we mean, of course, the learning of classes of structural entities (both initial and terminal). This learning involves the construction of application-specific class generating systems, which in turn involve some (structural) optimization process, based on a training set [54]. Consider, for example, the case when one is faced with learning a three-level class representation, and assume that the necessary constituent classes at levels 0 and 1 have previously been learned. Then, the corresponding optimization process is related to the selection of optimal constraints $\left\{ \mathrm{Con}^{2}_{j,i}(\mathscr{C}^{1}_{j,i}) \right\}_{i\in I}$, where optimality is understood in terms of "structural simplicity". An appropriate consistency measure should indicate when it is impossible to do so, i.e. when the available constituent classes are inadequate, or insufficient. Note that, to learn a three-level class, the training elements must be represented as level 2 structs (which, in turn, require the relevant level 1 classes, level 1 structs, and so on).

Next, once almost all of the identified/learned class elements maintain their integrity, or class identity, or autonomy, for a sufficiently long period of time, we call them *structural pro-*

[54] *Ideally*, by a training set, we mean a non-conventional analogue of the traditional concept, where the representation of each object is the result of a sophisticated dynamic process of interaction between an observer and the target object/process—mediated by *structural* (as opposed to numeric) sensors—which delivers structs. However, in the immediate future, before such structural event sensors are available, it is sufficient for developers to intuit (and postulate) such a dynamic process.

cesses for that representational stage, at which point one can proceed with the identification of transforms, followed by a transition to the next stage.

Keeping in mind that a transform encapsulates the pattern of interaction of several structural processes, one can first identify transforms (relying on the initial and terminal class representations) and then, if necessary, modify the corresponding class representations if the "transform evidence" suggests their inadequacy.

As far as computational complexity issues are concerned, we note that, *computationally*, learning is bounded by the number of various restricted partitions of the training structs (each into an assembly of substructs). The computational cost associated with the construction of a candidate optimal partition appears to be low order polynomial, since, locally, a struct is "almost linear": although the "width" of an entire struct is not constant, the width (and also length) of the substructs are always bounded by a small number (since, for all primitives, the branching factor is bounded by the maximal number of terminals). In addition, with each new stage of representation, since the size of the next-stage struct shrinks by at least an order of magnitude as compared to its previous-stage counterpart, the complexity of learning at the new stage is also reduced. Recall that we have assumed (and this assumption seems to be borne out by reality) that, at each stage, the number of classes as well as the number of levels for various class generating systems within a stage is relatively small.

12 How one should approach the ETS formalism

This section was written to address the situation we often find ourselves in when presenting talks on the ETS formalism. At the end of a talk, instead of discussing the various features of the formalism, we are often faced with answering the following questions (common in PR and ML communities): "Did you compare the performance of your approach with that of neural networks (or some other popular model)?" We would like to stress that these types of questions are currently quite unproductive. Such questions presume that an evaluation of ETS should proceed under conventional assumptions, i.e. the *first thing* one should do is to compare the performance of ETS learning algorithms with other learning algorithms. Hoever, even a cursory reading of this paper should immediately suggest that the very formulation of the *inductive learning problem* has changed radically and thus "the first thing one should do" is to better understand what this new formulation offers, rather than focusing *prematurely* on non-central issues.

So, which new insights into the nature of the inductive learning problem does the ETS formalism offer? It turns out that the new formulation of this problem[55]can be stated as follows: given a small training set for a class, construct the corresponding *class representation*. As one can now see, for the first time, the concept of class representation comes to the fore. So what is a 'class representation'? Although a *hint* at the answer to this question was suggested by the concept of generative grammars (i.e. a set of production rules), as we discussed in Section 1.5, this answer is inadequate for a more profound reason: the lack of *representational* formalisms in science in general, and in mathematics in particular ([9]). Put simply, there is no adequate formal representational setting for strings as object representations,

[55] See also a quotation from Vapnik and Chervonenkis given in [69], online.

and therefore all inductive constructions based on such an unsound foundation cannot be effective. Thus, the new problem formulation inevitably focuses one's attention on the very profound issues of both class and object representations, since the former cannot be properly addressed without the latter. Again, as can be seen from this paper, the development of the underlying representational formalism turned out to be an enormous undertaking.

Returning to the issue of the *evaluation* of the ETS formalism, the immediate focus should be on answering the following kinds of scientific questions:

- Is the new (multi-stage struct) representation more useful and/or powerful as the primary form of representation? That is, given a concrete application, develop an intuition about struct representation in that setting, and then ask the question, "Do these structs contain additional, important information about the actual objects/processes of the application?"[56]

  If the struct concept, as defined in this paper, turns out to be somewhat inadequate as a primary form of representation, can it be "repaired" within the current overall framework, or does the framework itself have to be rethought?

- What is an appropriate generalization of the concepts of structural constraint (Def. 10)?

- Can the proposed concept of class representation fulfill its intended purpose? Is the proposed overall approach to the class generating system a sufficiently powerful way of describing, or representing, actual classes in the chosen applied setting? Is the link between a small training set and its corresponding class representation computationally effective? In fact, it should be clear that the introduced concept of multi-level struct substantially reduces the gap between representations of objects and their classes as it exists in the current formalisms.

- What are the options for introducing a more explicit concept of transform, aligned with the definition of class generating system (once the above choice of the definition of a class generating system is settled)?

- In particular, what is an appropriate answer to the intuitive notion of "adjacency" of structural processes in a transform?

- What are the options for addressing the specification of the generative mechanism in the definition of class representation? A closely related issue is the one mentioned in Important Remark 6 on p. 58, i.e. how do we adequately generalize the concept of primitive attachment to the constituent class elements in a higher-level struct?

During this initial period, we believe that the most productive way to approach these issues is by exploring in a concrete empirical setting the nature of the class generating system by building class representations for the corresponding training structs (from level 0 structs, to level 0 classes, to level 1 structs, level 1 classes, etc., eventually considering transforms).

[56] We believe that there is preliminary support for this claim, e.g. for the first time, an object's formative history is now a part of the representation.

Appendix

Primitive-class: a very useful generalization of the concept of primitive transformation

We introduce a relatively simple but extremely useful generalization of the concept of primitive transformation, which is presented in Definition 1. This generalization appears in this appendix since we realized its usefulness sometime after the paper was completed, when we began to explore various applications (see for example [70, 71]). Such simple generalization gives one greater freedom to choose more appropriate primal classes without being concerned about an *explosion in the number of primitives*, many of which turn out to have *practically* identical structure.

Another reason we decided not to replace Definition 1 with the following definition is that such generalization would complicate the already non-trivial (basic) concept of a primitive with auxiliary considerations, and in addition such modification would similarly complicate all the subsequent basic concepts and notations.

Definition 1. Suppose that among the introduced abstract primitives $\mathbf{\Pi}$, we have the following family $\mathbf{\Pi}_*$ of "closely-related" primitives:

$$\mathbf{\Pi}_* = \{\, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_t \,\},$$

where $\forall\, i$

$$\begin{aligned} \text{Init}(\boldsymbol{\pi}_i) \;&=\; \langle\, C^i_{j_1}\,,\, C^i_{j_2}\,,\, \ldots\,,\, C^i_{j_p}\,\rangle \\ \text{Term}(\boldsymbol{\pi}_i) \;&=\; \langle\, C^i_{k_1}\,,\, C^i_{k_2}\,,\, \ldots\,,\, C^i_{k_q}\,\rangle \end{aligned}$$

(i.e., all $\boldsymbol{\pi}_i$ have the same number of initial and terminal primal classes), and moreover,

$\forall\, r\ (\forall\, i_1, i_2\ \ (C^{i_1}_{j_r}$ and $C^{i_2}_{j_r}$ are "similarly-structured" classes whose representations are refinements of the representation of some class C_{j_r})), [57]

$\forall\, r\ (\forall\, i_1, i_2\ \ (C^{i_1}_{k_r}$ and $C^{k_2}_{k_r}$ are "similarly-structured" classes whose representations are refinements of the representation of some class C_{k_r})).

In this case, it is useful to treat family $\mathbf{\Pi}_*$ —which can be thought of as a *class of abstract primitives*—as a *single* **primitive-class**, denoted $[\boldsymbol{\pi}_*]$, i.e. $\mathbf{\Pi}_* = [\boldsymbol{\pi}_*]$, as the name of the class induced by the corresponding *equivalence relation* on $\mathbf{\Pi}$ (associated with *all* such disjoint classes of abstract primitives). ▶

To motivate the above definition, we present a simple but typical example from Part III. Figure 1 shows three primitive-classes and a corresponding "generalized" struct that relies on such primitive-classes, depicting a Final Torso class element. If instead of primitive-classes

[57] Unlike the above primal classes $C^i_{j_r}$, class C_{j_r} does not have to be one of the primal classes for $\mathbf{\Pi}$ (the same applies to the terminal classes). Given current prejudices, it might be convenient (although formally not quite correct) to think of $C^i_{j_r}$ as a "sub-class" of C_{j_r}.

one would rely on (regular) primitives only, then on the one hand, the separate introduction of *each* of the corresponding primal processes would necessitate (unjustifiably) a separate set of primitives, and on the other hand, would make representation of the Final Torso class very unwieldy, requiring (also quite unjustifiably) a corresponding substantial increase in the number of class constraints, which artificially obscures the actual state of affairs.

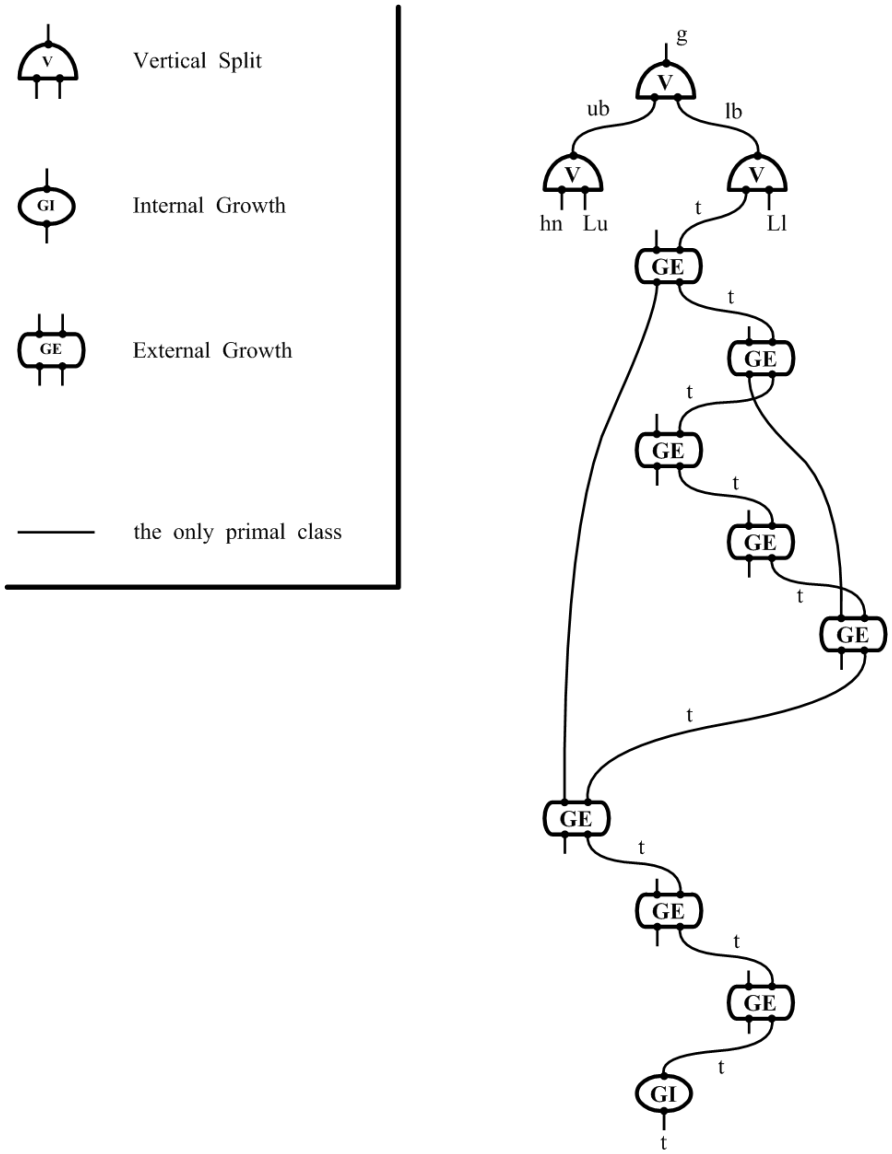


Figure 1: Three primitive-classes and an **abstract struct** (see Def. 7) relying on them. Note that we have only one primal class in our primitive-classes, which may be called the Basic Cell primal class, and the labels shown are those of the corresponding primal "sub-classes", see Figs. 31, 32 (there are no labels for concrete primal processes). Without the use of primitive-classes, for example, each instance of the GE primitive-class would have to appear as a distinct primitive (under its own distinct name, e.g. GE1, GE2, GE3, ...). The latter would have a complicating effect on the structs involved by obscuring their structural identity.

References

[1] L. Goldfarb, O. Golubitsky, D. Korkin, What is a structural representation?, Technical Report TR00-137, Faculty of Computer Science, UNB, 2000.

[2] L. Goldfarb, D. Gay, O. Golubitsky, D. Korkin, What is a structural representation? Second variation, Technical Report TR04-165, Faculty of Computer Science, UNB, 2004.

[3] L. Goldfarb, D. Gay, O. Golubitsky, What is a structural representation? Third variation, 2005 (unpublished).

[4] L. Goldfarb, D. Gay, O. Golubitsky, What is a structural representation? Fourth variation, Technical Report TR05-174, Faculty of Computer Science, UNB, July 2005.

[5] L. Goldfarb, D. Gay, What is a structural representation? Fifth variation, Technical Report TR05-175, Faculty of Computer Science, UNB, December 2005.

[6] G. Ifrah, *The Universal History of Numbers*, J. Wiley, New York, 2000.

[7] G. Sarton, *Ancient Science Through the Golden Age of Greece*, Dover, New York, 1993.

[8] E. Schrödinger, *Nature and the Greeks and Science and Humanism*, Cambridge University Press, Cambridge, 1996, pp. 143–145, 158.

[9] L. Goldfarb, Representational formalisms: what they are and why we haven't had one, this special issue, 2006.

[10] A. W. Crosby, *The Measure of Reality*, Cambridge University Press, 1997.

[11] L. Goldfarb, On the foundations of intelligent processes I: An evolving model for pattern learning, *Pattern Recognition* 23 (6), 1990, pp. 595–616.

[12] L. Goldfarb, What is distance and why do we need the metric model for pattern learning, *Pattern Recognition* 25 (4), 1992, pp. 431–438.

[13] L. Goldfarb, S. Nigam, The unified learning paradigm: A foundation for AI, in: V. Honavar, L. Uhr (eds.), *Artificial Intelligence and Neural Networks: Steps toward Principled Integration*, Academic Press, Boston, 1994, pp. 533–559.

[14] L. Goldfarb, J. Abela, V. C. Bhavsar, V. N. Kamat, Can a vector space based learning model discover inductive class generalization in a symbolic environment?, *Pattern Recognition Letters* 16 (7), 1995, pp. 719–726.

[15] L. Goldfarb, Inductive class representation and its central role in pattern recognition, *Proc. Conf. Intelligent Systems: A Semiotic Perspective*, Vol. 1, NIST, Gaithersburg, Maryland, USA, 1996, pp. 53–58.

[16] L. Goldfarb, What is inductive learning? Construction of inductive class representation, *Proc. Workshop "What Is Inductive Learning" in Conjunction with 11th Biennial Canadian AI Conf.*, 1996, pp. 9–21.

[17] L. Goldfarb, S. Deshpande, What is a symbolic measurement process?, *Proc. IEEE Conf. Systems, Man, and Cybernetics*, Vol. 5, Orlando, Florida, USA, 1997, pp. 4139–4145.

[18] L. Goldfarb, J. Hook, Why classical models for pattern recognition are not pattern recognition models, *Proc. Intern. Conf. On Advances in Pattern Recognition*, Plymouth, UK, 1998, pp. 405–414.

[19] K. S. Fu, *Syntactic Pattern Recognition and Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[20] M. A. Aiserman, Remarks on two problems connected with pattern recognition, in: S. Watanabe (ed.), *Methodologies of Pattern Recognition*, Academic Press, 1969, p. 1.

[21] S. Wermter, R. Sun, *Hybrid Neural Systems*, Springer-Verlag, Heidelberg, 2000.

[22] H. Bunke, A. Kandel (eds.), *Hybrid Methods in Pattern Recognition*, World Scientific, 2002.

[23] 3rd International Workshop on Hybrid Methods for Adaptive Systems, Oulu, Finland, July 2003.
http://adiret.cs.uni-magdeburg.de/∼nuernb/hmas2003/

[24] 4th International Workshop on Hybrid Methods for Adaptive Systems, Aachen, Germany, June 2004.
http://adiret.cs.uni-magdeburg.de/hmas2004/

[25] 2004 AAAI Fall Symposium Series Workshop on Compositional Connectionism in Cognitive Science, Washington, D.C., October 2004.
http://www.cs.wlu.edu/∼levy/aaai04/

[26] SRL2004 Workshop: Statistical Relational Learning and its Connections to Other Fields, Banff, Canada, July 2004.
http://www.cs.umd.edu/projects/srl2004/

[27] MRDM 2005: 4$^{\text{th}}$ Workshop on Multi-Relational Data Mining, Chicago, August 2005.
http://www-ai.ijs.si/SasoDzeroski/MRDM2005/

[28] Machine Learning Journal Special Issue on Multi-relational Data Mining and Statistical Relational Learning, 2005.
http://www.cs.kuleuven.ac.be/∼ml/mrdm-srl.html

[29] A. Bird, *Philosophy of Science*, McGill-Queen's University Press, Montreal, 1998.

[30] J. H. Holland, K. J. Holyoak, R. E. Nisbett, P. R. Thagard, *Induction*, MIT Press, Cambridge, Mass., 1986.

[31] J. Losee, *A Historical Introduction to the Philosophy of Science*, 3rd ed., Oxford University Press, Oxford, 1993.

[32] H. Margolis, *Patterns, Thinking, and Cognition*, University of Chicago Press, 1987, pp. 1, 3.

[33] E. G. H. Landau, *Foundations of Analysis*, Chelsea, New York, 1951.

[34] C. Lee, *Notes for Math 502*, 1998.
http://www.ms.uky.edu/∼lee/ma502/notes2/node7.html

[35] N. Bourbaki, The Architecture of Mathematics, *Amer. Math. Monthly*, 57 (4), 1950, pp. 221–232.

[36] J. Dieudonne, The Work of Nicholas Bourbaki, *Amer. Math. Monthly*, 77, 1970, pp. 134–145.

[37] N. Chomsky, *Knowledge of Language: Its Nature, Origin, and Use*, Praeger, New York, 1986, p. 12.

[38] M. Piattelli-Palmarini (ed.), *Language and Learning: The Debate between Jean Piaget and Noam Chomsky*, HUP, Cambridge, USA, 1980, pp. 100–103, 255–272.

[39] M. Leyton, *Symmetry, Causality, Mind*, MIT Press, Cambridge, Mass., 1992, p. 1–2.

[40] A. R. Lacey, *A Dictionary of Philosophy*, 3rd ed., Routledge, London, UK, 1996, p. 308.

[41] R. Dunbar, *The Trouble with Science*, Faber and Faber, London, UK, 1996, p. 17.

[42] O. Golubitsky, *On the generating process and the class typicality measure*, Technical Report TR02-151, Faculty of Computer Science, UNB, 2002.

[43] O. Golubitsky, *On the Formalization of the Evolving Transformation System Model*, Ph.D. thesis, Faculty of Computer Science, UNB, March 2004.

[44] D. Korkin, *A New Model for Molecular Representation and Classification: Formal Approach Based on the ETS Framework*, Ph.D. thesis, Faculty of Computer Science, UNB, 2003.

[45] D. Clement, *Information Retrieval via the ETS Model*, Master's thesis, Faculty of Computer Science, UNB, 2003.

[46] S. Falconer, D. Gay, L. Goldfarb, ETS representation of fairy tales, *Proc. ICPR 2004 Satellite Workshop on Pattern Representation and the Future of Pattern Recognition*, ed. L. Goldfarb, Cambridge, UK, August 2004.

[47] S. Falconer, *On the Evolving Transformation System Model Representation of Fairy Tales*, Master's thesis, Faculty of Computer Science, UNB, 2005.

[48] M. Al-Digeil, *Towards an Evolving Transformation System Representation of Proteins*, Master's thesis, Faculty of Computer Science, UNB, 2005.

[49] A. Gutkin, *Towards Formal Structural Representation of Spoken Language: An Evolving Transformation System (ETS) Approach*, Ph.D. thesis, School of Informatics, University of Edinburgh, 2005.

[50] J. M. Abela, *ETS Learning of Kernel Languages*, Ph.D. thesis, Faculty of Computer Science, UNB, 2002.

[51] V. N. Kamat, *Inductive Learning with the Evolving Tree Transformation System*, Ph.D. thesis, Faculty of Computer Science, UNB, 1995.

[52] S. Nigam, *Metric Model Based Generalization and Generalization Capabilities of Connectionist Models*, Master's thesis, Faculty of Computer Science, UNB, 1993.

[53] B. K. Hall and W. M. Olson (eds.), *Keywords and Concepts in Evolutionary Developmental Biology*, Harvard University Press, Cambridge, USA, 2003.

[54] S. B. Carol, *Endless Forms Most Beautiful: The New Science of Evo Devo*, Norton, New York, 2005.

[55] S. B. Carol, J. K. Grenier, S. D. Weatherbee, *From DNA to Diversity*, 2nd ed., Blackwell, Massachusetts, 2004.

[56] L. Wolpert, *Triumph of the Embryo*, Oxford University Press, Oxford, 1992.

[57] R. Schlegel, *Time and the Physical World*, Dover, New York, 1968.

[58] W. H. Cropper, *Great Physicists*, Oxford University Press, Oxford, 2001, p. 277.

[59] J. Jeans, *The New Background of Science*, University of Michigan Press, Ann Arbor, 1959.

[60] H. A. Simon, *The Sciences of the Artificial*, MIT Press, Cambridge, Mass., 1996.

[61] R. P. Feynman, *QED: The Strange Theory of Light and Matter*, Penguin Books, London, UK, 1990.

[62] G. Kane, *The Particle Garden: Our Universe as Understood by Particle Physicists*, Addison-Wesley, Reading, Massachusetts, 1995, Appendix A.

[63] K. W. Ford, *The Quantum World: Quantum Physics for Everyone*, Harvard University Press, Cambridge, MA, 2005, pp. 86–91.

[64] H. Quinn, *Theory: Feynman Diagrams*, 2003.
http://www2.slac.stanford.edu/vvc/theory/feynman.html

[65] N. David Mermin, *Boojums All the Way through*, Cambridge University Press, Cambridge, 1990.

[66] Amir Aczel, *Entanglement*, Plume, New York, 2003.

[67] L. Wolpert et al., *Principles of Development*, 2nd ed., Oxford University Press, Oxford, 2002.

[68] S. F. Gilbert, *Developmental Biology*, 7th ed., Sinauer, Sunderland, Massachusetts, 2003.

[69] ICPR 2004 Satellite Workshop on Pattern Representation and the Future of Pattern Recognition: A Program for Action, chair L. Goldfarb, Cambridge, UK, August 2004. http://www.cs.unb.ca/∼goldfarb/conf/ICPR-2004_Workshop.html

[70] L. Goldfarb and I. Scrimger, On ETS Representation of human movement, Technical Report TR07-184, Faculty of Computer Science, UNB, 2007 http://www.cs.unb.ca/∼goldfarb/ETSbook/Walking.pdf

[71] L. Goldfarb, I. Scrimger, and B. R. Peter-Paul, ETS as a Tool for Decision Modeling and Analysis: Planning, Anticipation, and Monitoring, Technical Report TR07-187?, Faculty of Computer Science, UNB, 2007 http://www.cs.unb.ca/∼goldfarb/conf/DecisionRisk.pdf

[72] J. A. Wheeler, *Geons, Black Holes, and Quantum Foam: A Life in Physics*, 1st ed., W. W. Norton & Company, New York, 1998.