Dynamische Symbolsysteme (Dynamic Symbol Systems)

Dissertation zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften

Eingereicht an der technischen Fakultät der Universität Bielefeld
von Herbert Jaeger, Wissenschaftlicher Mitarbeiter
in der Arbeitsgruppe "Wissensbasierte Systeme (Künstliche Intelligenz)"
von Prof. Dr. Ipke Wachsmuth

Die Arbeit ist in englischer Sprache verfaßt.

1994

Herbert Jaeger
Technische Fakultät, Universität Bielefeld
Postfach 100131, 33501 Bielefeld
email: herbert@techfak.uni-bielefeld.de

Danksagungen

Endlich ist der Zeitpunkt gekommen, wo ich kurz innehalten, zurückblicken und mich bei vielen bedanken kann.

Mein erster Dank gehört Ipke Wachsmuth. Wenn diese Arbeit am Ende zu einem produktiven Beitrag zur Künstlichen Intelligenz geworden ist, so ist das zum großen Teil sein Verdienst. Ich durfte mich auf seinen Beistand immer verlassen.

Alois Knoll, der zweite Hauptgutachter, hat (ebenso wie Ipke Wachsmuth) entscheidende Verbesserungsvorschläge zu einer Vorversion dieser Arbeit gemacht.

Anregungen zur Nomenklatur des mathematischen Teiles verdanke ich Andreas Dress und Josef Meyer-Fujara.

Die Mitglieder der Arbeitsgruppe schufen und erhielten über die Jahre hinweg eine durch Freundschaft geprägte Arbeitsumgebung, der ich viel verdanke. Josef Meyer-Fujara und Barbara Heller haben mich von Anfang an ermuntert und von Zeit zu Zeit aufgerichtet - nicht nur, was fachliche und dienstliche Belange betrifft.

Gert Rickheit befreite mich in einer ungewöhnlichen Blitzaktion für die allerletzte Phase dieser Arbeit von äußeren Sorgen.

Im fernen Bloomington, Indiana, setzte sich Helga Keller mit Eifer für mich ein. Sie verhalf mir (mit der Unterstützung von Douglas Hofstadter) zu einen zweiwöchigen Besuch an der Indiana University, wo ich in einer kritischen Phase viel Ermutigung für meine Arbeit fand.

Mein Bruder Manfred opferte ein Stück Weihnachtsurlaub und wies mir den Weg aus einer mathematischen Sackgasse.

Martin Achterholt bändigte meinen kleinen DOS-Rechner. Ohne ihn hätte die Dissertation nicht mehr auf die Festplatte gepaßt.

Teri Kinealy las die Arbeit in zwei Versionen durch und sorgte durch hunderte von präzisen (und lehrreichen) Anmerkungen für ein glattes Englisch.

Ihnen allen möchte ich von ganzem Herzen danken.

Ein elementarer Dank gehört meinen Eltern. Ihnen widme ich diese Arbeit.

Dynamic Symbol Systems

Abstract

This thesis introduces *dynamic symbol systems* (DSS). The approach combines a symbolic format of information with a self-organizing dynamics. It is primarily intended for the modeling of intelligent, situated agents.

The basic information processing module is a *self-organizing stream*. In computational terms, this is an anytime-algorithm for the processing of information that comes in a quite general stream format. In terms of thermodynamic systems, it is an open, dissipative, rapidly self-organizing system. In terms of cognitive science, a self-organizing stream is a module that can appear at any place from the peripheric sensomotoric interface to the central conceptual level, performing tasks of pattern completion, noise filtering, and gestalt formation.

Different self-organizing streams can be coupled, yielding complex, self-organizing information processing systems. These *associeties* can span the entire periphery-centre axis of an agent. Top-down and bottom-up influences mutually support each other, with none of them having causal or temporal precedence over the other.

At the most central, conceptual level, the DSS representation format is in many aspects comparable to classical semantic networks. The approach proposes answers to several controversial issues concerning the nature of concepts, e.g., context sensitivity, nonmonotonic inheritance, the nature of concepts vs. attributes, and conceptual cycles.

DSS is a concrete formal instantiation of a general, structuralistic epistemology for situated information processing, namely, *dynamic symbol structures.* The key assumption of this perspective is to consider symbols as empirical, physical observables, which can be reliably detected in an information processing system. This epistemological frame yields a unified view on two paradigms that are often considered incompatible, i.e., the physical symbol systems paradigm and situated action.

The dynamics is described in algorithmic terms. It should be relatively straightforward to make the formalism run on a computer. As yet, however, the approach is not implemented.

The thesis first establishes and explores the epistemological frame. It then develops, in a rigorous fashion, the concrete DSS formalism. An application is proposed, where DSS serves as an auxiliary mechanism for memory access in an otherwise classical system. The DSS representation format for symbolic knowledge is compared with classical methods for concept representation. General insights concerning the combination of a self-organizing dynamics with symbolic representation formats are derived from a comparison of DSS with related connectionist models.

Note added for the ftp-able version: latest news on DSS

The thesis concludes with looking out to some open questions (p. 151f). Two months after the thesis has been cast into its final form, I can report considerable progress concerning two of those open issues.

First, a learning technique for the induction of phase generators from observed associations has been developed. Furthermore, the phase generators thus constructed are augmented by probabilities for their transitions. These probabilities lead, in turn, to a refinement of the definition of information (p. 61).

Second, the simulation theorem (p. 64) has been generalized to cover the case of a unified symmetrification/abstraction operation. This should pave the way for a construction of dynamic symbol spaces consisting purely of phase generators.

(august '94)

Table of contents

1 Introduction

This thesis introduces *dynamic symbol systems* (DSS). DSS is a principled formal approach to symbolic information processing. It mediates between classical symbolic AI and emergent computation techniques in AI, the most prominent of which are neural network and classifier systems. On the one hand, the basic informational units in DSS are symbols, as in classical AI. On the other hand, the dynamics in DCS builds on self-organization, as in emergent computation (fig. 1.1).

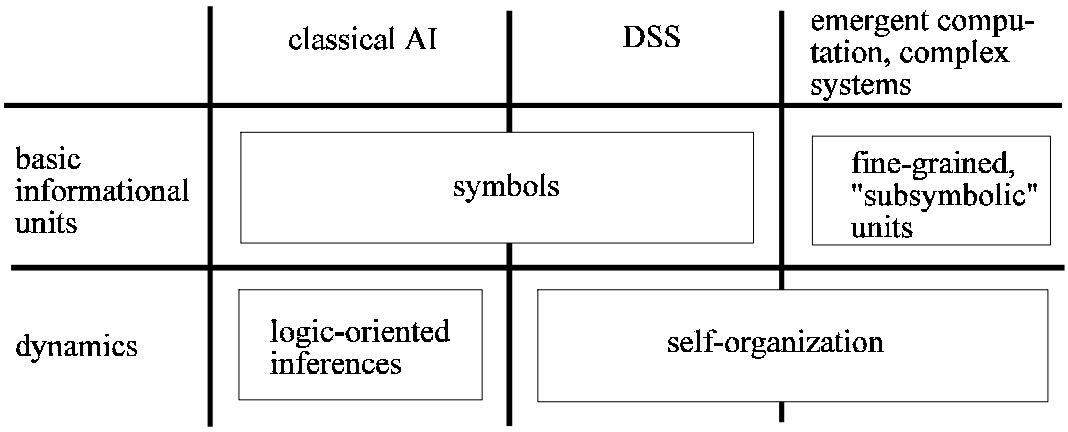


Fig. 1.1: The basic methodological coordinates of DSS.

In order for a symbolic approach to support a self-organizing dynamics, symbols have to be conceived in a somewhat different fashion than in classical AI. In particular, the meaning (reference) of symbols is reconstructed as an empirical observable in the DSS approach. This contrasts with the classical view on symbols, where reference is a "platonically" pre-established relation. In order to make this difference (and others) explicit, symbols are called *dynamic symbols* in DSS.

New formalisms should not be invented without necessity. I believe, however, that combining symbolic representation with a self-organizing dynamics is necessary for the modeling of agents that are both intelligent and situated, i.e., agents which have intellective skills like abstract reasoning and verbal communication, and which are physically embodied in a dynamic environment. More specifically, I argue that the information processing inside such an agent must combine two aspects:

- Informational entities (e.g., perceptual features, basic actuator commands, reflexes, or higher cognitive categories) must be coupled together in composites, and decoupled again, in a fast and causally effective way. This is needed to enable a fast setup of behaviors in unpredictable environmental circumstances.
- This fast composition of informational entities must not depend on explicit control structures. Rather, it must arise in a self-organizing fashion, in which all the agent's processing levels interact simultaneously bottom-up and top-down. This is necessary for the agent to become coupled into its environment in an essentially continuous agent/environment feedback loop.

The single most important contribution of DSS consists in a principled integration of these two aspects. Fast and effective compositionality reflects the formalism's classical symbolic heritage; a self-organizing dynamics links DSS to emergent computation and the situated action paradigm.

The dynamics of DSS is specified in algorithmic terms. However, DSS is not yet implemented.

Overview

DSS aims at reconciling symbolic AI with the situated action paradigm. Section 2 deals with some methodological facets of this endeavor.

In 2.1, I sketch how both traditions approach the task of agent design, highlighting why an integration is difficult. In order to establish an epistemological framework for such an integration, I introduce an abstract structuralistic view on agents, namely, I interpret them as *dynamic symbol structures.*

This perspective is outlined in 2.2. It is worked out in more detail in the following subsections.

2.3 is devoted to the problem of an agent's "vertical axis", which ranges from a sensomotoric periphery to an intellective centre. It can be investigated in morphological and functional terms. At least in biological agents, the periphery-centre axis is structured in a complex way. Morphological and functional aspects are intertwined in an opaque fashion. I argue that such complexity and opacity contribute to an agent's autonomy and adaptiveness. Many complex types of periphery-centre "morphologies" can be interpreted in terms of dynamic symbol structures.

In section 2.4, I explain how dynamic symbol structures can be interpreted as physical symbol systems in the sense of classical AI. The main difficulty is that classical symbols are intrinsically referential, whereas for dynamic symbols, reference is a secondary phenomenon, which must be reconstructed empirically.

Section 2.5 treats the relationship between dynamic symbol structures and the situated action paradigm. Again, a central question concerns the nature of symbolic units. The situated action critique against symbols has an anti-referential and an anti-discrete aspect. The first criticism is answered for by the empirical, contingent nature of reference of dynamic symbols. As to the anti-discreteness argument, dynamic symbols are not discrete in the sense of "yes-or-no entities". Their discreteness resides in the weaker notion of identifiability through an empirical observation procedure. A standard type of dynamic symbols are attractor states that can be observed in neural assemblies.

Section 2.6 further explores the nature of dynamic symbols as physical observables. They are characterized in terms of the information gained through their observation. The less information gained, the more *abstract* the dynamic symbol.

Section 2.7 considers the topic of emergence and grounding. In a restricted sense, these notions are interpreted in dynamic symbol structures by the relationships between fine-grained symbolic composites and coarse-grained dynamic symbols.

Section 2.8 concludes the elaboration of dynamic symbol structures by interpreting them as an instance of *structures* in the specific sense advanced by Piaget. Such structures are characterized by what Piaget calls totality, transformations, and self-regulation.

Section 2.9 addresses a fundamental problem of agent design, namely, the reconciliation of "free" compositionality with a self-organizing dynamics. I argue that both principles are needed for intelligent situated agents. An agent must be able to establish complex behaviors and perceptual schemata from simpler constituents in an essentially arbitrary fashion. This kind of ad-hoc compositionality is a hallmark of symbolic AI, but it is hard to achieve with present emergent computation techniques. On the other hand, an agent must be directly coupled into its environment by essentially continuous perception-action feedback loops. Such mechanisms are a basic ingredient of behavior-oriented robots, but they are impossible to realize with standard logic-oriented techniques. In the light of dynamic symbol structures, a picture of an agent emerges, where compositionality and self-organization co-occur on all levels from the sensomotoric periphery to the intellective centre.

Finally, in 2.10 I try to resolve the apparent contradiction between explicit design on the one hand, and an agent's autonomy on the other. The challenge is *open design*, i.e., equipping the agent with an initial outfit of explicitly designed faculties, such that it can autonomously develop further, unpremeditated faculties during its lifetime. I argue that mimicking biological "modify-and-test" evolutionary mechanisms is too slow a strategy toward this end. I propose to use a "discover-and-modify" technique instead, which exploits the inherent ambiguities of existing functionalities, plus the compositionality mechanisms of dynamic symbol structures. Given ambiguity, new functionalities can be discovered without having to modify an agent's structure first; given compositionality, a functionality that turns out a success can become fixed as a new unit.

Dynamic symbol *structures* provide a epistemological framework for viewing agents. Dynamic symbol *systems* (DSS) are a concrete mathematical and algorithmic instantiation of this general framework. The DSS model is developed in four stages in section 3, which constitutes the core of this thesis. Figure 1.2 shows the essential constructs of each stage.

The elementary underpinnings of DSS are treated in section 3.1. When a dynamic symbol system is monitored locally during some interval of time, a finite sequence of dynamic symbols is observed. Such sequences are called *associations*. The set of associations that can potentially be observed at a given locus in an agent forms a formal language over an alphabet of dynamic symbols. A particular subclass of regular languages, *coherent languages*, is determined by some plausible assumptions concerning observations of dynamic systems. The theory of coherent languages is the mathematical backbone of DSS. These languages can be conveniently described using finite, cyclic *generators*, i.e., edge-labeled transition graphs (fig. 1.2a). Generators can be interpreted as stochastic oscillators. When an association is observed, some *information* concerning the internal state of such an oscillator is gained. Maximally informative associations correspond to *phases* of the device. Normal form generators can be constructed from phases. The main results of 3.1 concern such *phase generators.*

Classical AI systems have a knowledge base, neural networks store long-term information in link weights. In DSS, the "long term memory" is provided by a *dynamic symbol space* (section 3.2). Technically, this is a set of generators, which, by way of an *abstraction* and a *symmetry breaking* operation, can be arranged in a two-dimensional array (fig. 1.2b). Seen from a classical angle, a dynamic symbol space is comparable with a semantic inheritance network. In the perspective of physical dynamic systems, it plays the role of a thermodynamic global state space. Therefore, the generators contained in a dynamic symbol space are called *global states*.

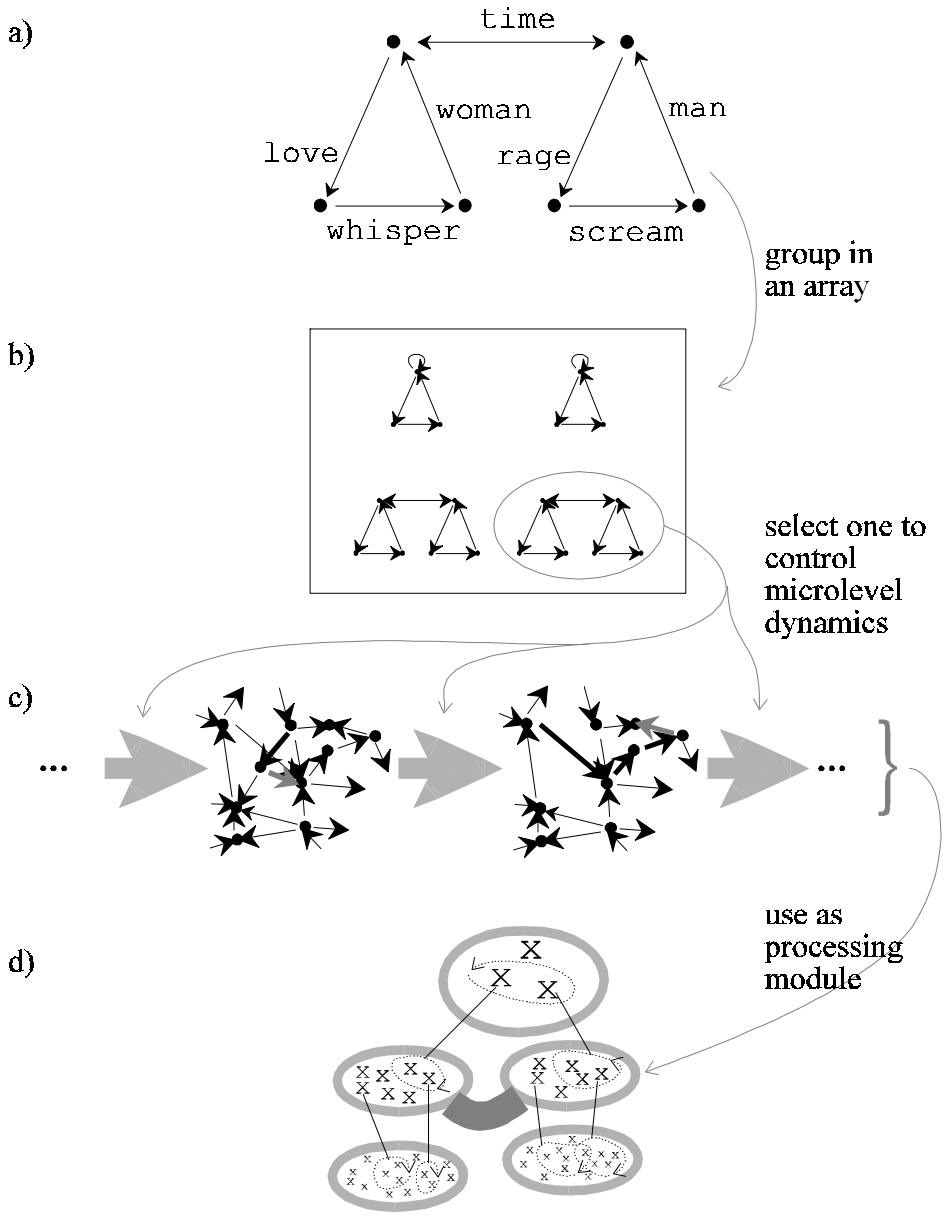


Fig. 1.2: The four stages of DSS. a) a generator, b) a dynamic symbol space, c) a self-organizing stream, d) an associety.

The DSS model of a single processing module is the *self-organizing stream* (section 3.3). I first treat processing modules without input and output (*self-organizing scenes*), which are then equipped with I/O mechanisms to arrive at self-organizing streams. Technically, self-organizing scenes and streams are described as a temporal succession of generators (called *configurations*). The succession is induced by an operation called *microchange* (big arrows in fig. 1.2c). Microchanges locally alter the graph structure of configurations and symbolic labels. Like in physical systems, where the dynamics at the microlevel is affected by global thermodynamic parameters, microchanges depend in their effects on the prior selection of a particular global state from an underlying dynamic symbol space. Since configurations are generators, each of them yields a formal language. Technically, the self-organizing aspect of microchange dynamics resides in the fact that "disordered" fragments of these languages are attracted by "orderly" fragments of the language described by the selected global state. Such

"orderly" fragments are characterized by internal feedback loops. They are called *resonances*. Resonances can be interpreted as a gestalt phenomenon.

Self-organizing streams can be coupled to form complex architectures, *associeties* (section 3.4). There are two coupling mechanisms. First, different streams can communicate via the output and input that they generate and accept (U-shaped band in fig. 1.2d). Second, a higher-level, coarse-grained stream can *ground* in a lower-level, fine-grained stream. Technically, lower-level resonances (dotted circles in fig. 1.2d) are interpreted by single higher-level symbols (straight lines in fig. 1.2d). For instance, a conceptual-level stream can ground in a stream that processes visual features. In that case, the finer-grained resonances made from visual features yield analogical representations for the concept-level symbols. Between adjacent higher and lower level streams, simultaneous top-down and bottom-up interactions exist, such that self-organization in each of the streams fosters self-organization in the other. Associeties are the DSS account of complex agent architectures.

An application of DSS is outlined in section 4. DSS techniques can serve an auxiliary function for accessing an otherwise classic knowledge base, which is used for a text understanding task. The basic idea is to initiate the text interpretation by feeding keywords from the text into a self-organizing stream. Resonances forming in the stream indicate which partitions of the the knowledge base should be accessed in order to load relevant knowledge for completing the interpretation task. A central feature of this keyword-oriented technique for knowledge access is that information contained in the temporal order of keywords is exploited.

In section 5, dynamic symbol spaces are explored as a representation for associationistic, conceptual-level knowledge. Several hard problems of concept representation appear in a new light. They include the nature of concepts vs. attributes, terminological cycles, nonmonotonic inheritance, and varying arity of relations.

Section 6 examines two related approaches in some detail, namely, Sejnowsky's "harmony theory" and Hofstadter and Mitchell's "Copycat". DSS shares a number of characteristics with these (and related) techniques. They include local interactions of informational entities; formation of "coherent" composites; coincidence of concepts, attributes, and relations; context sensitivity of informational entities; feedback and cyclicity; and thermodynamic states as global control parameters. The contribution of DSS to this family of computational approaches lies in its combination of stream processing with a multi-granular architecture.

Section 7 concludes the thesis by looking back, looking ahead, and looking around.

A synopsis of DSS notions

DSS combines ideas from symbolic AI with ideas from self-organizing systems. As a consequence, the notions defined in DSS will be to some extent unfamiliar to readers with either a classical AI or an emergent computation background. Table 1.3 provides a synopsis of important DSS notions, indicating how they roughly relate to concepts from classical symbol processing and emergent computation/complex systems. I hope that this table helps to "situate" my approach in both other fields.

classical AI	DSS	emergent computation, complex systems
symbol	dynamic symbol	A. local observable B. local attractor in a parallel dynamic system
assertion, formula	association	local short-time observation in a parallel dynamic system
symbolic reference	grounding of dynamic symbol in dynamic composite	explanation of observable in terms of finer granularity
textual context	context of an association	local spatiotemporal neighborhood of a local observable in a parallel system
terminologic knowledge base, semantic network	dynamic symbol space	global (thermodynamic) state space
mode of reasoning, e.g. excited vs. calm	global state	global (thermodynamic) state
symbol structure with gestalt quality	resonance	feedback cycle, attractor
anytime algorithm for filtering and completing infinite symbol sequence	self-organizing stream	open (sub-)system
multi-level symbolic information processing system	associety	complex system described on several levels of granularity

Table 1.3: A synopsis of DSS constructs, as they approximately relate to symbolic AI and emergent computation/complex systems.

2 Dynamic symbol structures: an integrative perspective on classical AI and situated action

The task of understanding and designing intelligent, physically situated agents is a prominent goal in current AI research. It motivates the DSS approach. This section examines the area more closely, in order to establish a firm epistemological frame for DSS.

The section is organized as follows. First, a controversy between classical AI and situated action approaches is highlighted (2.1). A structuralistic account of an intelligent, situated agent, as a *dynamic symbol structure,* is then sketched (2.2). It aims at an integrated perspective on both classical and situated-action-oriented agent models. The DSS formalism is a particular formal instantiation of this abstract account. In the main body of the section (2.3 - 2.10), I use dynamic symbol structures for a closer examination of the classical AI vs. situated action controversy, focussing on the task of designing artificial agents.

2.1 A controversy concerning the modeling of agents

In current AI, two complementary approaches to model agents capable of performing complex tasks in a physical environment can be discerned.

The first approach grows out of classical symbolic AI and robotics. A mobile robot platform is equipped with a knowledge-based control system, which includes classical AI components like a symbolic knowledge base, a planning module, and symbolic communication facilities. An example is the robot Flakey (Congdon et al. 1994). Flakey can, for instance, navigate through the corridors of the SRI institute, executing tasks like delivering an object to a person whose location must be asked from other persons.

The second approach has its roots in cybernetics, artificial life research, and in the epistemological perspective now called *situated action*. The agent's performance is achieved through a multitude of relatively simple *behaviors*. Each of them continuously receives sensor input and generates action responses; it acts essentially like a reflex. No explicit knowledge representation and inferencing exists. More complex behaviors emerge in a self-organizing fashion from the interaction between the basic built-in behaviors, and from the interaction of the robot with its environment. Paradigmatic examples are the robots built by Brooks (1989). These "insect-like" automata move in an unknown, changing, obstacle-cluttered environment, and exhibit behaviors like obstacle avoidance, wall following, or keeping distance from each other.

The two approaches correspond to different perspectives on intelligence. The first emphasizes symbolic reasoning, explicit knowledge, and verbal communication. It is the traditional, top-down perspective of symbolic AI. A brief account of this perspective is provided by Wachsmuth (1994). The second view sets out to reconstruct intelligence in a bottom-up fashion, stressing aspects of real-time adaptive behavior in a dynamic physical environment: *"Behavior is intelligent if it maximises preservation of the system in its environment"* (Steels 1994). A brief introduction is given by Brooks (1991).

Considering this complementarity, it seems natural to work towards an integration. To be sure, state-of-the-art mobile robots usually feature both reflex-like low-level mechanisms, and high-level symbolic reasoning capabilities. Four out of the five winners of the 1993 AAAI robot competition are of this "hybrid" kind (Nourbakhsh et al. 1993). But, these architectures can hardly be considered a true integration of the two perspectives, since they fall short of fully realizing the potentials either of classical AI or of behavior-based approaches.

First, mobile robots do not, as yet, realize the possibilities of current knowledge-based AI techniques in a satisfactory way. Real-time constraints severely limit reasoning capabilities. Contest-winning mobile robots do only minimal high-level reasoning, using a small, task-tailored knowledge base. For instance, Flakey's knowledge in the 1992 AAAI robot contest was centered around three kinds of objects (standardized boxes, standardized poles, and walls), and around a single task and the subtasks connected with it (namely, finding poles in a wall-bounded arena) (Congdon et al. 1994).

Second, current contest-winning mobile robots do not realize the potentials of behavior-oriented techniques either. A central issue in behavior-oriented AI is to make agents autonomously adaptive by letting them evolve new behaviors, which are not premeditated by the designer (cf. 2.10). Such mechanisms are not incorporated in the mobile robots in question here.

No theoretical framework is available for the design of agents that truly integrates the classical with the behavior-oriented perspective. Research in the field proceeds by rote experimentation (Hanks, Pollack & Cohen 1993). Nourbakhsh et al. (1993, p.61) sum up the lessons learnt from the 1993 AAAI robot contest by stating that *"one conspicuously missing aspect of the current work in robotics is a strong tie between theory and practice."*

There are at least two serious obstacles opposing attempts towards a theoretically well-founded, integrative research strategy.

For one, the two approaches rest on different formal foundations. Classical AI mostly uses discrete mathematical tools from logics. A wide range of well-understood formal techniques is available. In contrast, there are yet no comparable formal standards for behavior-oriented AI. Appropriate formal theories can be expected to be influenced by cybernetics, control theory, and formal theories of complex physical systems (Steels 1993a). A first sketch of such a formalization is given by Steels (1993b). He outlines a model in terms of complex systems theory, where the temporal development of agents in a physical environment is described by differential equations.

These differences in formal background, concerning both mathematical nature and degree of elaboration, make an integration of classical top-down and behavior-oriented bottom-up approaches technically difficult.

In addition, there is a "cultural" gap between the two views. Behavior-based approaches are an important instance of an epistemological perspective, the *situated action* paradigm, which is understood by some of its proponents expressly as a challenge of classical AI. In a special issue of the Cognitive Science magazine, opinions clash uncompromisingly. The proponents of classical AI (Vera & Simon 1993a,b,c) deny that situated action is an original paradigm at all. They claim that the classical physical symbol system hypothesis (Newell 1980a) is a sufficient methodological background for modeling situated agents, and that situated action is subsumed by classical AI. Defenders of situated action argue that an agent and its environment have to be understood as a single dynamic unity, where processes inside the agent cannot be theoretically separated from the agent's continuous interaction with the environment. The roots of

intelligence, in this view, lie in non-symbolic agent/environment feedback cycles. The basic research methodology has to be adapted from neurophysiology and the system sciences (Clancey 1993). Internal, symbolic representations are treated as second-order phenomena (Greeno & Moore 1993). This reverses the classical approach, where symbolic representation comes first.

The questions that surface in this debate have a direct impact on the methods used for modeling situated agents. I shall analyze them in the remainder of this section, thereby developing the epistemological framework of *dynamic symbol structures,* in which the DSS formalism is grounded. This framework is intended as a step towards a theoretically founded integration of classical AI with situated action.

For the sake of brevity, I shall use the term "agent" instead of "intelligent situated agent" throughout.

Summary of section 2.1

- The classical approach to agent design is top-down oriented, focusses on intellective, knowledge-dependent faculties, and relies on explicit, symbolic, logic-oriented techniques.
- Behavior-oriented approaches start in a bottom-up fashion from reflex-like behaviors, by which the agent is directly coupled into the environment, forming an integrated agent/environment system. Techniques are often non-symbolic.
- An integration of both approaches appears difficult since the formal backgrounds differ and, even more importantly, there are two quite different notions of intelligence involved.
- The work presented in this thesis claims to contribute to an integration.

2.2 Dynamic symbol structures

This subsection outlines a structuralistic model of an agent's architecture. An agent is abstractly described as a *dynamic symbol structure.* The proposal aims at a unifying perspective for classical AI and situated action approaches. The outline given in this subsection will be filled out in detail in subsequent subsections.

The viewpoint of dynamic symbol structures rests on three main points:

- An agent's information processing system is globally organized on a *multi-level periphery-centre axis.*
- Within each level, information processing is achieved by interactions of *dynamic symbols.* Dynamic symbols are in many respects similar to classical symbols. A fundamental difference is that dynamic symbols are rigorously construed as physical observables, not as "platonic" entities.
- Dynamic symbols in a given processing level form *dynamic composites.* Dynamic symbols in the next higher level *emerge* from these composites. These emergence relations are constitutive for the periphery-centre level topology.

In the remainder of the subsection, I elaborate a bit on the second and the third point, the first being a standard assumption in virtually all agent models (cf. 2.3).

Dynamic symbols are defined as entities which play a dynamic role in an agent's information processing, and which can be empirically observed when they occur. They resemble classical symbols in being reliably identifiable entities. Unlike the former, dynamic symbols are not interpreted as being referential in their nature. Being physical observables in a dynamic system, they exhibit various dynamic phenomena. For instance, they occur intermittently, they may vary on some intensity scale, or they may become superimposed on each other.

The notion of dynamic symbols is positively intended to include, in biological agents: neural spike discharges, spike trains, activation patterns in neural assemblies, EEG potentials; in artificial agents: sensor read-values, features extracted from sensor signals, the electronical embodiments of symbols or propositions manipulated in a classical inference engine; in both biological and artificial agents: reflex-like behaviors, utterances, gestures, and social interaction patterns. The observer who identifies these dynamic symbols can be external in all of these cases. Sometimes, the observer can also be the agent itself, e.g., when neural activation patterns correspond to identifiable experiences, or when an agent observes its own behavior. The notion of dynamic symbols will be successively refined throughout the remainder of section 2.

To be sure, the notion of dynamic symbols is not as well-defined as one might wish. The notions of "empirically observable", "dynamic role" and "an agent's information processing", which occur in the tentative definition of the term, are ill-specified. But, neither is the notion of classical symbols well-defined (criticism on this behalf in Clancey 1993). The fundamental notions of an epistemological perspective cannot be fixed by definitions in the strict sense, exactly because they are fundamental. Their interpretation becomes fixed as their usage stabilizes in the community of their users (extensive treatment of this and related methodological topics in Kamlah & Lorenzen 1972).

Within each level, dynamic symbols can form empirically observable *dynamic composites.* The composition mechanism can be (spatially) structural (as in classical symbolic composition), temporal (as in the composition of single spikes in a spike train), or spatiotemporal (as in the composition of spike trains in attractor states).

From a dynamic composite in a given level, a dynamic symbol in the next higher level can *emerge*. Conversely, dynamic symbols in a given level can *ground* in dynamic composites in the next lower level. For instance, from a dynamic composite of local activity patterns in the primary visual cortex, a particular activity pattern in the secondary visual cortex may emerge (which subsequently may lead to a report of a recognized "table", or which can be observed by a neuroscientist directly). Another example: the activation of a "table" node in a semantic network with a spreading activation dynamics grounds in the activation of a collective of feature nodes. The emergence/grounding relation between a dynamic symbol and a dynamic composite is not merely in the eyes of the beholder. It must be empirically justified, either by an observable, significant correlation, or (preferably) by tracing down a causal connection.

The emergence/grounding relation between dynamic composites and dynamic symbols is considered the primary phenomenon, the level structuring as a derived one. A clean linear ordering of levels on the periphery-centre axis can be considered an ideal case. More intricate global topologies of the level structure are expected in sufficiently complex agents. Fig. 2.1 sketches some examples.

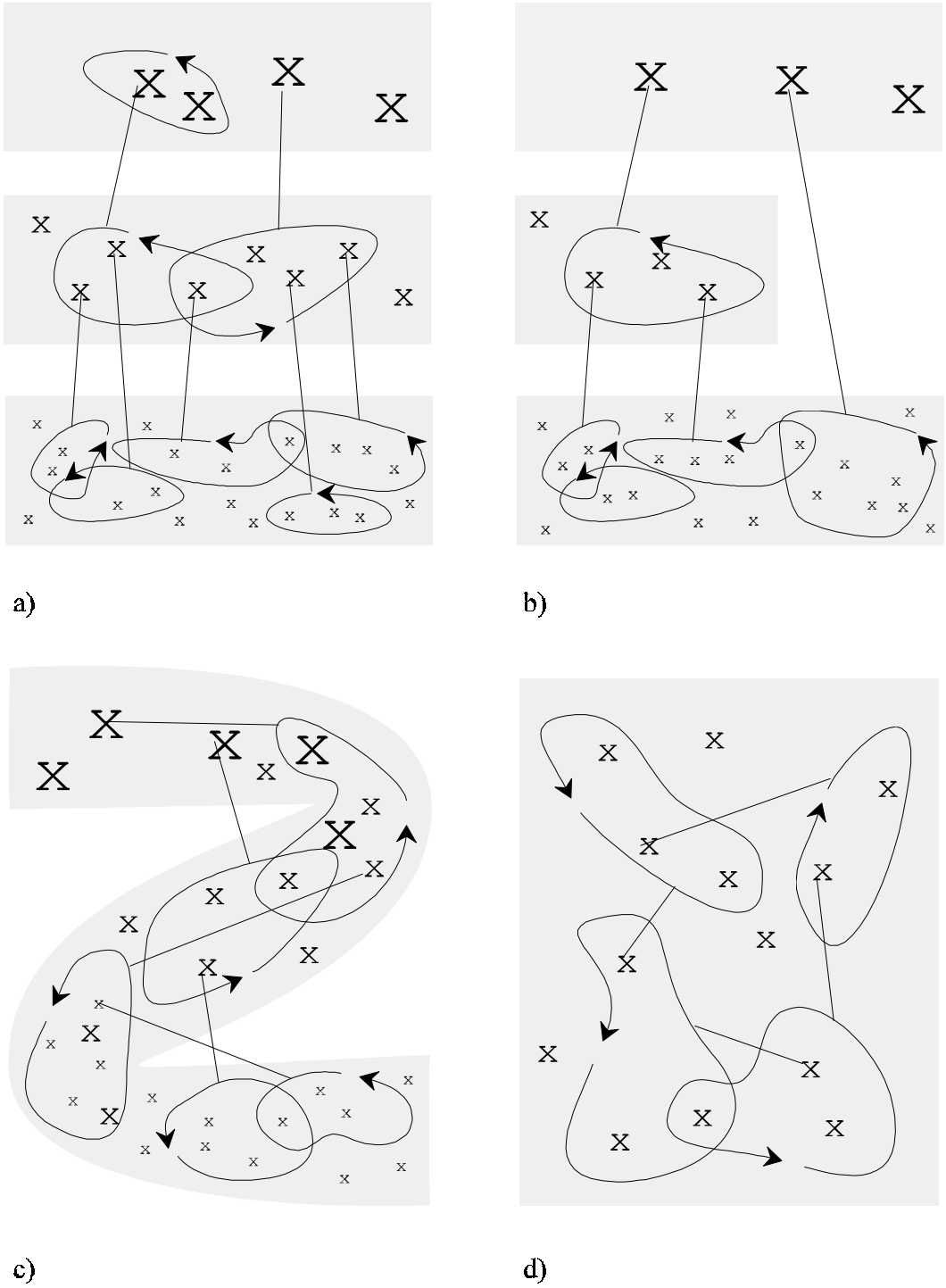
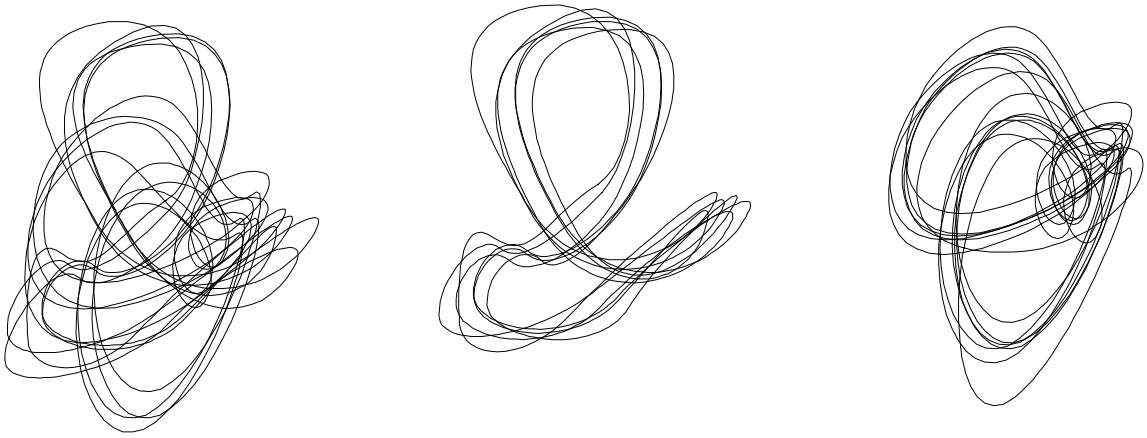


Fig. 2.1: An ideal linear ordering of levels in the periphery-centre axis (a), and some variations (b, c, d). Dynamic symbols are rendered as x's, dynamic composites as cyclic arrows enclosing the symbols participating in the composite, emergence/grounding relations as straight lines, and levels as shaded bands. In b), a level is cut short, in c) a level "spirals back" over itself, in d) the level structure breaks down due to emergence relations being recursive.

When higher levels successively ground in "finer" ones, the question of how the successive refinement ends arises. There are three principal alternatives: first, there can be an (arbitrary) "atomic" level; second, there may be an infinite regress; third, there may be cyclically recursive

references between levels. This problem has been called the "Münchhausen trilemma" in the epistemology of science. Note that there arises a mirrored version of the same trilemma by going upwards in the hierarchy. The dynamic symbol structure proposal for viewing agents is not committed to one of the trilemma's alternatives.

A dynamic symbol can come in different degrees of *differentiation*. For instance, a chaotic attractor state in a biological or artificial neural network may occur in varying degrees of noisiness (or "computational temperature"), where a high-noise, nearly random occurence yields a de-differentiated version, and a low-noise, nearly limit-cycle occurrence yields a differentiated version (fig. 2.2). Another example is an object scheme that is introduced into working memory (e.g., a KL-ONE ABox) to account for an external object that is about to be classified. At the time of generation, this scheme is typically poor in information. During the recognition process, it is enriched with detail, and initial inconsistencies are filtered out. This amounts to a differentiation.



a) b) c)

Fig. 2.2: A de-differentiated (a) and two alternative differentiated versions (b, c) of an attractor state (approximately redrawn from Yao & Freeman 1990).

The aptness of dynamic symbols to occur in varying degrees of differentiation is an obvious difference to classical symbols. It arises naturally from the perspective on dynamic symbols as physical observables. By contrast, classical symbols are "platonic" entities (cf. 2.5), for which a notion of empirical differentiation makes no sense.

The identification of a dynamic symbol is sensitive to the precision of the observation procedure. This leads to *high-resolution* vs. *low-resolution* versions of a dynamic symbol. For instance, the extracranial recording of electric potentials yields a low-resolution version of a cortical activation pattern, radioactive marker methods can yield medium resolution, and electrosensitive dyes monitored at the open brain achieve a high resolution. In a computer, monitoring the workload percentage of a processor provides a very low-resolution observation of its working state, whereas a coredump gives a high-resolution account.

The information available about a dynamic symbol occurence results from the combined effects of differentiation and resolution, i.e., from "what is there" and from "how sharply it is seen". A high-resolution observation technique yields little information when the dynamic symbol is poorly differentiated, and a noisy observation procedure tells little even about highly differentiated dynamic symbols. The net information afforded by an observation varies on a scale that is called, in the dynamic symbol structure framework, the *abstraction* scale (the inverse being *specialization*). Specialization increases with differentiation and/or resolution.

A dynamic symbol is defined as an entity (playing a dynamic role in an agent's information processing) *as it is observed.* I.e., it is defined in terms of information obtained, that is, in terms of abstraction. Different abstraction implies different dynamic symbols. For instance, an intracellular recording of neural activity is a delicate affair, and it is often difficult to filter any pattern from the noise at all. The noise in the the raw data can have two sources: the pattern itself can be weakly expressed (poor differentiation), or the recording procedure can be subject to disturbances (poor resolution), or any combination of both. These two sources of uncertainty are not distinguished in the definition of dynamic symbols.

Thus, a dynamic symbol is *not* considered an entity "objectively there" in the agent. But, one is very much accustomed to talking about observables in this way. In fact, one does not have to give up this way of talking entirely. It is justified when the resolution is "better" than the differentiation. Whether this is the case can be decided by increasing the resolution. When such an increase does not yield an increase in information (i.e., no specialization is obtained), then one can ascribe the information obtained fully to the observed entity. In other words, in such cases one does, in fact, have a handle on the entity as it is "out there". For instance, when the noise in an intracellular recording cannot be reduced by the use of more sophisticated electrodes and signal detecting machinery, one is justified in talking about "the" activity pattern (which, in this example, is inherently noisy).

A dynamic symbol appears in a researcher's theory as a *formal symbol*, which can be anything from a mathematical symbol (when the theory is highly formalized) to a telling name (when the theory is in a pre-formal stage).

This ends the first outline of the essentials of dynamic symbol structures. The formalism of *dynamic symbol systems* (DSS), to be presented in section 3, is a particular mathematical instantiation of this framework.

Summary of section 2.2

- Dynamic symbol structures provide an abstract, structuralistic framework, aiming at an integrative view on classical and situated action approaches to agent modeling.
- An agent is viewed as being organized on a periphery-centre axis.
- Each level in this axis is understood in terms of dynamic symbols and dynamic composites. They are conceived as physical observables, not as platonic entities.
- As a consequence of being physical observables, dynamic symbols occur in different degrees of abstraction, with abstraction having the two components of objective differentiation and of observation-dependent resolution.
- The way of talking about dynamic symbols as if they were "objectively there" is justified when resolution can be made finer than differentiation.

- From dynamic composites in one level, dynamic symbols in the next higher level emerge. This emergence relation induces the level topology.
- There are many other possible level topologies besides a simple linear ordering.

2.3 The periphery-centre axis

The idea of an agent's information processing being organized on a periphery-centre axis does not seem in principle problematic. It is certainly accepted in classical AI. In behavior-oriented research, it is de-emphasized in favor of behaviors, which at the present state of the art typically belong to peripheric levels. The existence of more central levels is acknowledged, as witnessed by the robots designed by Steels and Brooks. However, the periphery-centre dimension appears in these two approaches in two quite different fashions.

In Steels' architecture, there exist on top of the reflective behavior level two levels of a different nature, namely, a *(subcognitive) process layer* and a *(cognitive) symbolic layer* (Steels 1993a). The process layer concerns a variety of global parameters for integrative and representational aspects of low-level sensor and action data. The symbolic level is a rule-based symbolic reasoning system. Thus, the periphery-centre axis in Steel's robot is rather similar to classical AI architectures.

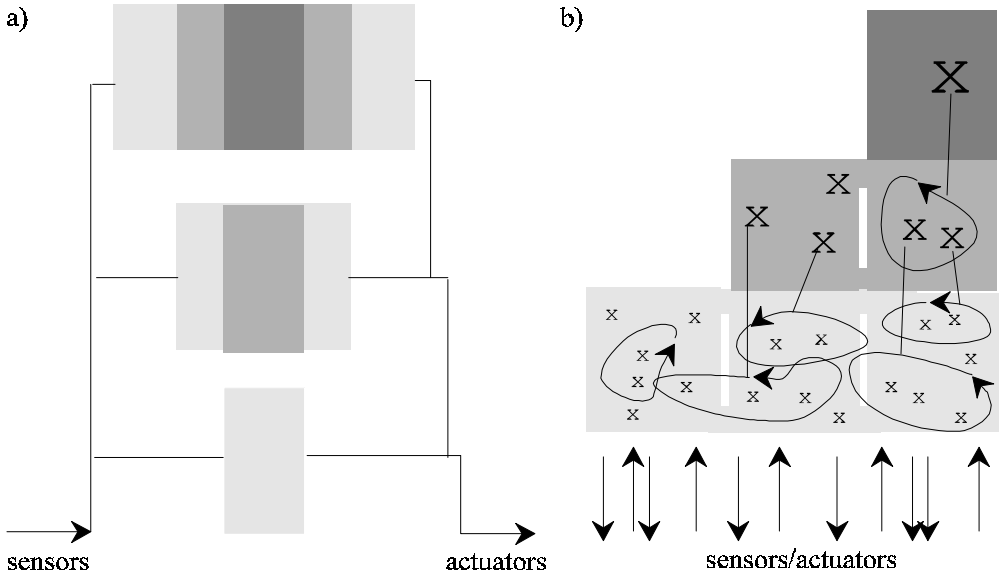


Fig. 2.3: The subsumption architecture and the periphery-centre dimension. Increase in centrality is indicated by darker shading.

In the subsumption architecture proposed by Brooks (1989), higher behaviors are put on top of more basic ones, after the latter have been made to function satisfyingly. Higher behaviors are intended to account for faculties of a more intellective character, like object identification or reasoning about object behavior. Since a behavior, by definition, has direct access to the sensomotoric interface level, higher behaviors in a subsumption architecture do not communicate with the sensor and actuator level through lower behaviors. But, a behavior as complex as object recognition must obviously proceed in several stages from raw sensor data to its final response. Brooks attests that to a certain extent one needs "*to decompose a single layer in the traditional manner*" (p. 435). The picture that emerges is that low-level behaviors, like biological reflexes, have a rather direct connection to sensors and actuators, whereas high-level behaviors presumably consist internally of several processes of increasing distance to the sensomotoric interfaces. Fig. 2.3a sketches Brooks' variant of the periphery-centre organization, and fig. 2.3b indicates how it can be recast as a dynamic symbol structure.

Contrasting Steels' and Brooks' architectures, one finds that although the periphery-centre dimension seems to be plausible in general, more than one way of telling the story exists. How one sees natural agents, and how one designs artificial agents, depends on some more or less explicit *background theory*. As an additional source of variance, these theories can typically each be practised in several "flavors", which I shall call *orientations*. For the present purpose, the most important ones are the *morphological* and the *functional* orientation. They turn up almost universally in disciplines concerned with agent modeling.

Before I take a closer look at morphological and functional aspects of the periphery-centre axis, I dismiss another viewpoint that might be mistakenly connected with this axis. It is exemplified in a classical paper of Newell (1980b). Newell describes artificial intelligence architectures in terms of levels that range from the *knowledge level* at the top, via some intermediate levels like a *program level* and a *circuit level*, down to a *device level*. I would like to call this a hierarchy of "disciplinary" levels, since each level is expressly considered by Newell in terms of a particular engineering (sub)discipline.

This hierarchy is basically different from a periphery-centre account because an entire agent can, in principle, be described on any of the disciplinary levels. This is made clear by Newell: *"Within each level, systems hierarchies are possible, as in the subroutine hierarchy of the programming level. Normally, theses do not add anything special in terms of the computer system hierarchy itself"* (p. 95). The only exception to this rule is, I understand, the knowledge level. It appears that peripheric sensomotoric processes cannot be described on this level.

Having cleared this point, I start with a brief discussion of problems faced when one tries to account account for the periphery-centre axis in morphological terms.

The place of a processing module on this axis is defined morphologically by its physical access route to sensomotoric information. Peripheric modules have a direct access; towards the centre, information from/to the exterior is relayed through an increasing number of intermediate modules. For instance, the vision system in vertebrates can be ideally described in terms of information coming from the retina, passing through a number of nuclei, and finally arriving in certain regions of the neocortex.

This ideal of a linear periphery-centre ordering is blurred by the ubiquity of feedback cycles. Brain nuclei and cortical regions are densely cross-connected (methodological consequences discussed in Arbib 1987). Even approximately, a linear ordering of modules in terms of distance from the sensomotoric interface level is lost.

Another problem is that morphology is quite often "fuzzy". Brain modules rarely are clearly delimited from each other morphologically. In von Neumann computers, all information processing happens in the same processor, and the most diverse subprograms are mixed up with each other on the same physical memory devices. Thus, the idea of a well-defined morphological separation of modules is a very coarse approximation at best.

In sum, the periphery-centre axis is mirrored in an agent's morphology only to a limited degree. However, morphology should probably not be dismissed altogether. At least for understanding biological agents, and in connectionist approaches that strive to exploit biological models for artificial architectures (e.g., Yao & Freeman 1990, Carpenter & Grossberg 1990), morphology considerations are indispensable for structuring an agent.

The functional orientation describes information processing in terms of goals and the mechanisms (algorithms) to satisfy them. Functional descriptions are often organized in a task hierarchy along a periphery-centre axis.

A prominent example is Marr's theory of vision (comprehensive discussion in Burge 1987). Marr describes in a rigorous fashion how objects are recognized. He proposes a series of representational formats and algorithms, which account for a one-directional information flow from the sensor-near level of extracted features to the central level of object recognition. The representational units occurring in this theory can be interpreted as counterparts of geometric and optical properties of the physical objects from which the sensory information originates. Marr's theory explains how fragmented, multi-channel information becomes stepwise integrated, such that external objects are *reconstructed* from the information transmitted by the signal.

Another example of a functional account organized along a periphery-centre axis is the classical linguistic hierarchy. It describes speech understanding in terms of an ascending process that runs through a variety of morphologic, syntactic, and finally, semantic stages. Again, the process is reconstructive. It is assumed that the sender puts a particular semantic content into an utterance, which is reconstructed by the recipient.

These examples suggest that a functional orientation concerning signal processing pathways (and analogically, actuator control) is a straightforward affair. On closer inspection, this impression blurs, at least in the case of biological agents.

With respect to vision, Arbib (1987) argues (i) that an appropriate modular decomposition of function has to be quite fine-grained, (ii) that there exists a large amount of parallelism and interconnectivity between functional modules, (iii) that functional architectures differ considerably across species, which mirrors the fact that *"there is no unique algorithm for solving a given problem"* (p. 344). The situation is further complicated by the fact that functions do not map one-to-one to physical modules. The mapping is many-to-many in biological systems. The general impression conveyed by such closer inspection is that signal processing in biological systems is realized by a fine-grained, partially parallel, highly redundant network of modules, whose function is not easy to determine and may even be intrinsically ambiguous.

The problems connected with functional descriptions for biological systems can hardly be overestimated. Even paradigmatically simple instances of biological information processing reveal themselves as intricate superpositions of mechanisms whose known complexity keeps growing as observation techniques are being refined. A striking example is the monosynaptic reflex that leads to a retractory muscle activation after mechanic stimuli. It appears, today, as an intricate network of regulatory pathways whose interaction and functionality is far from being completely understood. Only recently, for instance, and quite unexpectedly, this reflex

has been demonstrated to be modulated electrically by nearby fibres in a complex fashion (el Manira et al. 1993).

It can be argued that the functional opacity of morphological modules is advantageous for an agent's adaptiveness. The physics of a biological agent, in this perspective, is a reservoir of potential functionalities. When environmental pressures demand it, imperfectly available functions can become dedicatedly expressed by physical modifications which can be effected by a slight genetic change.

Artificial agents are, as yet, insufficient in many aspects. They break down when environmental disturbances exceed a narrow margin of standardization, and their scope of "understanding" is limited. It might be argued that these insufficiencies are in part caused by the clean functional decomposition that underlies their design. The more precisely the functions of subprocesses are specified, the closer the signal's characteristics must agree with these specifications. The reconstructive nature, which seems to be inherent in functional specifications, requires the signal's characteristics to conform with the abstract model of reality that serves as a basis for the functional specification of the signal processing apparatus. This conformity assumption will often fail, since reality is always phenomenologically richer than models thereof.

The comparison of adaptive, but functionally opaque biological agents with brittle, but functionally clear-cut artificial agents suggests that there might be a tradeoff between clear functional decomposition and the flexibility necessary for adaptive behavior. This impression is substantiated by considerations that originate in the situated action paradigm.

A functional account of a process is goal-oriented: functions inherently serve some specified goal. Typically, such goals are steps in a reconstruction process. For instance, in Marr's theory there is a functional module that serves to compute generalized cylinders from the 2½-D-sketch (which is one of the intermediate representational formats in this theory). Consequentially, a functional specification of an agent requires that a hierarchy (or some other kind of network) of goals has to be explicitly provided by the designer.

This does not conform with the *autonomy principle*, which is emphasized by the situated action paradigm. In an extreme philosophical version, the principle states that the internal functioning of an agent is not reconstructive at all. Rather, the only discernible goal that can be ascribed to a living system from the outside is to maintain itself in a dynamic environment. This is achieved by adaptive processes, which are claimed to be principally individual: *"Which neural activities are triggered by* which *perturbations is determined exclusively by the individual structure of each person, not by the properties of the perturbing agens"* (Maturana & Varela 1984, chapter 1, my translation from the German edition 1992[4]). The system incessantly constructs itself anew. This is spelled out in the notion of *autopoiesis* in the epistemological theory of *radical constructivism* (Maturana 1978). Maturana and Varela are regularly quoted by situated action proponents. Ascribing goals to internal processes from the outside is, in their view, incompatible with a system's autonomy.

In a more practical vein, the autonomy requirement leads to unsupervised, "*autocatalytic*" (Steels 1994) learning schemes for behavior-based robots. Learning can be augmented by evolutive mechanisms. Robots that learn and evolve cannot be functionally designed beforehand in an exhaustive fashion. Steels (1994) discusses the issue in some detail.

Taken together, these considerations recommend to use functional descriptions with care. For biological agents, neither clear nor complete functional accounts are likely to be obtainable. When they are used as blueprints for artificial agent design, they are prone to interfere with

autonomy. However, functional descriptions are a natural and common way of viewing agents. They contribute to discover (or construct) a periphery-centre topology that makes sense. The issue will be reconsidered in subsection 2.10.

In dynamic symbol structures, there is no systematic place for functionalities. Functional explications of dynamic symbols and of composite-forming mechanisms can be introduced whenever they seem appropriate, but they remain outside the structuralistic account proper. The only "task" that becomes visible for the structuralistic perspective proper is that the system "tries" to maintain itself dynamically (cf. 2.8).

Summary of section 2.3

- Two important orientations in theories concerning agent architectures are the morphological and the functional.
- Both contribute partially to a periphery-centre structuring.
- The periphery-centre topology is, at least in biological agents, more complex than a linear ordering of levels. Biological agents are structured in a highly complex, if not opaque, fashion with respect to a combined morphological-functional architecture.
- This appears to be connected with their autonomy and adaptiveness, which suggests that artificial agent design has to be cautious about clear-cut functional blueprints.

2.4 Dynamic symbol structures and classical AI

The background philosophy of classical AI is codified in the physical symbol systems hypothesis (Newell 1980a). This hypothesis states that every intelligent information processing system can be understood as manipulating physically realized symbols. Symbols and composite symbolic structures are defined by two aspects: being physical patterns, and being referential.

First, classical symbols are physical patterns of some (indeed, any) sort: *"... in any event, their physical nature is irrelevant to their role in behavior"* (Vera & Simon 1993a, p.9). Expressly, neural activation patterns are cases of symbol structures: *"The patterns of neural structures and processes* are *the ... symbols, just as the patterns of holes in an old IBM card* are *the symbols that contain the card's information"* (Vera & Simon 1993c, p.121). Being patterns, symbols are subject to identification and comparison mechanisms of a discrete yes/no character:

"When we say that symbols are patterns, we mean that pairs of them can be compared (by one of the system's processes) and pronounced alike or different, and that the system can behave differently, depending on this same/different decision." (Vera & Simon 1993a, p.9)

So far, the physical symbol systems hypothesis is not different from the assumptions underlying dynamic symbol structures. Both perspectives see symbols as physical entities with a well-defined identity. There are, however, differences in the conception of how the identification of a symbol is actually achieved. In the case of dynamic symbols, some observer with a

background theory is required, where only the theory enables the observer to establish an operational observation procedure. In the case of classical symbols, identifiability is implicitly taken as granted; classical symbols are assumed to exist independently of an identification procedure. This makes the observation of classical symbols an unproblematic basic fact, upon which the system itself (as in the quotation above) or external observers can depend: *"To varying degrees,* [symbols] *can be observed, handled, even dissected"* (Vera & Simon 1993c, p.121).

The second characteristic of classical symbols is their referential nature: *"We call patterns symbols when they can designate or denote"* (Vera & Simon 1993a, p.9). Potential referents of a symbol or composite symbol structure are manifold. They include other symbol structures, patterns of sensor stimuli, or motor actions. These are cases of *internal* reference, in that they occur within an agent (a possible exception being other symbol structures, which can occur external to the agent, as in verbal communication). The prototypical referents are, however, *external* objects or situations. The symbol-referent link is implicitly treated as a pre-established, platonically true fact. This is not contradicted, but supported by conceding imperfections: *"Of course, the internal representation of a real scene will be highly incomplete and may be inaccurate"* (Vera & Simon 1993a, p. 10). In this quotation, an ideal of completeness and accuracy is tacitly assumed, which can serve as a measure for deviations. In a nutshell, then, object recognition is the reconstruction of a pre-established, ideal symbol-referent link, by possibly imperfect means. This reflects an ancient philosophical view, which has passed to classical AI via positivistic mathematical logics and classical linguistics: for Frege and Russell, *the* symbol "Venus" has *the* meaning of the star that appears in the morning and in the evening.

This understanding of symbolic reference follows a straight route to a logic-oriented, model-theoretic methodology. In particular, the meaning (i.e., reference) of composite symbolic structures is derived from the meaning of its constituents (Fodor & Pylyshyn 1988). This allows to formalize notions of "correct" interpretations of symbol structures, which in turn allows a mathematically rigorous development of theories of symbolic representation and inferencing. This course is taken in classical "disembodied" AI applications like expert systems or machine language translation systems.

Due to the convincing scientific progress achieved in such classical applications, a logic-oriented methodology is also considered appropriate for modeling agents. Several strategies are presently pursued in this very active area of research, e.g. situation semantics (Devlin 1991) or modal logic approaches (e.g., Cohen & Levesque 1990). However, it is not clear whether they can ultimately lead to success. As yet, there exist no comprehensive logical formalisms that accounts simultaneously for complex internal states of an agent, a changing environment, and a dynamic coupling of the two via sensoric and motoric events. Present research proceeds largely on a theoretical level. Very likely, computational complexity will turn out a serious obstacle when it comes to put complex future logics to work (more about this in 2.9).

But, the physical symbol systems hypothesis is not intrinsically committed to be worked out in terms of model-theoretic logics, and no mention thereof is made in its most recent reformulation (Vera & Simon 1993a, b, c). I would even further this point, claiming that the substance of the physical symbol systems hypothesis is left intact when symbol reference is not considered as a platonic ideal but as reasonably reliable, empirically observable mechanism. Such a shift in background philosophy would still be compatible with the idea of reasoning as a computational manipulation of discrete, physical symbols. To my understanding, the latter is the core of the classical paradigm, whereas the platonic accent is an implicit remain of an ancient epistemological tradition, inherited through (and enhanced by) mathematical logic and classical linguistics.

Interpreted in this fashion, the physical symbol systems paradigm does not a priori conflict with dynamic symbol structures. In order to arrive at a true match between the two perspectives, it remains to be examinend how symbol reference can be accounted for in the latter.

The dynamic symbol perspective is open to a treatment of external reference in the following fashion. A link between, for example, a cortical activation pattern and a cat "out there" can be constructed by providing theory-guided observation procedures for the cat, for the optical transmission of information from the cat to the retina, for the emergence of activation patterns on several relay stations from the retina to the cortex, plus theory-guided explanations for the machinery that leads to the various emergence steps. The recognition of a cat by an artificial agent would have to be explained in a similarly complex fashion. The result of such a (tremendously complex) effort would be an empirically verifiable cat-to-pattern link.

From the point of view of dynamic symbol structures, object recognition is a complex physical process, which is inseparably coupled to the particular physical processing apparatus of the agent, and which requires a large amount of background theory on the side of the observer to become detectable at all. In this respect, the dynamic symbol structure worldview is the same as in radical constructivism. This is concisely expressed by Maturana and Varela (1984, my translation from the German edition 1992[4], p. 31):

"In this sense we will always find that one cannot understand the phenomenon of cognition as if there were «facts» and objects «out there», which one only would have to fetch and put into the head. [...] *The experience of every thing «out there» becomes configured by the human structure in a specific way..."*

Unlike its platonic counterpart, the dynamic symbol reference link is susceptible to physical disturbances of all kinds due to its empirical nature. False positves or false negatives are explained in an epistemologically straightforward fashion. By contrast, the platonic perspective requires involved arguments to treat such failures. The difficulties become manifest in the intricacies of nonmonotonic reasoning, which can be seen (among other aspects) as the logical account of recapturing from false assumptions, and restoring truth.

Inside the agent, the empirical reconstruction of external, object-to-concept reference (e.g., a cat-to-pattern link) traverses the entire periphery-centre dimension. Each transition between levels on this axis has to be accounted for individually. In a dynamic symbol structure, these steps are described as the emergence of dynamic symbols on some level from dynamic composites at the next lower level. These emergence/grounding links, then, can be seen as internal reference links, and the dynamic composite as the referent of the corresponding dynamic symbol. Although the physical symbol systems hypothesis is not usually stated in terms of a periphery-centre axis, the emergence/grounding interpretation captures at least a considerable portion of that what Vera and Simon have in mind when they talk about instances of internal reference (which they do only in passing). Explicitly, *"Symbols may designate ... patterns of sensory stimuli, and they may designate motor actions"* (Vera & Simon 1993a, p.9). This statement could be rephrased naturally in the dynamic symbol framework by substituting "emerge from" for "designate".

To sum up, symbol reference can be reconstructed in the dynamic symbol framework. External reference, with which the classical perspective is typically concerned, spans a wide empirical distance between the referent object and the (conceptual-level) dynamic symbol. The agent-internal stages in this wide-distance reconstruction are emergence/grounding transitions, which can be regarded as internal references.

Van Gelder and Port (1993) propose a systematic description of many kinds of informational entities, which they call *symboloids*. They present a multi-dimensional space of characteristics of such entities. Besides classical symbols, this space contains, e.g., neural firing patterns or attractors in complex dynamic systems. From this work, it becomes obvious that classical and dynamic symbols can be regarded as two extremes on a continuum of possibilities to describe "symboloid" informational entites.

A classical knowledge-based agent (such as Flakey) can be interpreted as a dynamic symbol structure. An example will clarify how this has to be done. Assume that the inference machine on board such an agent generates the motor command `set_speed +100`. This is how a classical AI researcher would state what is going on. He or she would further say that a composite consisting of the (classical) symbols `set_speed` and `+100` is generated. From the perspective of dynamic symbols, things look different. `Set_speed` and `+100` are no observables. Observables are the robot's movement, as it is affected by this command, or an on-off-pattern at the level of switching gates (given suitable monitoring equipment). Such observations can readily be interpreted in terms of `set_speed` and `+100` only if the observer knows how the inference machine is programmed, through all levels from machine œde via (say) Lisp to the high-level motor control language implemented on top of it all. An observer without such knowledge is in situation similar to a neurosurgeon observing overt behavior while monitoring neural activity. S/he will find it extremely hard to carve out from the rote 0-1-switching activity any "meaningful" patterns at all. Now, as a rule, the researcher *does* know how the inference engine is programmed. This knowledge enables him or her to identify just those patterns of 0-1-activity that correspond to the known programming constructs like `set_speed` and `+100`. In other words, this knowledge plays the role of a theory, which tells what entities there are to be looked for. In this case, the observation procedure consists in reading alphanumeric strings from a computer screen. The classical symbol, e.g. `set_speed`, belongs to the researcher's background theory, and it is justified by a reliable observation of a dynamic symbol, namely, a particular 0-1-pattern.

All in all, the classical worldview, as it is expressed in the physical symbol systems hypothesis, appears to be compatible with the dynamic symbol structure framework, when

- the customary platonic interpretation of symbols is traded for an empirical view, and
- the notion of symbol reference is suitably reconstructed in dynamic symbol structures.

When these measures are taken, dynamic symbol structures can serve as a framework to discussing agents programmed in a classical AI fashion.

In the remainder of the subsection, I briefly examine the "uppermost" level of processing, which for obvious reasons I shall call the *conceptual level*. At this place, I make only some preliminary considerations. The matter of conceptual-level information processing will be re-considered at greater depth in section 5.

The issue of concepts is loaded with unresolved philosophical questions concerning introspection, qualia, consciousness, social conventions, and the like (extensive discussion in Chalmers 1993, Maturana & Varela 1984, chapters 7-10). I do not venture to face the challenge of such issues. I merely hint at few distinctive properties of the conceptual level, which can be stated in the structuralistic framework of dynamic symbol structures:

- *Level-internal emergence*: dynamic symbols, which emerge from dynamic composites on the conceptual level, belong to that same conceptual level.

- *Comprehensive composition*: for any two conceptual-level dynamic symbols there exists a dynamic composite, in which the two co-occur.
- *Emergent cyclicity*: for any two conceptual-level dynamic symbols s, s', there exists a sequence $s = s_1, s_2, ..., s_n = s'$, such that s_i emerges from a dynamic composite that contains s_{i+1}.

I assume that in humans, dynamic composites made from conceptual-level dynamic symbols are characterized, in introspection, by some experience of gestalt. The term "gestalt" usually refers to "good" combinations of sensoric (in particular, visual) features. This is, however, not what I mean here. In the present context, by "gestalt" I want to refer to "coherent" combinations of conceptual-level dynamic symbols. What "coherent" means, must be left to the reader's intuition for the time being; the DSS formalism will provide a formal account of the term. For instance, a composite made from the dynamic symbols `Eve serpent apple Adam` is coherent; `Eve apple price` is incoherent. Then, the three above properties are motivated by the following observations concerning everyday human concept use.

First, the level-internal emergence property accounts for the observation that coherent conceptual-level composites can be "unitized" (the term is taken from Lightfoot & Shiffrin 1992), giving rise to a new concept. For instance, the coherent composite made from `Eve serpent apple Adam` can give rise to `fall-of-Man`.

Second, the comprehensive composition property is intended to capture that virtually any two concepts can be linked together in a coherent network of associations. For a demonstration, I take two concepts that are as distant from each other as can be, namely, the first and the last noun entry in Collins Concise English Dictionary: "aardvark" (an ant-eating mammal with a long snout) and "zymurgy" (the branch of chemistry dealing with fermentation, as in brewing). Figure 2.4 shows how these seemingly disparate concepts can be, in fact, associated with each other in a composite of convincing gestalt quality:

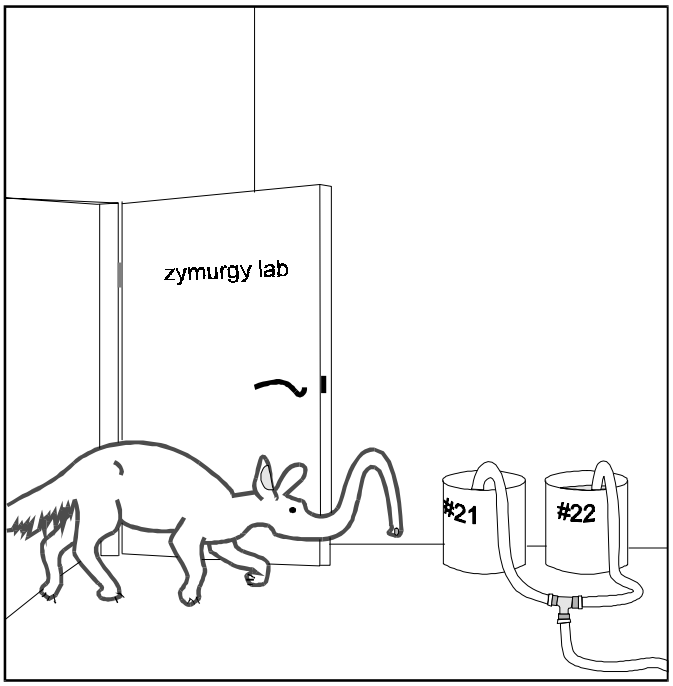


Fig. 2.4: The aardvark has become a plague in zymurgy labs.

Third, the emergent cyclicity property aims at the cyclic closedness of conceptual systems, which is revealed, e.g., in the ultimate self-referential nature of an encyclopædia.

Taken together, these properties distinguish the conceptual level from others by a strong flavor of internal cyclicity, connectivity, and closure. Fig. 2.5 attempts to capture this graphically.

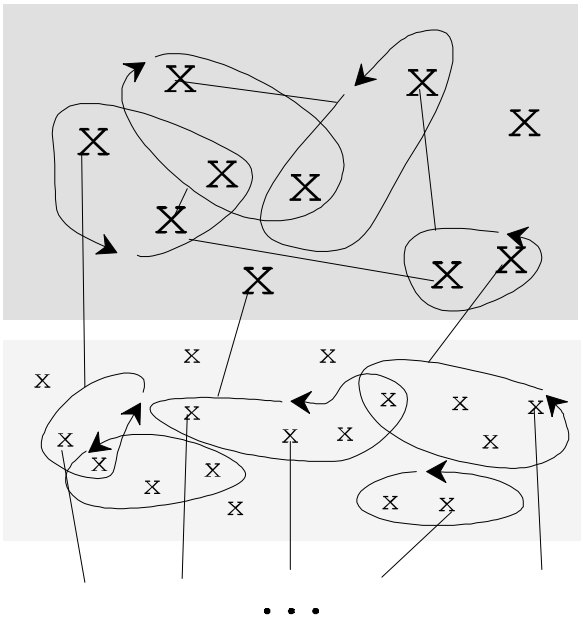


Fig. 2.5: An impression of the conceptual level (darker shading).

The emphasis on cyclicity, connectivity, and closure is a natural consequence of viewing an agent as a complex dynamic system. This distinguishes dynamic symbol structures from the perspective of classical AI, where the focus is less on global system properties and more on local (inference) mechanisms. However, the distinction is only a matter of degree and taste. It does not invalidate dynamic symbol structures as a perspective on classically programmed agents. Section 5 provides a more thorough discussion of the conceptual level in terms of both dynamic symbol structures and classical AI.

Summary of section 2.4

- The classical AI perspective, as codified in the physical symbol systems hypothesis, defines symbols as physical patterns that denote.
- The first part of this definition (physical patterns) is shared with dynamic symbols.
- External reference (e.g., from a cortical activation pattern to a cat "out there") is reconstructed in the dynamic symbol framework as a complex multi-stage process. The agent-internal steps in this process, i.e. dynamic symbols emerging from dynamic composites, are interpreted as instances of internal reference.

- Such an empirical notion of symbol reference is unusual for classical AI, where traditionally a platonic notion is implied. Arguably, however, this idealistic conception can be traded for an empirical one without harming the core of the physical symbol systems hypothesis.
- The uppermost, conceptual processing level is tentatively characterized by three properties that emphasize cyclicity, connectivity, and closure of conceptual systems.

2.5 Dynamic symbol structures and situated action

Proponents of situated action claim that intelligence rests on a fundament of non-symbolic agent-environment interaction. There are two principal objections against symbols. The first criticism concerns the *representational* implications of classical symbols, the second holds that not everywhere in an agent are *discrete* entities at work.

I turn to the anti-representational critique first. I have argued in the preceding subsection that symbol reference can be empirically reconstructed in dynamic symbol structures, and this reconstruction is necessary for a dynamic symbol oriented discussion of classically built agents. When the dynamic symbol framework is to be useful as an integrative view on both classical and situated action oriented agents, this empirical reconstruction of symbol reference must be compatible with situated action. I examine three versions of the anti-representational critique separately, and explain in each case why the dynamic symbol perspective is compatible with the respective author's view.

Greeno and Moore (1993) understand symbols as representational units on a high, conceptual level. They do not deny that symbols of this kind are important in cognitive processes, but they claim that cognition cannot be understood in a top-down fashion, *starting* from the idea of symbolic representations. Rather, cognition arises, in the first place, from immediate, non-symbolic agent-environment interactions. The authors adopt the theoretic approach of Gibson who argues that some kind of information *"is not perceived by creating cognitive representations ... but by a more direct process ... called* direct perception" (p. 52). Direct perception refers to an agent's direct reactions to the "affordances" (a Gibsonian term) of a physical situation, without recurring to symbolic representations that mediate between perception and response. Meaning comes prior to symbols, and is constituted by these agent-environment interactions: *"We treat semantic interpretation as something that people* do" (p. 50, my emphasis).

Symbols in the high-level sense adopted by Greeno and Moore correspond to dynamic symbols on the conceptual level. Then, their view on cognition is in agreement with the dynamic symbol structure framework. In both cases, high-level representational entities are included in the picture. The representational faculties of theses entities are explained, in both cases, in terms of physical, empirically observable agent-environment interactions. Both accounts share the non-platonic, non-aprioristic view on symbol reference. A difference remains in that the notion of symbols is restricted to high-level entities for Greeno and Moore, whereas dynamic symbols refer to the entire periphery-centre axis. Greeno and Moore do not include such an axis into their account.

Clancey (1993) distinguishes between a "first person" and a "third person" use of symbols. The former refers to an agent's conscious access to symbolic representations, which becomes causally effective, e.g. in planning. The latter refers to an external observer's view of the information processing in an agent. Clancey argues that these aspects are confounded in classical AI, and denies that "first person" symbol use can be modeled as a computation-like manipulation of pre-established symbols. Rather, "first person" representations are created during activity and cannot be separated from ongoing processes: *"...how you categorize the world,* arises together *with processes that are coordinating physical activity. To be perceiving the world is to be acting in it - ... - dialectically, so that what I am perceiving and how I am moving co-determine each other"* (p. 95). Clancey outlines a neurophysiologically motivated account of intelligent information processing, using many notions from current system sciences.

The system-oriented view, which takes neurophysiology as a guideline, is in a general good agreement with the dynamic symbol perspective. The two perspectives also agree in the empirical nature of representations. Clancey's position is, however, stronger than the dynamic symbol view. His emphasis on the dynamic, ever-in-the-change continuity of information processing, leaves no place for the emergence of discrete informational entities, which can be reliably and repeatedly identified. This anti-discrete aspect of Clancey's critique will be refuted further below.

As a last example, Maturana and Varela (1984) discuss a special case of symbols, namely, natural language words. They locate the "semantic content" of words neither within the agent using them, nor in external objects or facts. Rather, words appear in a unified dynamic system which comprises words, agents, and other objects. The semantic content of words is ascribed to them by external observers of the system *"*as if *that which determines the course of interaction* were *the meaning, and not the dynamics of a structural coupling of the interacting organisms"* (p. 223, translated from the German edition 1992[4], my emphasis).

Although Maturana and Varela's writing is often extremely abstract, it seems clear that in their view words should not be interpreted as representing objects or actions for their users. Rather, word meaning is an (unavoidable) artefact on the side of an external observer. This is, in fact, a philosophically abstract version of the empirical, theory-dependent reconstruction of symbol reference in the dynamic symbol framework. In both cases, symbol (or word) meaning is not a platonically given fact, but a contingent fixation which depends on how one looks at it. Maturana and Varela's view is more specific than the dynamic symbol perspective in that the authors imply a functional orientation on the side of the observer, and in that they ascribe meaning only to high-level, conceptually interpretable symbols. By contrast, the dynamic symbol approach allows to describe a semantic interpretation (i.e., a reference) to dynamic symbols on all levels, and the theoretic orientation need not be functional.

Summing up the anti-representational critique of these exemplary cases, it appears that it is mainly directed against the platonic notion of symbol meaning, and against the idea that such symbol meaning is causally effective in intelligent information processing. A contingent, theory-dependent empirical reconstruction of externalistic reference is not affected by this critique, since it does not ascribe causal effects to symbol reference, and thus treats it not as something that explains something else, but as something that must (and can) itself be explained.

The second kind of situated action critique of symbolic AI denies that *discrete* identifiable entities are effective in an agent, especially towards the periphery. This rejection comes in two forms, which I treat separately.

First, it is emphasized that an agent is coupled into its environment by *continuous* feedback loops of interaction. This suggests to use differential equations, instead of discrete symbol-manipulating techniques, for a formal account of an agent's functioning. Such techniques are advanced by Steels (1993a, 1993b).

I believe that the discrete vs. continuous issue can be resolved by cleanly distinguishing between two kinds of discreteness, which one might call *1-0-discreteness* and *identifiability*. A system of differential equations describes the temporal development of given parameters. The parameters change continuously; thus, they are not discrete in the particular sense of being on-or-off, i.e., 1-0-discrete, entities. But, the parameters that occur in a system of differential equations describe a finite set of distinct entities. These entities are separately identifiable by specific observation procedures. Thus, when discreteness is explained in terms of individual identifiability rather than in terms of a 1-0-characteristics, then the notion of symbols remains compatible with a continuous account of perception and action. The discrete nature of symbols is understood in this fashion in dynamic symbol structures.

The second kind of rejection of discrete identifiable entities derives from the argument that an agent's basic interactions with its environment form a complex dynamic system, where virtually everything modifies everything else, such that nothing remains stable and no entities can be *isolated*. This view is expressed by Clancey (1993) as follows: *"... the neural structures and processes that coordinate perception and action are created during activity..."* (p. 94), *"In contrast with the classical, symbolic architecture, the processors coconfigure each other"* (p. 94). The continuous, ever-changing nature of situated information processing is emphasized, which leads to the opinion that concepts have *"no inherent formal structure; cannot be inventoried"* (p.111). The complex systems argument is related to Maturana and Varela's notion of autopoiesis, and to the functional opacity of biological agents (cf. 2.3).

But, a closer look at complex systems reveals that they are more orderly than Clancey's description suggest. Research in physics and mathematics in the last decades has led to a deeper understanding of various phenomena of self-organization in complex nonlinear systems. The behavior of such systems is typically organized by attractor states, i.e., patterns to which the system's trajectory in its state space converges in the absence of perturbations. There may be infinitely many of such attractor states, they may be superimposed on each other on varying time scales, and they may be highly sensitive to changes in external "control" parameters. Learning or evolution can lead to a long-term change in the set of such observables. This great complexity may have motivated Clancey's judgement. It is nonetheless unjustified, since attractor states *can* be formally identified and hence, "inventoried". In evolving systems, the inventory may be subject to change in the long run, but this does not impair repeatable identification of each attractor for a certain time period.

An example of such a formal analysis of a complex system is Yao and Freeman's (1990) reconstruction of the olfactory bulb. The authors demonstrate how external stimuli give rise to (chaotic) attractor states in a neural network described by differential equations, and how sequences of stimuli lead to sequences of such states (cf. fig. 2.2). Steels (1993b) suggests a similar analysis for agent-environment systems.

Another important aspect of self-organization in complex systems is their spatial organization. A classical example is the emergence of convection cells in fluids (Haken 1983). The importance of compartimentation by membranes in biological cells, and the existence of specialized modules like the nucleus or mitochondria, is obvious (and emphasized by Maturana and Varela). On a larger scale, the behavior of agents can be interpreted in terms of typical

situations which are characterized in terms of places, behavior patterns, and participants. Although such situations rarely appear in reality in a "clean" fashion, they can in practice be identified by an observer. This possibility has enabled the use of various script and situation scheme representations in classical AI. Such schemes can be made flexible and open-ended without sacrificing identifiability (Schank 1982). The identifiability of dynamic sitatuations is also the fundament for the very notion of behaviors in behavior-oriented AI.

In sum, complex systems can be described in terms of individually identifiable temporal, spatial, or spatiotemporal phenomena. Such descriptions may be difficult to achieve, ambiguous, incomplete, and they may be subject to a long-term drift, but this does not invalidate the effort in principle. The anti-discrete argument on the grounds of a complex systems argument can, I believe, be rejected.

The anti-discreteness critique can also be refuted from a more fundamental perspective. Every empirical scientific theory, and every engineering discipline, fundamentally builds on a particular set of observables. Progress in research is typically coupled to the introduction of new observables, with new detection or construction procedures. An empirical or engineering approach without observables is inconceivable. They appear in the corresponding theories as technical terms or as mathematical symbols. Seen from this global perspective, the existence of identifiable entities is a necessity for any description of reality.

Summary of section 2.5

- The situated action critique against symbols has an anti-representational and an anti-discrete aspect. The first kind of critique does not affect the empirical reconstruction of symbol reference in dynamic symbol structures.
- The anti-discrete critique can be reconciled with dynamic symbols by distinguishing between 1-0-discreteness and discreteness in the sense of repeatable identifiability.
- Complex systems almost universally exhibit attractor states. They are a standard type of dynamic symbol observables.

2.6 Differentiation, resolution, and specialization

To recapitulate things said in 2.2, dynamic symbols are defined in terms of specialization/abstraction, i.e., the information afforded by an observation. This information has two components, one of which is due to the entity observed (differentiation/de-differentiation), and the other to the observation procedure (high/low resolution). The relative contributions of these two components are not discriminated in the determination of a dynamic symbol. However, when an increase in resolution does not yield an increase in information (i.e., an increase in specialization), then one is justified to assume that the obtained information reflects nothing but the entity's differentiation, i.e., the information about the entity "as it is objectively there".

A dynamic symbol appears in a researcher's theory as a *formal symbol*, which can be anything from a mathematical symbol (when the theory is highly formalized) to a telling name (when the

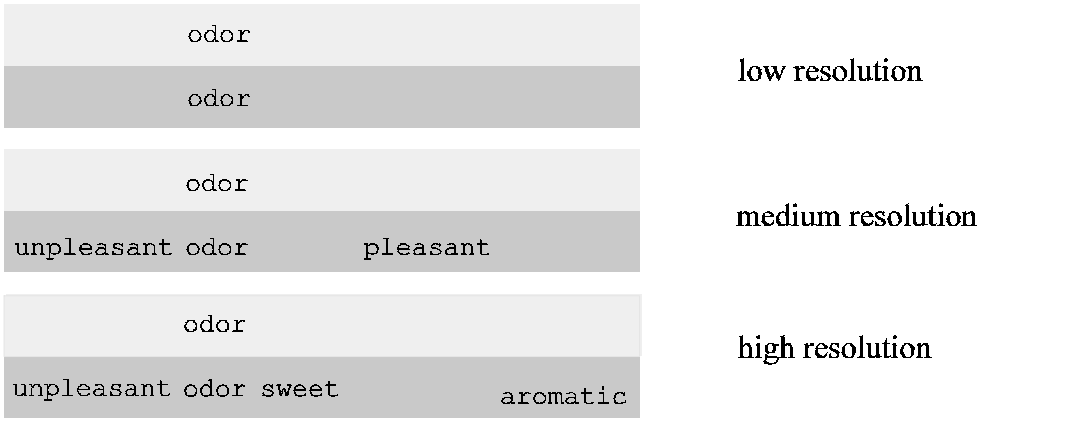
theory is in a pre-formal stage). Different dynamic symbols are handled by different formal symbols, i.e., the abstraction dimension is mirrored in a variety of formal symbols.

A closer look at an example will be instructive. The example consists in observations of a system capable of classifying odors. I compare a biological with an artificial system: the olfactory bulb (as in Yao & Freeman 1990), and a (fictive) apparatus whose chemosensor readings are interpreted by a symbolic, knowledge-based algorithm. I treat the olfactory bulb first. I assume that the researcher observes oscillatory patterns, which correlate with the (known) qualities of the stimulus. Three observation procedures are used.

A low-resolution procedure provides information amounting to nothing more than deciding whether any oscillatory pattern is or is not present. Since this is found to correlate with the presence or absence of an olfactory stimulus of arbitrary quality, the pattern (as observed with the low-resolution procedure) is given the formal symbol `odor` in the researcher's theory.

A medium-resolution procedure reveals three kinds of patterns: the first is a highly de-differentiated one, which is observable when the olfactory bulb is in a responsive but very noisy state. It is again called `odor`, since it provides the same information as the low-resolution observation. This pattern is found when the stimulus is ambiguous, or when the bulb is in a highly excited, "computationally hot" state. When the bulb is in a "cooler" state, there appear three patterns. First, the de-differentiated `odor` pattern is repeated when the stimulus is ambiguous. Second and third, two more differentiated patterns, which are given the telling names `unpleasant` and `pleasant`, can be observed. They correlate with the (known) qualities of stimuli.

a)



b)

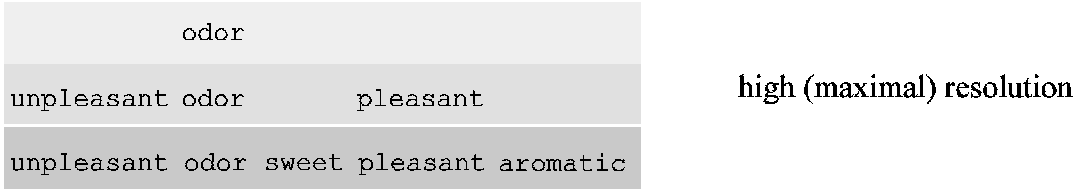


Fig. 2.6: Dynamic symbols observed under various conditions. Darker shading corresponds to system state that allows higher differentiation.

Finally, a high-resolution procedure again yields the highly de-differentiated `odor` pattern for excited states or ambiguous stimuli. In a "cooler" state, the `unpleasant` pattern is repeated, whereas the `pleasant` pattern is further resolved into `sweet` and `aromatic` patterns. The `pleasant` pattern is no longer observable: it turns out that even when the stimulus is a mixture of sweet and aromatic, the bulb responds by settling either into the `sweet` or into the `aromatic` pattern (and possibly fluctuates between the two). Fig. 2.6a illustrates the situation.

The situation is different in the case of the assumed artificial system. The observation of "response patterns" has always maximal resolution: system response states are read off the computer's screen, where they appear directly as alphanumeric strings `odor`, `unpleasant`, `pleasant`, `sweet`, and `aromatic`. It is important to note that these strings belong to the observation procedure; they do not coincide with the corresponding dynamic symbols that are causally effective in the device. The dynamic symbols are physical bit-patterns in the CPU. That they are observable with maximal resolution is due to the system's artificial nature, where a 100% precise theory is available, which ensures that only certain (five) dynamic symbols can actually occur, and that they can be reliably observed through the screen output. The system can (in this fictive example) be set to different levels of recognition precision (motivated, e.g., by a tradeoff between response time and precision). In the lowest precision mode, only absence or presence of a stimulus is detected by the system, i.e., an `odor` response will or will not be observable. In a medium mode, `odor`, `unpleasant`, and `pleasant` are possible observations (depending on ambiguity and character of stimulus). In the highest precision mode, all fiveresponses are possible: `odor` for completely ambiguous stimuli, `unpleasant` for unpleasant stimuli, `pleasant` for ambiguously agreeable stimuli, and `sweet` and `aromatic` for sweet and aromatic stimuli, respectively (fig. 2.6b).

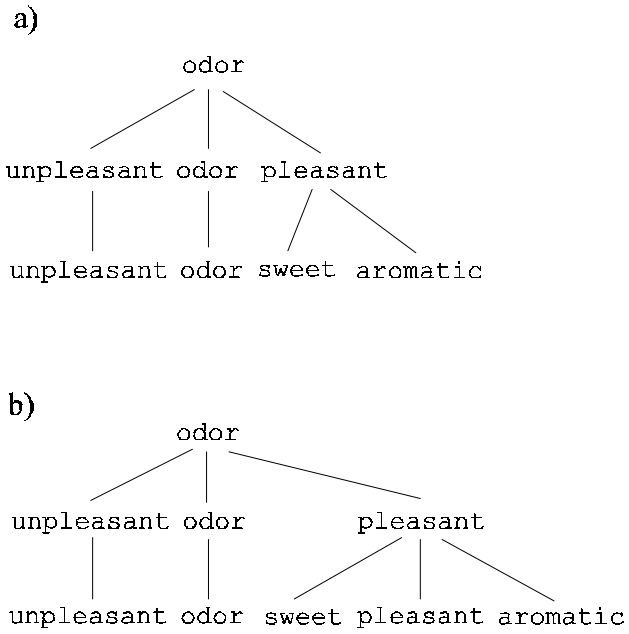


Fig. 2.7: Abstraction trees derived from the diagrams in fig. 2.6.

In Fig. 2.6, the observations are structured in a way that accounts for both differentiation and resolution. Both sources of information are inseparably superimposed in the scale of abstraction. This leads to the notion of an *abstraction tree*, which is derived from diagrams like fig. 2.6 by collecting all different "observation bands" (shaded rectangles in fig. 2.6) and ordering them in a suitable fashion (fig. 2.7). The resulting tree accounts for the purely abstractional aspects of possible observations. Information about the relative contributions of differentiation and resolution is lost. As a consequence, the same abstraction hierarchy can be the result of different "complete" diagrams of the kind of fig. 2.6. Note that the same dynamic symbol (more precisely, the formal symbol used for it on the theoretician's side) can occur at several levels in an abstraction tree.

The observation of a relatively abstract dynamic symbol can be due to a relatively low resolution and/or a relatively low differentiation. When it is *only* due to low resolution, an increase in resolution will make the abstract dynamic symbol vanish from the scene. This happens with `pleasant` in the case of the olfactory bulb case, but not in the artificial system. In the first case, the `pleasant` pattern is an example of a *disjunctive* dynamic symbol. It is observed because "objectively" the pattern corresponding to a sweet stimulus *or* the pattern corresponding to an aromatic stimulus is present. When a disjunctive dynamic symbol is specialized in an abstraction tree (e.g., `pleasant` to `aromatic` and `sweet`), it does not reappear at the more specialized level in the hierarchy. The dynamic symbol perspective thus provides both an empirical explanation and a formal characterization of disjunctive categories.

The picture is still too simple in an important aspect. Both differentiation and resolution, which I have treated as "objective" scales so far, depend on a background theory that tells the researcher what to look for, and how to increase resolution. The objectivity test sketched in subsection 2.2 (i.e., when an increase in resolution does not yield more information, then what is observed is "objectively there") is not, in fact, truly objective for two reasons. First, the statement that an increase in resolution does not provide more information depends on how the data are interpreted. "Not more information" should be better read as "not more information with respect to the theoretic perspective taken". When a noisy spike train is found equally noisy after a refinement of the observation procedure, one has implicitly used some particular operationalization of the notion of noise. It may be the case, e.g., that the refined procedure actually reaveals some new kind of long-term spike correlation, only it escapes the theoretic filter used at the moment. Second, the very method by which one increases resolution is also theory-dependent. For instance, one might narrow some numeric interval of confidence by allowing a longer observation time, or one might make the procedure more sensitive to weak signals. Which way one chooses for increasing resolution again depends on a theoretic background that tells which way is appropriate.

Thus, one should always be aware of the fact that talking about observations as indicating "objective entities" remains a metaphor. This way of talking and thinking is, however, so deeply engrained, that a more correct treatment of observables is likely to appear forced and unnecessarily "epistemology-loaden". Therefore, I usually think and talk myself about dynamic symbols in a loose way, as if they were objectively there, and be contended with the relative justification afforded by the above-mentioned objectivity test. It works as long as a particular background theory is practically sufficient for the purposes at hand. The prize for this convenience is that one runs the danger of turning blind for interesting alternatives in explanation or design.

Dynamic symbols interact in an agent in various ways. For instance, a dynamic symbol can influence the dynamic composite in which it grounds in a top-down fashion; it can itself be influenced by the composite in a bottom-up fashion; and the dynamic symbols within a dynamic composite can interact with each other "laterally". In order for one dynamic symbol, say *s*, to be influenced by another, say *s′*, there must be some information passed from *s′* to *s*. In dynamic symbol structures, this information is understood exactly as the information obtained by a dynamic symbol by an external observer, i.e., *s* "observes" *s′*. The notions of differentiation, resolution, and abstraction carry over to the system-internal kind of "observation". Differentiation concerns the observed dynamic symbol "as it is", resolution concerns the quality of the observation, and abstraction concerns the net information gain. System-internal resolution can be poor e.g. due to noisy neural projective pathways in biological systems, or due to low-bandwidth signal channels or symbolic simplification procedures in artificial systems. What *s* "knows" about *s′* is reflected by the position of *s′* in an abstraction tree. This view on information processing agrees with the original intentions of information theory, where the quality of channels (here: resolution) is a central concern.

Thus, the introduction of abstraction trees is not merely of epistemological relevance, as describing what an external observer knows about a system. Abstraction is also crucial for the internal dynamics of dynamic symbol structures. Actually, two such hierarchies are necessary for a full account of an agent (more precisely, of one of its levels in the periphery-centre hierarchy). The first concerns the dynamic symbols as they are observed by the researcher, the second, as they "observe" each other. But, since the way dynamic symbols observe each other must itself be observable by the researcher (otherwise it would escape a scientific treatment), the researcher's abstraction tree should ideally cover the system-internal one. For the sake of simplicity, I assume that both coincide, and speak of "the" abstraction tree.

For the remainder of this subsection, I propose a way of looking at bottom-up vs. top-down processing.

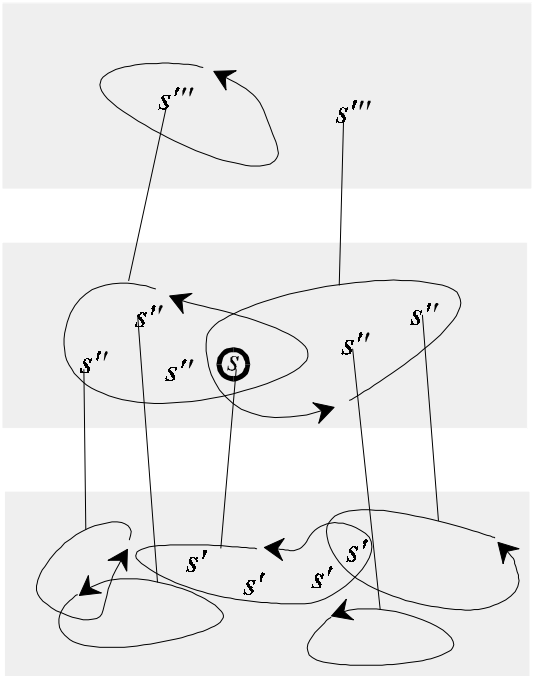


Fig. 2.8: Interaction partners of a dynamic symbol *s*.

The differentiation of a dynamic symbol can be assumed to be affected by the abstraction of all other dynamic symbols with which it interacts. There are three principal kinds of interaction partners of a dynamic symbol *s*: first, members *s′* of the dynamic composite in which *s* grounds; second, partners *s″* of *s* in dynamic composites; third, dynamic symbols *s‴*, which emerge from composites in which *s* participates (fig. 2.8).

A general tendency can be expected in many cases, namely, that the abstractions of interacting dynamic symbols correlate with each other, i.e., a specialization of *s* induces (as a tendency) a specialization of *s′*, *s″*, *s‴*, and vice versa.

For an example, consider an artificial speech understanding system, where syllable hypotheses *s″*, *s* are combined to yield a word hypothesis *s‴*, and where the syllable hypotheses are derived from compositions of still more basic feature hypotheses *s′*. A syllable (or word, or feature) hypothesis is a set of possible readings. The smaller the set, the more specialized the corresponding dynamic symbol. Specialization is maximal when there is only one reading left, which is the ultimate goal of processing. In this example, maximal specialization of the word hypothesis *s‴* will induce a likewise maximal specialization of the syllable hypotheses *s″*, *s*, since a word determines its syllables. Vice versa, a maximal specialization of the syllable hypotheses will lead to a maximal specialization of the word hypothesis, since syllables determine "their" word. Analogically, the specialization of feature hypotheses correlates in both ways with the specialization of the syllable hypotheses, although the correspondence might be weaker here, since a syllable can possibly be pronounced in different ways, which means that feature readings need not be uniquely determined by a syllable reading. Thirdly, an increase in specialization of some of the syllable hypotheses *s″* may induce a specialization of the syllable hypothesis *s*, since readings of syllables that co-occur in a word constrain each other.

This suggests the idea of an *abstraction gradient*. When dynamic symbols are relatively specialized compared to their higher, lower, or lateral neighbors, the latter are induced to follow in their specialization. Bottom-up processes are, in this view, induced by a relatively higher specialization of the lower level; top-down processes are induced in the inverse case. Both directions have equal rights in dynamic symbol structures, and a third, lateral direction of influence is added to the picture. This view emphasizes the system nature of multi-level information processing, where all informational entities influence each other. In section 3.3, a mechanism for a "lateral" abstraction gradient mechanism is provided, and in 3.4, I sketch a realization of a level-crossing abstraction gradient mechanism.

The idea that the direction of processing is determined by an abstraction gradient has not, of course, the status of a law. It captures a tendency of effects of processing mechanisms, as they are due to be found in many cases.

Summary of section 2.6

- Dynamic symbols appear in the researcher's theory as formal symbols.
- Formal symbols reflect the information gained by an observation, regardless of whether it is due to the observable "as it exists objectively" (differentiation), or to the precision of the observation procedure (resolution).

- This net information yields the dynamic symbol's level of abstraction, which makes abstraction trees a fundamental notion in the approach.
- Differentiation and resolution are theory-dependent notions and must be used with care.
- The account of external observations (in terms of differentiation, resolution, and abstraction) carries over to a notion of system-internal observation. This paves the ground for an information theory oriented understanding of the interaction of dynamic symbols.
- An abstraction gradient helps to understand the direction of information processing (bottom-up, top-down, or lateral).

2.7 Emergence and grounding

Emergence and grounding are established notions in the methodology of AI and computation theory. In this subsection, I justify the adaptation of these terms for the relation between a dynamic symbol and a lower-level dynamic composite.

The term "emergence" characterizes a particular outlook on computation, namely, *emergent computation*, which is by now an established paradigm (Forrest 1989). The basic idea is that useful computations can be performed by making a collective of informational entities interact in a fashion that is not premeditated or controlled in detail. Typically, the entities and the interactions are comparatively simple, and the latter are defined in a local and parallel fashion. The collective is expected to self-organize in some way or other. The desired result of the computation is read off, not at the level of the basic interacting entities, but at the coarse-grained level of patterns that "emerge" from the collective. Prominent examples of emergent computation are connectionist networks, classifier systems, and cellular automata. The collective behavior of (simple) computational "agents" is sometimes also included in the paradigm (Shoham 1993).

In a slightly different vein, the term "emergent" is used in the behavior-oriented robotics community to denote the fact that an undesigned behavior results from the interactions of other behaviors (which are explicitly designed or can be emergent themselves). An example is wall-following behavior, which emerges from a move-forward and an avoid-obstacle behavior under suitable conditions.

This usage of "emergence" differs from the computation theoretic usage in that the number of interacting entities is comparatively small (in emergent computation, it is typically large), and in that the emergent entity belongs to the same level of description as the entities from which it emerges (in emergent computation, there is a transition between levels of granularity implied). Both usages of the term are similar in that the emergent phenomenon is unpredictable, dynamic, to some degree robust against perturbations, and requires the introduction of a new descriptive category on the observer's side. The latter point is particularly important. For the behavior-oriented version of "emergence", Steels (1994, p.5) makes it part of the term's definition:

"A behavior is emergent if it can only be defined using descriptive categories which are not necessary to describe the behavior of the constituent components."

In dynamic symbol structures, the term is used in a way that is compatible with its usage both in emergent computation and behavior-oriented robotics. As in both of these, the notion characterizes the occurence of a detectable, organized pattern (i.e., a dynamic composite) that arises from collective interaction of entities (i.e., dynamic symbols within one level). Unpredictability, dynamic nature, and relative robustness of such patterns are likewise emphasized. The origin of new descriptive categories is also a central topic. It will be discussed in subsection 2.10.

Where emergent computation and behavior-oriented robotics differ in the usage of the term, the dynamic symbol perspective takes an intermediate or unspecified stance. This concerns, first, the number of dynamic symbols that make a dynamic composite. It can be relatively small. For instance, one might expect the (ominous) number 7±2 for conceptual-level dynamic symbols interacting in short time memory. By contrast, the number of retinal receptor cell responses that interact to make a dynamic composite corresponding to, say, a red blob, can be very large. Second, a dynamic symbol can belong to a higher level than the dynamic composite from which it emerges. This corresponds to the emergent computation perspective, and it is the case when an emergence event corresponds to a transition between levels of the periphery-centre dimension (as in fig. 2.1a). A dynamic symbol can also emerge from dynamic composites, all withinone level, as in behavior-oriented robotics. This is the case, e.g., for conceptual-level dynamic symbols (as in the upper portion of fig. 2.5). Intermediate cases are also possible, where emergence gives rise to what might be called a gradual level ascent (as in fig. 2.1c).

Now I turn to grounding. The notion occurs in a debate at the philosophical borders of AI. The "symbol grounding problem" (Harnad 1990), actually an entire problem family, starts from the question whether computers demonstrate to possess a mind when they solve problems in a fashion that looks human-like from the outside (e.g., answering questions in Chinese, as in the famous Chinese room argument). This can be restated in the question whether cognition can be reduced to a purely syntactic manipulation of symbols, or whether symbols must have an intrinsic, system-internal meaning, the latter being that in which symbols are then said to ground. As a side branch, this leads into a discussion of the role of consciousness and the experience of qualities (Chalmers 1993). A more modest attempt to settle the issue is to claim that symbols in truly cognitive agents derive their intrinsic meaning from being grounded in a "subsymbolic" substrate. Harnad proposes "iconic representations" and "categorical representations" for the lowest, subsymbolic level of cognition. The former directly derive from sensory projections (e.g., retinal images), the latter from automated processes that convert such projections mandatorily into basic object and event categories ("categorical perception", cf. Harnad 1987). Both Harnad and Chalmers (1990) suggest to use connectionist networks for dealing with the technical side of this endeavor.

The arguments of Harnad and Chalmers boil down to the idea that symbols can have an intrinsic meaning, when they are not treated as atomic units but as structured patterns. The information contained in the pattern constitutes that matter in which symbols ground. Seen this way, grounding is exactly the inverse of emergence, which has motivated the usage of the term in dynamic symbol structures.

One should be aware, however, that the problem of symbol grounding has other aspects, many of which are of a more philosophical nature. Some ramifications are discussed in Chalmers (1990).

Summary of section 2.7

- In the emergent computation paradigm, "emergent" denotes the formation of patterns in a collective of many, simple, interacting computational units.
- In behavior-oriented robotics, the term characterizes new behaviors that result from interactions between a few given ones.
- In one of its interpretations, the term "grounding" refers to symbols as being internally structured entities.
- The usage of emergence/grounding in the dynamic symbol framework essentially conforms with these three usages, adding the aspect of an iterated hierarchy of formally similar descriptive levels.

2.8 The structuralistic nature of dynamic symbol structures

In the preceding subsections, I have repeatedly mentioned that dynamic symbol structures provide a *structuralistic* view on agents. Now I explain what this means.

"Structuralistic" is not a well-defined term. It characterizes a large family of approaches to the epistemology of science in general, or of scientific disciplines in particular, notably in linguistics, ethnology, psychology, and sociology. Roughly, the structuralistic way of doing (or explaining) science is characterized by a highly abstract, often mathematically formal outlook on the discipline's subject matter, which enables the researcher to detect fundamental inter-instance similarities between various realizations of the subject matter (e.g., between languages, social systems, mathematical theories). The abstract descriptions, called *structures*, should be self-supporting and closed in the sense that the described systems are revealed as integrated wholes on purely formal grounds, i.e. without recurring to the particular things and mechanisms that are observable in individual applications of the scientific discipline. This is expressed by Piaget (1968, p. 6, my translation) as follows:

"... one finds at least two aspects that are common to all structuralistic approaches: on the one hand, an ideal or a hope of an intrinsic intelligibility, which is founded in the postulate that a structure is self-sufficient and does not require for its understanding a resort to some elements that are alien to its nature; on the other hand, realizations [...] [where the interpretation in terms of structures] *reveals characteristics that are general and apparently necessary, in spite of their variety* [across different realizations]*."*

For dynamic symbol structures, I adopt the notion of structuralism that Piaget develops as a specialization of this general account. A structure, then, is characterized as follows (p. 6f):

"In a first approximation, a structure is a system of transformations, being lawful as a system (in contrast to the properties of its elements) and conserving or enriching itself by these very transformations, without the latter leading out of the system [...]. *In a nutshell, a structure is characterized by its totality, its transformations, and its self-regulation.* [...] *In a second approximation,* [...] *a structure must lend itself to a formalization."*

After this preliminary sketch, Piaget explains the notions of totality, transformations, and self-regulation in some detail. I give a brief summary:

- The aspect of *totality* means that a structure has to be understood as a systematic whole, not as a collection of components. Piaget makes it a point that the structure's totality is not an irreducible, "gestalt"-like property. It has to be explained by tracing the relations and dynamic laws that bind the components together.
- The notion of *transformations* emphasizes the dynamic character of the regularities that define a structure. Piaget opposes attempts to define structures by virtue of some static form, or by timeless laws. He adopts a "constructivist" view (the term is used by Piaget himself), where a structure arises from temporal operations of some kind. The nature of these transformations is not further specified; Piaget's standard example is the operation that gives rise to mathematical group structures.
- Finally, the aspect of *self-regulation* concerns the "cybernetic" character of the transformation, which warrants that the structure persists in time without dissolving. Again, no specific definition of the term is given. Piaget refers to examples like feedback mechanisms in cybernetics, physiological homeostasis, and biorhythmic patterns.

All in all, this picture of a structure corresponds largely to what one would nowadays call a (closed) self-organizing dynamic system. The difference mainly lies in formal generality. The formal nature of structures is virtually unrestricted, whereas the notion of self-organizing dynamic systems typically connotes a physical view, implying the use of differential equations and methods from thermodynamics.

Dynamic symbol structures are Piagetian structures in the sense of his "first approximation". They provide an abstract account of a class of systems, satisfying the requirements of totality, transformations, and self-regulation.

The *totality* of dynamic symbol structures can be seen from two angles, which are alternately adopted by Piaget. First, when one assumes an atomic lowest level, and a cyclically closed highest, conceptual level, the resulting dynamic symbol structure has a lower and an upper boundary. Thus, it is a "totality" by virtue of being bounded. Second, even when a dynamic symbol structure is constructed as a hierarchy that is open-ended on both sides, it can still be described within a closed theoretic framework (Piaget gives the arithmetic system of the whole numbers as an example for a theoretically closed, yet open-ended structure). Seen from this angle, dynamic symbol structures are totalities by virtue of being described uniformly across levels.

The aspects of transformations and of self-regulation are reflected in dynamic symbol structures in the assumed interactions between dynamic symbols. There are two kinds of interactions. First, within one level, they give rise to dynamic composites in a self-organizing fashion. Second, there are top-down and bottom-up interactions between levels, which are due to the emergence/grounding relations between dynamic symbols and dynamic composites.

Self-regulation can be interpreted as the system's top-level "task" to maintain itself (compare the comment at the end of 2.3). It is the only task with which a structuralistic account is concerned. Tasks (goals, functionalities) are not part of a structure in the structuralistic sense. They only appear in the more concrete considerations of what a given structure can be used for, or how it adapts to its environment. Considerations of this kind are, however, claimed by Piaget a *necessary* supplement to a structuralistic account of a system.

Transformations and self-regulatory mechanisms are not further specified in the "first approximation" dynamic symbol structures framework, as it is used in this section. These notions become fully specified in the DSS formalism, which, therefore, is what Piaget calls a "second approximation" structuralistic theory.

Structuralistic approaches have traditionally been motivated from within linguistics, cognitive psychology, group psychology, and sociology. This connects them directly with the issue of agent modeling. In particular, Piaget has developed his epistemological ideas on a background of research in cognitive development. Although I do not further pursue this route in this thesis, it can be hoped that AI can make a connection to this tradition. A large step in this direction has already been taken by Drescher (1991).

Summary of section 2.8

- I adopt the notion of a structure as it is defined by Piaget by the three characteristics of totality, transformations, and self-regulation.
- Dynamic symbol structures are abstract "first approximation" structures, and the DSS formalism leads to specific "second approximation" structures.
- The notion of a structure is closely related to the modern notion of a self-organizing dynamic system.

2.9 Self-organization and compositionality

In this subsection, I argue that both compositionality and self-organization are necessary aspects of information processing on all levels of the periphery-centre axis. I explain how I understand these notions, and relate them to particular shortcomings of the classical and the behavior-oriented approach to agent modeling.

I shall first take a closer look at self-organization. The term has no standard interpretation (historical overview and outline of present usage in Krohn, Küppers & Paslack 1987). Typical notions connected with self-organization are feedback, dissipation of energy, time arrow, attractor state, collective behavior, and others. Formal disciplines concerned are e.g. thermodynamics, statistical thermodynamics, theories of automata and formal languages, fractal geometry, population dynamics, and more dedicatedly, chaos theory and synergetics (a collection of papers that highlights this interdisciplinarity is provided by Dress et al. 1986).

In the context of AI, some particular connections between the perspective of self-organization in complex systems and intelligent information processing are worth mentioning. Gestalt psychology can be considered as a manifestation of self-organizing principles. Viewing reasoning as a dynamic, self-organizing process can be traced back at least to Bartlett (1932), who described how memory elements aggregate in a way that would be called self-organizing today. Piaget's theory of the emergence of object concepts in infants via the "primary and secondary circular reaction" is a clear case of an agent/environment feedback loop; generally, Piaget's way of thinking is system-oriented (a standard introduction to Piaget's many-faceted

psychological research is Flavell 1968). Consequently, his work is taken up by situated action oriented AI; Drescher (1991) integrates it with ideas from radical constructivism. Lashley (1951) examined the problem of how spatially distributed representations can give rise to temporal phenomena. Long before the rise of connectionism, he envisioned the brain as a complex system made from many interacting, "recurrent" (his term!), incessantly active, oscillation generating subsystems.

Under the impression of more recent developments in the system sciences, interest in complex systems is rekindled in cognitive science and linguistics (e.g., Krause 1989, Port 1990, van Leeuven 1990, Strohner & Rickheit 1991, van Gelder & Port 1993). This is also witnessed by an exploratory "Conference on Dynamic Representations in Cognition" at the University of Indiana in 1991 (no proceedings). In connectionist research, analytic methods from various dynamic systems approaches are routinely used. Classifier systems are originally motivated by considerations concerned with dynamic adaptation (Holland 1975). They can be considered a formal account of self-organization in its own right, but they also lend themselves to an analysis in terms of other dynamic systems models (e.g., Forrest & Miller 1990).

In the face of this diversity, fixing a particular notion of self-organization would be arbitrary and overly restrictive. Instead, I will use the term in a broad manner for any information processing mechanism that shows most or all of the following characteristics: (1a) equilibration towards an attractor state in the absence of perturbations, (1b) internal feedback cycles, (1c) acceptance of a (quasi-)continuous stream of input, (1d) integration of multi-channel input, (1e) high degree of parallelism, (1f) local interactions of a collective of informational entities, and (1g) robustness against incomplete and noisy input. Currently, such mechanisms are used preferably at the periphery of agents; they are often closely coupled to the agent-environment interactions; they are typically realized by connectionist networks, classifier systems, analog circuits, or populations of simple C-procedures; and they are characteristic for behavior-oriented approaches.

The other relevant notion, compositionality, is likewise ill-specified. In a classical AI setting it usually denotes the building of complex symbol structures from simpler ones, by mechanisms like symbol concatenation, logical junction of formulae, construction of graph-like representations (e.g., Sowa 1992), or coupling of dedicated processing modules (e.g., Minsky 1985). The connectionist community, after having been challenged by Fodor and Pylyshyn (1988), has come up with several compositional mechanisms, e.g., by variable link weights or temporal correlation of spike trains (e.g., Shastri & Ajjanagadde 1992). A discussion of connectionist compositionality is given by Chalmers (1992), and a systematic catalogue of compositional mechanisms in general is proposed by van Gelder and Port (1993). Compositionality can be achieved both in a static, "spatiostructural" way, and in a dynamic, temporal fashion. These two aspects are sometimes alternative aspects of the same thing (e.g., a sentence is temporal when uttered and spatial when written), and sometimes they are inherently coupled (e.g., cortical activation patterns).

I understand an information processing mechanism as compositional when it exhibits all of the following traits: (2a) ability to rapidly build and to decompose information structures with arbitrarily many components and/or arbitrarily compositional depth, (2b) causal relevance of composition-forming phenomena for the functioning of the mechanism, (2c) the composing of components is a physical effect, and (2d) constituents can to some extent be arbitrarily selected to make ad-hoc composites.

The qualification "rapidly" in (2a) is crucial. It rules out an evolutionary, asymptotic buildup of compositional structures, like spatially compositional cortical feature maps (Obermayer, Ritter

& Schulten 1990), or clusters of classifiers linked together in a feedback loop of the bucket-brigade algorithm (see below). (2b) and (2c) are meant to exclude purely epiphenomenal compositionality. This will later turn out as important for understanding composition of behaviors. Compositional mechanisms in the sense of (2a) - (2d) are mostly implemented towards the central side of the periphery-centre axis; they are often to some degree detached from the agent-environment interactions; they are typically realized by classical symbolic techniques; and they are characteristic for classical approaches to agent modeling.

To be sure, self-organizing and compositional aspects are integrated to some extent in existing approaches. However, the integration is still superficial. I shall examine some important examples, explaining in each case in what sense it falls short of the goal.

An integrative aspect is obvious in localist networks, where nodes are directly labeled by symbols. This method is often found in neural network applications in cognitive science, linguistics, and at the borders of symbolic AI. A much-cited classic is the proposal of Waltz and Pollack (1985). They implement positive and negative contextual correlations between words as weighted links between labeled nodes, achieving that the correct reading of semantically ambiguous words in the presence of contextual clues is found via equilibration of activation in the network. Shastri and Ajjanagadde (1992) present a highly refined localist architecture, which realizes robust, rapid semantic-network-like classification and scales up to larger terminologies in sublinear time. They exploit, among other effects, temporal correlations of spike trains. - The dynamics of these localist networks is clearly self-organizing in the sense indicated above. They are also compositional in that discrete informational entities (i.e., labeled nodes) become linked to each other: by link activation in Waltz and Pollack's work, or by spike correlation in Shastri and Ajjanagadde's. However, in these examples, like in other localist network approaches, the composite structures are simple and predetermined, which disagrees with (2d). Composition cannot be iterated to yield second-order composites, which conflicts with (2a). Also, the dynamics in these examples drives the system through an equilibrative one-way trajectory, which is not in accordance with the reversibility required in (2a).

Another connectionist approach to compositionality is Smolensky's (1986) "harmony theory". It will be reviewed in detail in section 6. Smolensky's architecture accounts for compositional structures of unrestricted compositional depth and of arbitrary numbers of components. It falls short, however, of reversibility due to its one-way equilibrative dynamics. Also, possible compositional links between components are fixed beforehand by hand-coding or learning, which disagrees with the ad-hoc spirit of (2d).

Classifier systems (Holland 1975, 1986) offer a quite different route towards an integration of self-organization with compositionality. A classifier is a discrete, rule-like piece of information. By virtue of the "bucket-brigade" algorithm, chains of classifiers that contribute to a given task are reinforced. Parallel to the bucket-brigade algorithm, the pool of available classifiers is incrementally modified by deleting unsuccessful classifiers and mutating/copying successful ones. The bucket-brigade and the genetic algorithm together lead to the formation of relatively stable, successful chains of classifiers, which in turn can further be stabilized by forming feedback loops. The latter can be considered as relatively well-defined composite structures. Patel and Schnepf (1991) view them as complex concepts. The dynamics of classifier systems is clearly self-organizing, and the feedback loop composites can in principle contain arbitrarily many components, and can themselves iteratively be coupled into higher-order "hypercycle" structures. But, the linking mechanism through the bucket-brigade and genetic algorithm is slow (even awkwardly so, as emphasized by Forrest & Miller 1990). This slowness is connected to the classifier system background philosophy of genetic adaptation and optimization. The linking mechanism is coupled to the success of overall system behavior. By

contrast, the intuition behind compositionality, as I want to fix the term, is that components can be put together and taken apart in a free, "playful", tentative, or explorative way.

Examples for an integration of self-organization with compositionality are harder to detect in classical AI. Classical AI has, of course, no difficulties with compositionality. But self-organization is alien to inferential logics. Most classical symbolic reasoning systems are built on the fundament of some high-level deductive calculus, where sequences of inferences follow the routes of an explicitly formulated control strategy. In order to enable *self*-organization, by contrast, the inferencing would have to be left to itself in some way or other. Thus, the only examples of self-organization in classical setups seem to occur in cases where a high-level logical calculus is not involved, as in spreading activation (e.g., Mehl 1992) or marker-passing (e.g., Charniak 1983) techniques. Such approaches are typically applied to problems where "soft", evidential constraints must be satisfied, as in word sense disambiguation. The systems of Mehl and Charniak, and others of the same basic category, are in principle similar to the localist network of Waltz and Pollack, which I have discussed above.

I believe that self-organization must ultimately be missed by logic-oriented methods. The reason is, in a nutshell, that a necessary precondition for self-organization is nonlinear feedback, and that nonlinear feedback translates to deduction systems as nonmonotonic feedback, which is impossible. I explain this in more detail.

Nonlinear feedback is a precondition for self-organization in systems described by differential equations. "Nonlinear" is there clearly defined. However, for the present purpose a more general notion of "nonlinear" is required, since agents are usually not described by differential equations (exception: Steels 1993b). Therefore, I understand "nonlinear" in the more abstract sense that the trajectory of a dynamic system cannot be explained as a superposition of trajectories of some simpler partial systems (where a "partial system" can be, e.g., a subset of variables, a spatially defined subsystem, or any other partial contribution to the complete system description). Rather, at every moment in the system's history, the updating results of one partial system irreducably affect the premises for determining the behavior of other partial systems in the next time increment. Still more dramatically, it makes no sense to update even a partial aspect of a partial system without taking into account the most recent update of other partial systems.

This abstract version of nonlinear feedback translates to logical deductive systems as follows. A deductive system is, in the simplest case, a set of formulas, with partial systems being subsets thereof. This system is updated by the application of inference rules to some formulas, which yields new formulas (formulas might also be deleted, which is irrelevant for the present concern). Now, when a particular subset of formulas is considered, this partial system *can* be updated without considering the outcomes of updates of other partial systems. It makes sense to apply inference rules within arbitrary partial systems, i.e., it makes sense to update a partial aspect of a partial system without taking into account the most recent update of other partial systems. Note that this holds even for nonmonotonic logics. In default logics (Reiter 1980), for instance, any monotonic inference rule can be applied to any partial system at any time. Thus, partial systems are not inherently coupled to each other in all their aspects, as required by nonlinear feedback.

The analogue of nonlinearity and irreducible coupling of partial systems would be a "nonmonotonic feedback". This would occur when the outcomes of updating some partial system would potentially afflict every inference in other partial systems. As a special case, the update of a partial system would influence further updates of the same partial system in a nonmonotonic fashion. This is inconceivable, since it would essentially mean that theorems derived from a set

of axioms might invalidate some of the axioms. This is a rather abstract argument. I shall now examine it from some more concrete angles.

Nonmonotonic feedback obviously occurs in human reasoning. Humans can start reasoning about some task or problem and be lead to a detection of inconsistencies in their premises on the way, without import of new information (by contrast, in nonmonotonic reasoning and truth maintenance systems, inconsistencies arise not from within a closed reasoning process but are due to additional information). These inconsistencies are then remedied, e.g., by dropping assumptions or by encapsulating them as exceptions. Note that dramatic inconsistencies, which require immediate remedy, are rare. Mental states typically contain rather inconspicuous inconsistencies. Carrying along inconsistent notions is harmless as long as mutually inconsistent pieces of information do not effectively clash in inferences.

Admitting a certain amount of inconsistency in an agent's mental state is an advantage rather than a drawback. For instance, a reservoir of notions that are moderately inconsistent with each other can be helpful for swift adaptation to change, very much like genetic variance in a population is (on a larger time scale). When some external circumstance that deviates from the dominant foreground of current assumptions becomes suddenly relevant, there might already exist some piece of information in the background of the mental state that can serve as a starting point to handle the new circumstance. The priorly dominant portion of assumptions shrinks into the background, and a new foreground image "grows organically", so to speak, out of the suddenly relevant germs. In a classical reasoning system, where the reservoir of current assumptions has to be maintained in a state of logical consistency, externally triggered deviations from this state require an expensive reset of some kind (e.g., computing a new state of belief in a justification-based truth maintenance system; cf. Doyle 1979).

One can sketch the following plausible, overall image of mental dynamics. Short-term memory can be assumed to be graded along an axis that ranges from a highly active, conspicuous foreground to an inconspicuous, less active background (discussion of empirical findings and their theoretical interpretation in Shiffrin 1992). Inconsistencies can occur primarily between fore- and background, but rather not within the foreground, where they would induce active coping. Parts of the background can grow into the foreground and vice versa. Such "zooming" can be induced by the internal dynamics alone, or by input of new information, which leads to an increase in relevance of some piece of background information. This is an aspect of nonlinearity insofar as the (logically nonmonotonic) dynamic interdependency of all parts of short-time memory is emphasized. Besides being advantageous for adaptation to changing circumstances, this conception of short-time memory also has benefits for memory search. Since inconsistencies are admitted, the overall content of short-term memory can cover mutually exclusive interpretations of the current external situation, which might each become relevant as time progresses. When the temporal development of internal reasoning and/or external circumstances homes in on one of the alternatives, the activation of relevant knowledge from long-term memory is facilitated, since an appropriate clue for the access is already present in short-term memory.

Self-organization is intimately connected with situatedness. The situated action paradigm emphasizes that an agent's every information processing (sub-)mechanism is intrinsically tied up with the agent/environment interactions. It makes no sense trying to explain an agent without taking into account every aspect of this feedback loop. This is a version of the case made for nonlinear feedback above.

By contrast, logic-oriented approaches to agent modeling rely on some kind of sense-reason-act cycle, where during the "reason" portion of the cycle the agent (or the part of the agent that

does the reasoning) is essentially *disconnected* from the environment. This is a fundamental implication of a logical approach to reasoning. A logical inference algorithm, monotonic or nonmonotonic, starts in each inferencing episode from a given set of premises, executes a number of inference steps, and comes up with some result if all goes well. It is not possible to change the premises *while* the inferencing is under way. An inference algorithm, so to speak, retracts form the rest of the world for the time it is occupied with a given task. A logic-oriented approach to agent modeling thus finds itself in an intrinsic conflict with self-organization.

The logic-induced detachment from the environment becomes particularly apparent when the reasoning task is complex and requires a considerable amount of computation time, as in planning tasks. This explains, partially, why the situated action critique often focusses on planning (as in the classic, Suchman 1987).

A direct consequence of logic-oriented reasoning being decoupled from the environment concerns real-time demands. In order to enable the agent to respond to environmental change sufficiently fast, a typical compromise in logic-oriented design is to sacrifice logical expressiveness and implement but a small fragment of some logics. The reason-part in the sense-reason-act cycle can then be confined within a narrow time slice, and the entire cycle be iterated at a high frequency. The net effect is that the overall performance is quasi-continuous. This approach seems to characterize the current state of the art (e.g., Shoham 1993, Nilsson 1994). An interesting variant is realized in the robot Flakey (Congdon et al. 1994, p. 11):

"Flakey's software system is designed so that all processes operate in parallel with a basic cycle time of 100 milliseconds. [...] *Even though some processes, such as map registration, could take many seconds ... to complete, all processes were written so that they save partial results and complete within the cycle time."*

Although it is not explained in detail in the article, it appears that Flakey's time-consuming map registration (i.e., the buildup of a "mental image" of its environment) is an example of a relatively long-term detachment of a part of the robot's information processing from the environment. This can work satisfyingly only when there are no changes in the environment that are faster than the map registration procedure. In the case of Flakey, only the outer walls of the AAAI competition arena are reflected in the map. They do not change at all.

In "theoretical" classic AI, highly expressive formalisms are developed that are relevant for situated agent modeling (e.g. rich spatial representation formalisms, situation semantics, situation calculus, modal logics for intentions and desires or temporal reasoning). In "practical" classical AI approaches to mobile robot design, only fractions of such formalisms are exploited. This may turn out to be more than merely a reflection of imperfections in the current state of the art. It is a well-known fact that logical expressiveness has to be paid for with large allowances for computation time, which become intractable very easily. Thus, a higher degree of expressiveness necessarily leads to longer periods of an agent's being detached from the environment, with the imminent danger of virtually infinite detachment periods. Combining expressiveness with situativity might in the end be principally infeasible on the grounds of logic-oriented approaches.

The issue is connected with the frame problem (McCarthy & Hayes 1969). In its original version, this problem concerns the update of facts that hold in a temporal succession of situations, where both the situations and the transitions between them are described by logical formulae. The frame problem can be stated in more general forms. Fodor (1987) claims it to arise whenever rational inferences occur. The problem is virulent exactly because reasoning is

modeled as a process that is decoupled from the environment. It vanishes when a self-organizing agent/environment interaction loop is taken to be effective, which does not rely on off-line manipulations of internal representations: *"...this architecture* [i.e., self-organizing biological brains] *coordinates perception and action without intermediate decoding and encoding into descriptions of the world ..., thus avoiding combinatoric search ... and the frame axiom problem ..."* (Clancey 1993, p.94). When internal representations are continually coupled to sensor input, maximising consistency as a relative tendency instead of maintaining absolute consistency of situation representations, the question of whether some assumption holds in the (changing) environment becomes a matter of provisonality and degree. Thus, the frame problem cannot even be properly stated.

A self-organizing kind of information processing is fundamentally in accordance with real-time demands, since there is no sense-reason-act working cycle and, therefore, no intrinsic need to detach the system from the environment for performing reasoning.

Now I turn to the issue of compositionality in behavior-oriented agent design, choosing Steels' (1993a, 1994) robots as a basis for discussion.

The basic building blocks in Steel's architecture are simple C procedures. They are executed in (simulated) parallel in a "process network". With a few exceptions, there are no explicit influences between them. They realize low-level processes, e.g., a slow tendency to maintain a standard forward speed, or a fast retract movement triggered by bumper sensors. Only in some cases, behaviors are realized by a single such process (e.g., the retract reflex). Typically, a behavior "emerges" from the parallel execution of several processes in the agent-environment interaction loop. This situation can be restated in terms of compositionality, or rather, in terms of the absence of it. Save for a few exceptions, there are no mechanisms to couple processes with each other by dedicated mechanisms. The coupling is caused by contingent agent/environment interactions; it is an epiphenomenon that is not in itself causally relevant for the agent's performance. Thus, considering the conditions (2b) and (2c) from above, the emergence of behaviors from processes does not qualify as a composition of processes.

How, then, would a true composition of processes look like? There would have to exist some mechanism for linking together more or less arbitrary selections of processes. This mechanism must be of a physical, causally effective nature. For instance, there might be a C procedure `compose` that takes arbitrary processes as arguments and explicitly labels them as an "active selection". The intended interpretation of active selections is to view them as behaviors. Only processes that occur in an active selection are executed. When `compose` is slightly modified to accept as arguments active selections besides individual processes, the composition operation can be iterated to an arbitrary compositional depth. Together with an inverse procedure `decompose` (with obvious semantics), condition (2a) is satisfied. The building of composites is also causally relevant for the agent's functioning, i.e., (2b) is satisfied. Labeling and execution are physical effects; thus condition (2c) is satisfied. Finally, (2d) is satisfied, since `compose` accepts every combinatorially possible selection of arguments.

This (or a similar) modification of Steel's process network yields a clearly compositional type of information processing. The question arises whether this modified architecture is still self-organizing. The answer depends on how the `compose` procedure gets triggered. When there is some central control for this procedure, then one has essentially arrived at a classical architecture, and self-organization is likely to be lost. Self-organization is only possible when there is an irreducible feedback between partial systems. This implies that different `compose` (and `decompose`) events must influence each other. By one `compose` triggering, some

other currently active selection might be induced to become decomposed, composed further to a higher compositional depth, enlarged by the aggregation of further processes, or be reduced in size; or other rote executions of `compose` might be triggered.

Actively selecting some processes, while turning others off, is a precondition for a particular, rapid kind of adaptation, which one might call "rapid behavior setup". The key observation is that in a given situation an intelligent agent typically uses only a fraction of the low-level processes that are potentially at its command, and that the selection of relevant processes varies considerably across situations. When I pour coffee, processes like compensating for the can's weight, balancing, and stereo vision control of hand position are active. When in the next moment I walk to the cupboard to fetch the sugar pot, most of these mechanisms become irrelevant, and a quite different selection is active.

The "pure" situated action paradigm explains such sequences of behaviors in terms of the agent/environment interaction loop, viewing behaviors as exclusively emergent phenomena. This suffices for a reflex-driven, "insect-like" mode of operation, as is witnessed by current realizations of behavior-based robots. I claim that when higher forms of intelligence come into the play, an additional mechanism for an active, internally co-determined, causally effective setup of behaviors is needed. An intelligent agent pursues goals (which can be as simple as pouring coffee), and these goals arise at least partially within the agent in a top-down fashion. They cannot be explained completely bottom-up by the agent/environment interaction loop. For the time a goal is pursued, a particular pattern of processes must to some extent be kept physically stable against perturbations that incessantly arise in the agent/environment interaction loop. This does not imply that the relevant pattern of processes is fully determined in a solely top-down, arbitrary fashion. The basic agent/environment interaction loop is not replaced, but superimposed by an additional selection mechanism, and the "affordances" of the situation still co-determine the agent's behavior.

So far, I have argued from the side of motor actions. A similar case for a causally relevant composition mechanism can be made for perception. Again, as long as an "insect-like" mode of operation is concerned, perceptive processes might be fully explained in terms of the agent/environment interaction loop. This does not suffice when the level of intelligence rises. At some level, intelligence implies that an agent can perceive objects hitherto unknown, or perceive known objects in new circumstances where the effective sensor signal deviates from earlier occurances. Such a faculty requires that perceptive features can to some extent be grouped arbitrarily, and the established selection be stabilized for the time the object is monitored. For instance, a porcelaine fish with wings that one sees in a fancy shop becomes established in perception almost instantaneously as an object. At such an instance, features are composed that one has never composed before. This composition of features is partially accounted for by the stimulus in a bottom-up way; at the same time, it is also influenced top-down, e.g. by animal schemes. It is certainly hard to explain how exactly one manages to perceive a fish with wings. It seems clear, however, that a relatively stable composite of features is set up ad hoc, which helps to keep hold of the thing in the face of all kinds of perturbations.

The coffee-pouring and the winged fish monitoring examples have much in common. In both cases, active selections are generated on the spot, which cannot be explained solely as emerging from the agent/environment interaction loop. This is a consequence of a relatively high intelligence level that brings internal top-down influences into play. At the same time, the lessons taught by situated action imply that one should not try to explain either example *without* taking self-organization in the agent/environment interaction loop into account.

I come to a conclusion for this subsection. The discussion has shown that intelligent, situated information processing must be explained as a combination of self-organization with compositionality, both on central, intellective and on peripheric, sensomotoric levels. Self-organization and compositionality are here understood as fast, if not instantaneous, mechanisms (long-term adaptation and learning would have to be included into a more comprehensive account). Classical AI is concerned mainly with central levels, where it focusses on compositionality but has fundamental difficulties with self-organization due to its orientation towards logics. Behavior-oriented approaches favor peripheric levels, where they handle self-organization but omit compositionality. Compositionality on lower levels is connected with top-down influences of an intelligent character, self-organization on higher levels is connected to the agent/environment interaction loop. Integrating self-organization and compositionality across all levels is thus an important research task. It amounts to integrate classical AI with situated action, and bottom-up with top-down influences, in a principled way. In the next subsection, I will outline how the dynamic symbol perspective and the DSS formalism can contribute to this task.

Summary of section 2.9

- The notions of self-organization and compositionality are specified in a broad (not quite standard) sense, each by a collection of characteristics of information processing mechanisms.
- No true integration of both aspects is currently available.
- Self-organization implies that partial systems influence each other in all their aspects (nonlinear feedback). This translates to logical deductive systems as "nonmonotonic feedback", which is not feasible in logic-oriented approaches, since it would essentially mean that theorems derived from axioms can invalidate the axioms. Logic-oriented AI is thus in principle unable to account for self-organization.
- Nonmonotonic feedback does, however, occur in human reasoning. It is beneficial for fast adaptation to a changing environment, memory access, real-time demands, and it makes the frame problem obsolete.
- Compositionality is relevant on peripheral levels for the ad hoc setup of behaviors and perceptive schemes. This does not invalidate the self-organizational aspects of the agent/environment interaction loop, but it requires the addition of a causally effective mechanism for the fast and essentially arbitrary coupling of simple motor processes or perceptual features.
- An overall picture of intelligent, situated information processing emerges, where self-organization and compositionality co-occur on all levels on the periphery-centre axis. This makes an integration of both aspects a centrally important issue for further research.

2.10 Design vs. autonomy and development

Agent design seems to follow a straightforward and natural pattern. The designer has in mind a particular performance profile of the future agent. This profile can be stated in terms of high-level capabilities, as is typical for classical AI; it can also consist in the specification of low-

level functionalities that are apt to guarantee the agent's basic "survival", as in typical behavior-oriented projects. Starting from these premises, the designer works out a system of mechanisms that can be expected to yield the desired performance. The perspective is, above all, an explicitly functional one. I shall call this the "principle of explicit design": *Artificial agents are designed in terms of functions, and mechanisms that serve them, both of which are explicitly stated by the designer.*

The principle of explicit functional design obviously agrees with practice in classical AI approaches to mobile robot design. At a first glance, behavior-oriented design is not different. Behavior-based robots are equipped with behaviors whose function is prescribed by the designer. For instance, Steels (1993a) describes the implementation of a "retract reflex" that serves a goal of coping with collisions. The name `retract_reflex` even occurs in the program code.

But, this design strategy is at odds with the autonomy requirement, which I have already described in the discussion of functional aspects of the periphery-centre axis (cf. 2.3). In the theoretical framework of situated action, agents are defined to be autonomous and subject to their own proper laws. This conflicts with a design strategy that starts from predetermined tasks. The inconsistency becomes apparent in the following quotation (Steels 1993a, section 5):

"The first step in the design of the agent is usually a decomposition into the major tasks and subtasks. This decomposition is not the basis of the design of the internals of the agent [...] *but a design-oriented decomposition which helps to identify the behavior systems that need to be present."*

This statement is not quite consistent since the decomposition *is* in practice the basis of the design of the agent's internals. Mechanisms for behaviors that serve the identified tasks are effectively and explicitly implemented (cf. the `retract_reflex` example).

The inconsistency roots in a fundamental incompatibility between *autonomy* and *design*. Autonomy means that an agent has developed, functions at the present, and will further develop in the future, all according to its own rules, which themselves are subject to continual development and cannot be fully fixed by an observer. By contrast, design implies that an agent's functioning is externally prescribed, and that the agent does not develop but comes to the world as a fixed, ready-to-work device. Biological agents can be autonomous only because never in their phylogenesis and ontogenesis they have been subject to an explicit design procedure. They have, in a way, constructed themselves, and radical constructivism teaches that they continue to construct themselves even as adult individuals. Artificial agents are created from nothing. In order to guarantee survival from their existence's beginning, their immediate needs have to be understood by their creators, and they are provided with an explicit outfit of mechanisms to satisfy these needs.

A way out of the theoretical dilemma is to require that the design be *open*. Artificial agents have to be designed in a way which allows them to develop their own proper responses to the environment during their individual "life". A functional design, in this view, provides but some starting parameters for an unpredictable, designer-independent further development. Explicit design can be considered an unavoidable *ersatz* for the evolutionary and ontogenetic outfit of which biological agents afford. Autonomy, then, grows from what design has left open for the agent to develop on its own. I shall call this the "principle of open design": *Artificial agents must be capable to modify their design.* This is a necessary complement to the explicit design principle. Unfortunately, it is quite difficult to realize open design in practice.

The only computational methods presently known that can in principle lead to autonomous development are modeled on natural evolution. There are two main schools of research, genetic algorithms (cf. Goldberg 1989), and evolutionary strategies (cf. Schwefel 1977), of which only the former is currently relevant for AI, in particular through its standard combination with classifier systems, which are symbolic in their nature. By contrast, evolutionary strategies are applied to numerical optimization. Both approaches rely on the selection of stochastic variants according to some fitness function. The stochastic element renders the design open at least insofar as the system's development becomes unpredictable. Unpredictability is, however, not sufficient for openness in the sense of autonomy. Autonomy requires that the fitness function itself must be open, i.e., it must not be specified by design: *"In the context of emergent functionality, we expect that the fitness function should be subject to evolution and should be local to the organism that evolves"* (Steels 1994, section 5.1).

The autonomy requirement, thus, leads directly to the task of providing the agent with a faculty of estimating the success of its (stochastically varied) actions. This is a crucial difference to natural evolution, where success is not evaluated for an individual's individual actions, but on the population level in terms of long-term survival. This global, implicit "mechanism", which essentially equates non-success with agent elimination, cannot be transferred to an individual agent's development, since the agent has to survive its adaptation (discussion in Steels 1994). Thus, within the agent there must exist a mechanism for estimating the quality of actions in an explicit (or at least, causally effective) fashion, locally for individual actions.

In the simplest case, such an evaluation amounts to the computation of a single "success" variable. It has been argued that biological agents afford of such a universal variable, namely, pleasure/displeasure (Cabanac 1992). Although I tend to subscribe to this view, the question remains of how pleasure (or a more technically termed correlate) can be effectively computed without getting trapped again in the pitfalls of external predetermination of the agent by the designer.

The discussion so far may have led to the impression that open design is very hard to come by, if it is possible at all. However, a reconsideration of some ideas developed in earlier subsections suggests a novel strategy to open design that appears promising. The basic idea is to exploit functional ambiguity for autonomous adaptation. I describe this in some detail.

When an agent is observed and explained, the descriptive categories that are used root in some implicit or explicit theory on the side of the observer. The design itself is guided by some such theory (or theories), which warrants that *this* theory is obviously suited to observe or explain the agent. But, this original theory is not the only one in which the agent can be framed. There is no uniquely proper way a given agent should be perceived.

As a special case of this phenomenon, the functions which a substructure in the agent can serve cannot be determined a priori in a comprehensive way. In subsection 2.9, I have emphasized the adaptive potential of this intrinsic functional ambiguity. I believe that here lies an opportunity for an interesting variant of the currently exploited evolutionary mechanisms for agent development.

Evolution needs some source of variance. In genetic algorithms and evolutionary strategies, variance is introduced by random modifications which are then "tested" for their value (e.g., for their potential to afford pleasure), becoming firmly established when they are good or weeded out when they are not. I shall call this the *modify-and-test* strategy for evolution.

The variant I have in mind does not rely on modifications as a source of variance. Rather, I interpret functional ambiguity as a variance *that is already there*; new functions only have to be *discovered* when they happen to reveal their value, and after their discovery they must be effectively established, in order to render them repeatable. I shall call this a *discover-and-modify* strategy for evolution.

To understand this strategy, note that an agent's functionalities reveal their existence through their being triggered by suitable external circumstances. Functional ambiguity implies that the agent may always (in a new situation) find that available mechanisms interact in an unforeseen way in a functional, "valuable" fashion. Earlier in this section, I have introduced functional ambiguity as something that results from the theory-dependence of the observations of an external observer. Now, to make this idea effective for the agent itself, the latter must act as its own observer. It must be capable to self-monitor and to "discover" functions as they reveal themselves in suitable circumstances. This need not imply consciousness; "discovery" here means that a physical trace can be left in the agent when a function reveals its existence by its circumstance-triggered execution. This physical trace can then causally effect the agent's subsequent behavior.

For an example, consider an architecture like the one proposed by Steels (1994), where a comparatively large number of processes is carried out in parallel (compare 2.9). Functions (called "functionalities" by Steels) are physically realized by the combined execution of subsets of these processes (such subsets are called "behavior systems" by Steels). The set of functions that are implicitly "already there" is practically as good as infinite; every subset of the set of processes potentially realizes a function. Now, circumstances may occur where a subset of the currently active processes achieves some success with respect to the agent's measure of success (e.g., pleasure). Assume that the agent can cope with the credit assignment problem, i.e. it detects which of the currently active processes contribute to the success. Then, this subset of processes can be recorded, leaving a physical trace, which can subsequently be used to re-activate this subset of processes in a "deliberate" fashion. The event of forming this trace is what I mean by "discovery". Technically, the trace could, e.g., consist in an augmentation of the `compose` procedure (cf. 2.9).

Some points from this example should be pointed out. First, this adaptation mechanism does not rely on modifications as a source of variance. A modify-and-test strategy would *first* establish a (random) new mechanism (e.g., by augmenting `compose`), and then try out the success of the new architecture. In the above mechanism, by contrast, a particular, unpremeditated success of the old architecture is first discovered, and *then* the mechanism is established in a subsequently causally effective fashion. Second, discovery-and-modify is likely to be cheaper and faster than modify-and-test, since no dysfunctional changes are first physically established only to be weeded out again. Third, the mechanism is more plausible than modify-and-test. It does not import into an individual a strategy that originates in population dynamics, which is a questionable transfer. Also, one can easily find examples for biological agents adapting by discovery and subsequent fixation of functional behaviors. In fact, classical behaviorism describes discovery-and-test mechanisms. Conversely, it is hard to find examples where a random behavior is first fixed in an individual and then either kept or weeded out again.

Discovery, as it is here conceived, is intimately related to compositionality (cf. 2.9). An event of discovery can be interpreted as the causally effective memorizing of a causally effective composite. In the example of Steel's architecture, this is a composite of processes, but any other composite of informational entities in an agent potentially supports a function.

Summary of section 2.10

- Artificial agents must, to some extent, be designed in terms of explicit functions. They substitute the functional outfit endowed to biological agents by phylogenesis and ontogenesis, mechanisms that are practically infeasible to repeat in artificial agents.
- Autonomy requires that an agent's design be open, i.e., the agent must be enabled to further develop its own design.
- Known mechanisms for autonomous development are derived from the model of natural evolution. The original fitness function, which works on the level of populations, must be individualized and internalized. This leads to considering a causally effective pleasure/displeasure variable in agents.
- A variant of the standard modify-and-test evolutionary strategy for autonomous development is provided by discovery-and-modify mechanisms. They exploit the fact that an agent implicitly possesses a large variety of functions, which can be discovered in suitable circumstances, and recorded in a subsequently effective fashion.

Conclusion

The dynamic symbol structure framework presented in this section offers an integrative view on classical and situated action approaches to agent modeling. This is achieved essentially by taking an empiricist stance and defining symbols as observables. The resulting perspective is compatible with the classical physical symbols systems hypothesis and with situated action. The integrative potential of the approach is emphasized by that it allows to identify a central classical notion, namely, (internal) reference, with a notion that is central in the behavior-oriented paradigm, namely, emergence.

Modeling agents in terms of observational categories rather than in terms of assumed "objective" entities has notable consequences:

- Dynamic symbols come with an abstraction dimension, which has an observation-dependent resolution component, and an observed-object-dependent differentiation component. This leads to an explanation of abstraction hierarchies that is different from the dassical one, which builds on the inclusion of extensions. However, the two types of abstraction hierarchies are formally quite similar, and disjunctive concepts can be explained even without resorting to extensions.
- The notion of functional ambiguity, which is a consequence of the theory-dependent nature of observational categories, leads to a plausible variant of evolutionary mechanisms, namely, discover-and-modify strategies. This is a contribution to the task of open design.

The dynamic symbol framework brings together a typical classical and a typical situated action aspect of information processing, namely, compositionality and self-organization. It emphasizes that both aspects must be intimately linked on all levels in order to enable an agent to react swiftly and effectively to changing circumstances. Classical AI, where it is logic-oriented, cannot support self-organization, since the latter implies a direct nonmonotonic feedback of the effect of an inference step to its premises. Behavior-oriented approaches shun fast and effective

compositionality, since it has an explicit top-down control flavor, and since these approaches account for composition (of behaviors) either in a fast but epiphenomenal, or in a causally effective but long-term adaptive fashion. In the dynamic symbol perspective, by contrast, it becomes apparent that

- self-organization is necessary even on central levels, in order to avoid the pitfalls of decoupling the agent from its environment temporarily, and the frame problem, and that
- a fast and causally effective kind of compositionality is necessary even on peripheric levels to enable a fast setup of behaviors and perceptual schemes.

I feel that the integration of self-organization with fast compositionality, which is concretely realized in the DSS formalism, is the single most important contribution of this thesis to research on agent design.

3 Dynamic symbol systems

This section presents the DSS formalism in formal rigor. There are four subsections. In 3.1, a subclass of the regular languages is treated, *coherent* languages. They are motivated by basic assumptions concerning the nature of local observations of dynamic systems. In 3.2, the DSS analogue of long-term memory, *dynamic symbol spaces*, is introduced. A dynamic symbol space is in many aspects similar to a classic terminological knowledge base; seen from a different angle, it resembles a thermodynamic state space. In 3.3, *self-organizing scenes* and *self-organizing streams* are presented. They are closed (scenes) or open (streams) dynamic systems, which develop in time by virtue of local operations, *microchanges*, that can be applied randomly and in parallel. The resulting global dynamics exhibits rapid equilibration and self-organization effects, which are "measured" with respect to an underlying dynamic symbol space. Finally, in 3.4, I show how several self-organizing streams can be coupled together in order to achieve complex, multi-level information processing architectures, called *associeties*.

3.1 Coherent languages and coherencies

Coherent languages are a subclass of regular languages. They are characterized by closure under subwords and a certain cyclicity condition. The basic motivation to consider such languages lies in the idea of an observation of a dynamic system. When a parallel dynamic system, in which dynamic symbols interact, is observed "locally" (i.e., only a single dynamic symbol is observed at a time), what one will find are temporal, spatial, or spatiotemporal sequences of dynamic symbols. Seqences are temporal when the locus of observation is fixed and one records in time; e.g., when a single output channel of a system is monitored. They are spatial when the focus of observation cuts a trace line through the system at infinite velocity; e.g. when a brain region is scanned that is coupled to visual input via a topographic mapping from the retina. Spatiotemporal sequences are obtained by an observation focus that moves with finite velocity. Jaeger (1992) gives some temporal examples of such observations, concerning sequences of words uttered by a subject. When an observation is considered as amounting to a sequence of dynamic symbols, it is natural to require that subsequences should also qualify as observations. This, then, leads to languages that are closed under subwords. The cyclicity condition will be motivated further below.

I assume that the reader is roughly acquainted with regular languages (standard reference: Hopcroft & Ullman 1979). However, no results from the theory of regular languages are used in the sequel, since the little theory that I develop comes with its own techniques, which are tailored to the particular character of coherent languages. Thus, while an acquaintance with regular languages will be helpful for a background, it is not strictly required for an understanding of the following material (with the exception of the proof of proposition 8, which is, however, not crucial for the whole).

The material presented in this subsection provides mathematical prerequisites for DSS rather than being a part of DSS proper. Therefore, I postpone the usage of DSS-typical terms until subsection 3.2. In particular, I say "symbols" and "words" (rather than "dynamic symbols" and

"associations"), since the present subsection obviously belongs to the area of formal languages, where another terminology than the customary one would be confusing.

Definition 1 recalls the basic notions of formal language theory.

Definition 1:
(i) An *alphabet* is a finite set $\Sigma = \{a_1, ..., a_n\}$ of **symbols**.
(ii) $\Sigma^* := \{s_1...s_k \mid k \geq 0, s_i \in \Sigma \text{ for } i = 1, ..., k\}$ is the set of **words over** Σ.
(iii) If $\boldsymbol{s} = s_1...s_k$, $\boldsymbol{t} = t_1...t_l$ are words over Σ, then $\boldsymbol{st} := s_1...s_k t_1...t_l$ is the **concatenation** of $\boldsymbol{s}$ and $\boldsymbol{t}$.
(iv) If $\boldsymbol{s} = s_1...s_k \in \Sigma^*$, then $|\boldsymbol{s}| = k$ is the **length** of $\boldsymbol{s}$.
(v) If $|\boldsymbol{s}| = 0$, then $\boldsymbol{s}$ is the **empty word**. It is denoted by ε.
(vi) A subset $L \subseteq \Sigma^*$ is a **language over** Σ.

By convention, *r*, *s*, *t* are variables for symbols, *a*, *b*, *c* are constants, and bold print indicates words. Words of length 1 are identified with symbols.

I approach coherent languages from a slightly more general angle than from the perspective of regular languages, considering at the outset all languages that are closed under subwords:

Definition 2: A language L over Σ is **segmentable** :iff L is closed under subwords, i.e., if the following two conditions hold:

(i) $\forall r_1 r_2...r_n \in L \; \forall \; 1 \leq i \leq j \leq n: r_i r_{i+1}...r_j \in L$,
(ii) $\varepsilon \in L$.

Condition (ii) is an arbitrary auxiliary condition; one might just as well require $\varepsilon \notin L$. Making a commitment, one way or the other, is an act of formal hygienics.

Next comes some nomenclature that will be frequently used in the sequel:

Definition 3: Let L be segmentable.
(i) For $\boldsymbol{rs} \in L$, $\boldsymbol{s}$ is a **continuation** of $\boldsymbol{r}$ in L, and $\boldsymbol{r}$ is a **context** for $\boldsymbol{s}$ in L. For $\boldsymbol{r} \in L$, the set $continue_L(\boldsymbol{r}) := \{\boldsymbol{s} \in L \mid \boldsymbol{rs} \in L\}$ is the set of all continuations of $\boldsymbol{r}$ in L. When the reference to L is clear, the subscript L can be dropped.
(ii) $r_1 r_2 r_3...$ is an **infinite continuation** of $\boldsymbol{r}$ in L :iff $\boldsymbol{r} r_1...r_k \in L$ for every $k \geq 0$. The set of all infinite continuations of $\boldsymbol{r}$ in L is denoted by $continue_L^{\infty}(\boldsymbol{r})$. In the same vein,

$$L^{\infty} := \bigcup_{\boldsymbol{r} \in L} continue_L^{\infty}(\boldsymbol{r})$$

denotes the set of all infinite continuations in L.

The following proposition is a direct consequence of definitions 2 and 3:

Proposition 4: Let L be segmentable. Then the following statements hold:
(i) $L = continue(\varepsilon)$, $L^{\infty} = continue^{\infty}(\varepsilon)$.
(ii) For $r_1...r_k \in L$, it holds that $continue(r_k) \supseteq continue(r_{k-1} r_k) \supseteq ... \supseteq continue(r_1...r_k)$. □

The moral of (ii) is that contexts act as "filters" with respect to concatenation of words. The further a context is lengthened to the left, the more the set of its possible continuations to the right is restricted. When words are interpreted as temporal or spatiotemporal observations, then the intended interpretation of contexts is to consider them as information that is already recorded, and continuations as potential future observations. Then, the "filtering" property can be restated by saying that the more one knows about the past, the more precisely the future can be predicted. Contextual filtering is a recurrent theme in DSS.

Next I introduce a convenient description for segmentable languages. A *generator* of such a language is a directed graph, where the edges (called *transitions*) are labeled by symbols from Σ. Words can be read out of this graph by following a finite path through the graph and collecting symbols on the way.

Definition 5: Let $L \subseteq \Sigma^*$. Let G = (S, *trans*), where S is a set and $trans \subseteq S \times \Sigma \times S$. Then G is a **generator** of L :iff

$\boldsymbol{r} \in L$ iff $\boldsymbol{r} = r_1 ... r_n$ and $\exists\, x_0, x_1, ..., x_n \in S\ \forall\, i = 1, ..., n: (x_{i-1}, r_i, x_i) \in trans$,
or
$\boldsymbol{r} = \varepsilon$.

The elements of S are called the **local states** of G. Variables for local states are x, y, z (not in italics). Specific local states are usually represented by natural numbers. The elements of *trans* are called **transitions**. It is convenient to write simply x*r*x' instead of $(x, r, x') \in trans$. When for i = 1, ..., n, it holds that $(x_{i-1}, r_i, x_i) \in trans$, then $x_0 r_1 x_1 r_2 x_2 \ldots r_n x_n$ is a **derivation** of $r_1 ... r_n$ in G. The language L generated by G is denoted by L_G.

Example 6: Fig. 3.1 shows two generators of a language L, where

$\Sigma = \{a, b, c\}$,
$L = \{r_1 ... r_n \in \Sigma^* \mid n \geq 1, r_i \neq r_{i+1} \text{ for } i = 1,...,n-1\} \cup \{\varepsilon\}$.

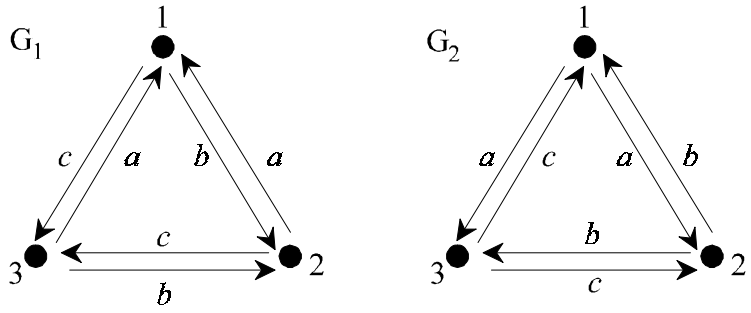


Fig. 3.1: Two non-isomorphic generators of L from example 6.

Proposition 7:
(i) If there exists a generator for L, then L is segmentable.
(ii) If L is segmentable, then there exist infinitely many non-isomorphic generators for L.

Sketch of proof: (i) is a direct consequence of the fact that subpaths of paths in a graph are paths, i.e., subderivations of derivations are derivations, i.e. subwords of derivable words are derivable words. For (ii), a **trivial generator** for a given segmentable L can be constructed by assigning to every word $\boldsymbol{r} = r_1...r_k$ of L an own linear-graph-like generator

$$G_{\boldsymbol{r}} = (\{x^{\boldsymbol{r}}_0, ..., x^{\boldsymbol{r}}_k\}, \{(x^{\boldsymbol{r}}_0, r_1, x^{\boldsymbol{r}}_1), ..., (x^{\boldsymbol{r}}_{k-1}, r_k, x^{\boldsymbol{r}}_k)\}),$$

whose unique derivation of maximal length derives $\boldsymbol{r}$. The disjoint union of these generators yields a generator for L. Non-isomorphic variants of this generator can be obtained by adding disjoint copies. Example 6 shows that this is not the only way to arrive at non-isomorphic generators for a segmentable language. □

The moral of proposition 7 is that being segmentable and possessing a generator is the same thing for a language. This can be specified further: for a language, being segmentable and regular is the same as having a finite generator.

Proposition 8: Let L be segmentable. Then L is regular iff there exists a finite generator of L.

Sketch of proof:
"⇒": Let $A = (S, trans, x_{start}, S_{accept})$ be a deterministic finite automaton accepting L, where S is the set of states, $trans \subseteq S \times \Sigma \times S$ is the set of transition rules, $x_{start} \in S$ is the initial state, and $S_{accept} \subseteq S$ is the set of accepting states. Remove from S all states from which no accepting state can be reached, or which cannot be reached from x_{start}, and remove from *trans* all transitions which are thereby affected. Let the result be S' and *trans*'. Define a generator $G = (S', trans')$. Now if $x_0 r_1 x_1 ... r_n x_n$ is a derivation in $(S', trans')$, then there exists a word $s_1...s_m r_1...r_n t_1...t_k$ in L which is accepted by A on a path which contains $x_0 r_1 x_1 ... r_n x_n$. Since L is segmentable, it holds that $r_1...r_n \in L$. Therefore, all derivations in $(S', trans')$ yield elements of L. Conversely, each word of L can be derived in $(S', trans')$, since there exists for this word a derivation in A leading from x_{start} into S_{accept}, and this path is conserved in $(S', trans')$. Therefore, $(S', trans')$ is a finite generator of L.
"⇐": Let $G = (S, trans)$ be a finite generator of E. Introduce a new local state s_{start} and connect it to every local state of G by an transition labeled by ε. Declare every $x \in S$ an accepting state. Then interpret the result of these modifications as a non-deterministic finite automaton. It accepts the language L, which is, therefore, regular. □

Regular segmentable languages could be further examined in the spirit of regular language theory. For instance, it is easy to show that if L and L' are regular and segmentable, then their union and intersection are regular and segmentable. Likewise, it is not particularly difficult to formulate what is called an "algebraic characterization" for regular segmentable languages. Normal form theorems for finite automata can be exploited to define normal forms for finite generators. The latter, however, does not lead far. Finite automata normal forms have to be awkwardly transformed to make generators, since in generators every state basically is both an initial state and an accepting state, which renders them quite different from finite automata. I will develop a more suitable normal form for *cyclic* generators further below, using methods which do not originate in the theory of finite automata.

Regular segmentable languages can also be characterized by grammars (by adding suitable conditions to the right (or left) linear grammars for regular languages). Although it is very common to describe formal languages by grammars, I will not do so. The reasons for preferring generators are, first, that the cyclicity condition that specifies *coherent* segmentable

languages (see below) is directly reflected in the graph-theoretical cyclicity of generators, whereas an analog condition for grammars would be less transparent; second, that the fundamental operation of *symmetry breaking* (see below) could not be easily defined for grammars; third, that the *dynamics* of self-organizing scenes and streams (see 3.3, 3.4) is expressed in terms of local operations on generators; fourth, that generators have a normal form (*phase generators*, see below), whose local states afford of an insightful interpretation in terms of the *information* that is provided about a sequence-generating dynamic system by the "observation" of a generated sequence. In sum, generators (in particular, phase generators) are transparent models of sequence-generating *dynamic* systems, whereas grammars are *structural* descriptions of languages. In this thesis, the emphasis is on dynamics and systems.

The next definition introduces *homomorphisms* between generators.

Definition 9: For G = (S, *trans*), G' = (S', *trans*'), η: S $\rightarrow$ S' is a **homomorphism** from G to G' :iff $x_1 r x_2 \in$ *trans* implies $\eta(x_1) r \eta(x_2) \in$ *trans*'. This is written as η: G $\rightarrow$ G'.

Proposition 10:

(i) If η: G $\rightarrow$ G', then $L_G \subseteq L_{G'}$.

(ii) If L, L' are segmentable, $L \subseteq L'$, $L' = L_{G'}$, then there exists a generator G of L and a homomorphism η: G $\rightarrow$ G'. When L, L' are additionally regular, and G' is finite, then a finite such G exists.

Sketch of proof: (i) is straightforward. For the general case in (ii), the trivial generator of L (as sketched in the proof of 7(ii)) can be taken for G. For L, L' being regular and G' = (S', *trans*') finite, and L being a language over Σ, take some arbitrary finite generator $G_0 = (S_0, trans_0)$ of L and define G as the product of G_0 and G':

$$G := (S_0 \times S', \{((x, y), r, (x', y')) \mid (x, r, x') \in trans_0 \text{ and } (y, r, y') \in trans'\}).$$

The desired homomorphism is the projection of this product generator on its second component, i.e. on G'. □

10(ii) signals some imperfection. What one would like to have, rather, is a full converse of (i). This will be obtained later with another kind of morphism (simulations), and a normal form of generators (phase generators).

After these basic and general definitions, I now turn to a special kind of regular segmentable languages and their generators, namely, to *coherent* languages and *coherencies*. They are the languages and generators on which DSS rests. Their characteristic property is cyclicity. The motivation to consider such languages is again a basic property of local observations of dynamic systems. When such a system does not qualitatively evolve (through learning or structure-changing adaptation), a long-term observation should exhibit repetitions of patterns. In the limit, every short-term observation should be repeated infinitely often in an infinitely long observation of a "random walk" of the system.

Definition 11: A language L is **coherent** :iff it is regular and segmentable, and for all $\boldsymbol{r}, \boldsymbol{s} \in L$ there exists $\boldsymbol{t} \in L$ such that $\boldsymbol{rts} \in L$. Coherent languages are denoted by the symbol C (instead of L for languages in general). C^∞ corresponds to L^∞ (cf. def. 3(ii)), i.e., it is the set of infinite continuations in C.

The *universal continuations* introduced in the next definition will turn out to be helpful in the further examination of coherent languages. The background interpretation is to consider them as observations of infinite duration, of a system that is not clamped in some special state by external control conditions but rather performs a random walk through its entire state space.

Definition 12: An infinite continuation $\boldsymbol{u} \in C^{\infty}$ is a **universal** continuation in C :iff every $\boldsymbol{s} \in C$ appears as a subsequence of $\boldsymbol{u}$ infinitely often.

Proposition 13: For all $\boldsymbol{r} \in C$, $continue^{\infty}(\boldsymbol{r})$ contains a universal continuation.

Sketch of proof: Since C has a finite generator, it is enumerable. Let $C = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3, ...\}$. Choose an enumeration $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, ...$ of C (with repetitions) in which every $\boldsymbol{r}_i$ occurs infinitely often, and where $\boldsymbol{s}_1 = \boldsymbol{r}$. From definition 11 it follows that there exist $\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{t}_3, ... \in C$ such that $\boldsymbol{s}_1\boldsymbol{t}_1\boldsymbol{s}_2\boldsymbol{t}_2\boldsymbol{s}_3\boldsymbol{t}_3... \in continue^{\infty}(\boldsymbol{r})$. This is a universal continuation. □

Definition 14:

(i) A generator is **cyclic** :iff for all local states x, x' there exists a derivation $x_0 r_1 x_1 ... r_n x_n$, where $x = x_0$ and $x' = x_n$.

(ii) A generator is a **coherency** :iff it is cyclic and finite.

Proposition 15: A language C is coherent iff C is generated by a coherency.

Sketch of proof: "⇒": Let $\boldsymbol{u}$ be a universal expansion in C, and let G be a finite generator of C. Apply Koenig's lemma to conclude that there exists an (infinite) derivation of $\boldsymbol{u}$ in C. Since $\boldsymbol{u}$ is infinite, some local states are visited infinitely often in this derivation. The reduct of G on these local states is a coherency that generates C.
"⇐": Obvious. □

I will now cast a closer look at epimorphisms $\eta: G_1 \rightarrow G_2$. When a coherency G_2 is an epimorphic image of a coherency G_1, G_1 is called a *symmetry breaking* of G_2. Before I go into explanations of why this is interesting, I supply the definition:

Definition 16:

(i) Let $\eta: G_1 \rightarrow G_2$, where $G_1 = (S_1, trans_1)$ and $G_2 = (S_2, trans_2)$. Then η is is an **epimorphism** :iff it is surjective, and for every $yry' \in trans_2$ there exists some $xrx' \in trans_1$ such that $\eta(x) = y$ and $\eta(x') = y'$.

(ii) When for two coherencies G_1, G_2 there exists an epimorphism $\eta: G_1 \rightarrow G_2$, G_1 is a **symmetry breaking** of G_2.

When η is interpreted "backwards", as an operation that yields G_1 from G_2, then, intuitively, a symmetry breaking is obtained by "breaking" the local states of G_2 and re-distributing the original transitions to and from an original local state over its breaking products, to arrive at a generator G_1. Figure 3.2 shows a simple example.

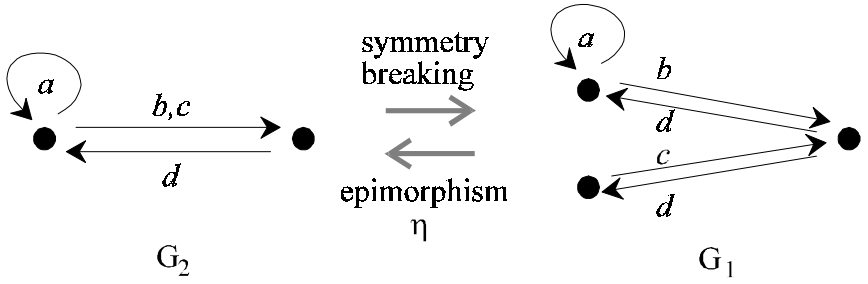


Fig. 3.2: A symmetry breaking.

The term "symmetry breaking" is borrowed from physics. It is used generally for processes where some single phenomenon "breaks" into distinguishable phenomena (as in the derivation of the electromagnetic, the strong and the weak interaction forces from an assumed unified force), or for processes where a system looses some degrees of freedom (as in cooling water to ice). Typically, symmetry breakings are induced by a decrease in temperature. All of this transfers to symmetry breakings as they are defined here. Single local states break into several distinct ones; words that can be derived starting from the original local state can become underivable after the symmetry breaking (e.g., *ac* is lost in the symmetry breaking from fig. 3.2), and last but not least, symmetry breakings can be interpreted in terms of a "computational temperature" (cf. subsection 3.2).

Next I introduce a normal form for coherencies, *phase generators*. Their local states are *phases*. Intuitively, a phase is the information carried by a "maximally informative" context. A context is maximally informative, in a sense, when its filtering effects cannot be augmented by lengthening the context to the left. In the interpretation of words as observations, this means that an observation is maximally informative when no additional knowledge about the future development of the observation can be gained by considering deeper extensions of it into the past. Such observations provide maximal possible knowledge about the state of the observed system. In this sense, such observations can be identified with the states of the system. It is customary to call the states of an oscillation generating system "phases". Using the term "phase" for the local states of generators of coherent languages is motivated by viewing (as a background interpretation) such generators as oscillation generating systems. The "oscillations", in this view, are the (infinite) continuations of a coherent language.

Definition 17: Let C be a coherent language.

(i) Let $\boldsymbol{p} \in C$. Then $\boldsymbol{p}$ is **phase-fixing** :iff $continue(\boldsymbol{p}) = continue(\boldsymbol{rp})$ for all $\boldsymbol{rp} \in C$.

(ii) Let $\sim_\varphi$ be the equivalence relation on $\{\boldsymbol{p} \in C \mid \boldsymbol{p}$ is phase-fixing$\}$ defined by $\boldsymbol{p} \sim_\varphi \boldsymbol{q}$:iff $continue(\boldsymbol{p}) = continue(\boldsymbol{q})$. The equivalence classes are called the **phases** of C. Phases are denoted by the symbol φ. The set of all phases of C is denoted by $\Phi(C)$. The equivalence class of $\boldsymbol{p}$ is denoted by $\varphi_{\boldsymbol{p}}$.

The next proposition collects some properties of phases.

Proposition 18: Let C be a coherent language.

(i) For every $\boldsymbol{r} \in C$, there exists some $s_1...s_n\boldsymbol{r} \in C$ ($n \geq 0$), such that $s_1...s_n\boldsymbol{r}$ is phase-fixing.

(ii) $\Phi(C)$ is finite.

(iii) If $\boldsymbol{p} \in C$ is phase-fixing, and $\boldsymbol{rp} \in C$, then $\boldsymbol{rp}$ is phase-fixing, and $\varphi_{\boldsymbol{p}} = \varphi_{\boldsymbol{rp}}$.

(iv) If $\boldsymbol{p} \in C$ is phase-fixing, and $\boldsymbol{pr} \in C$, then $\boldsymbol{pr}$ is phase-fixing.

Sketch of proof:

(i) Show first that the set $\{\mathit{continue}(\boldsymbol{t}) \mid \boldsymbol{t} \in C\}$ is finite, as follows. Let $G = (S, \mathit{trans})$ be a finite cyclic generator of C. Let $S_{\boldsymbol{t}} = \{x^{\boldsymbol{t}}_1, ..., x^{\boldsymbol{t}}_n\} \subseteq S$ be the set of those local states which are terminal points in derivations of $\boldsymbol{t}$. Let $x^{\boldsymbol{t}}_j{}^+$ be the set of all words that can be derived in G, where the derivation is started from $x^{\boldsymbol{t}}_j$. Then $\mathit{continue}(\boldsymbol{t}) = x^{\boldsymbol{t}}_1{}^+ \cup ... \cup x^{\boldsymbol{t}}_n{}^+$, i.e. $\mathit{continue}(\boldsymbol{t})$ is uniquely determined by a finite subset of S. Since S is finite, there can be only finitely many such subsets, and hence $\{\mathit{continue}(\boldsymbol{t}) \mid \boldsymbol{t} \in C\}$ is finite.
Proposition 4(ii) implies that for every sequence $\boldsymbol{r}$, $r_1\boldsymbol{r}$, $r_2r_1\boldsymbol{r}$, ... in C it holds that $\mathit{continue}(\boldsymbol{r}) \supseteq \mathit{continue}(r_1\boldsymbol{r}) \supseteq \mathit{continue}(r_2r_1\boldsymbol{r}) \supseteq ...$. Since $\{\mathit{continue}(\boldsymbol{t}) \mid \boldsymbol{t} \in C\}$ is finite, this implies that there exists some $s_n...s_1\boldsymbol{r} \in C$ such that $\mathit{continue}(s_n...s_1\boldsymbol{r}) = \mathit{continue}(\boldsymbol{s}s_n...s_1\boldsymbol{r})$ for all $\boldsymbol{s}s_n...s_1\boldsymbol{r} \in C$. I.e., $s_n...s_1\boldsymbol{r}$ is phase-fixing.

(ii) Follows from the finiteness of $\{\mathit{continue}(\boldsymbol{t}) \mid \boldsymbol{t} \in C\}$.

(iii) Trivial.

(iv) Assume that $\boldsymbol{pr}$ is not phase-fixing. Then there exists some $\boldsymbol{spr} \in C$ such that $\mathit{continue}(\boldsymbol{pr}) \supset \mathit{continue}(\boldsymbol{spr})$. It follows that $\mathit{continue}(\boldsymbol{p}) \supset \mathit{continue}(\boldsymbol{sp})$, which contradicts that $\boldsymbol{p}$ is phase-fixing. □

The following overall picture of universal continuations can be extracted from proposition 18. Let $\boldsymbol{u} = r_1r_2r_2... \in C^\infty$ be universal. Let $\boldsymbol{u}_i := r_1...r_i$. From (iii) and (iv) it follows that $\boldsymbol{u}$ "phase-locks" at some point, i.e., there exists some i_0, such that $\boldsymbol{u}_j$ is phase-fixing for all $j \geq i_0$, and $\boldsymbol{u}_j$ is not phase-fixing for $j < i_0$. Furthermore, every phase φ appears infinitely often in $\boldsymbol{u}$, i.e., $\varphi = \varphi_{\boldsymbol{u}_i}$ for infinitely many indices i.

Phases can be used as local states for a generator of C, which is thereby uniquely determined:

Definition 19: Let C be a coherent language. Then $G_\varphi = (\Phi(C), \mathit{trans}_\varphi)$ is the **phase generator** of C, where $\mathit{trans}_\varphi := \{(\varphi_{\boldsymbol{p}}, r, \varphi_{\boldsymbol{pr}}) \mid \boldsymbol{p}$ is phase-fixing in C, $\boldsymbol{pr} \in C\}$.

Using the facts collected in proposition 18, it is straightforwardly confirmed that G_φ is well-defined and in fact a finite, cyclic generator of C. G_φ has the following properties:

Proposition 20:

(i) For $\varphi \in \Phi(C)$, let $\varphi^+ \subseteq C$ denote the set of all words which can be derived in G_φ by starting from φ. Then, $\varphi_{\boldsymbol{p}}{}^+ = \mathit{continue}(\boldsymbol{p})$.

(ii) $\varphi \neq \varphi'$ iff $\varphi^+ \neq \varphi'^+$.

(iii) G_φ is "deterministic": if $\varphi s\varphi_1$, $\varphi s\varphi_2$ are transitions in G_φ, then $\varphi_1 = \varphi_2$.

(iv) $\boldsymbol{r} \in C$ is phase-fixing iff all derivations of $\boldsymbol{r}$ in G_φ terminate in a unique phase. This phase is $\varphi_{\boldsymbol{r}}$.

(v) If $\eta: G_\varphi \to G_\varphi$, then $\eta = \mathrm{id}$.

(vi) The following is a useful characterization of phase generators. Let $G = (S, \mathit{trans})$ be a cyclic generator of C. Then, $G = G_\varphi$ (up to isomorphism) iff the following conditions hold:

(a) G is "deterministic", i.e., if xsy, xsz are transitions in G, then $y = z$.

(b) A local state $x \in S$ exists, which is fixed in G by a suitable $\boldsymbol{s} \in C$, i.e., $\exists\, x \in S, \boldsymbol{s} \in C$: if $\boldsymbol{s}$ can be derived in G on a path terminating in y, then $y = x$.

(c) Let, for $z \in S$, $z^+ \subseteq C$ denote the set of all words which can be derived in G by starting from z. Then, $\forall\, x, y \in S$: $x = y$ iff $x^+ = y^+$.

Sketch of proof:

(i) is straightforward.

(ii) is a corollary to (i).

(iii) Follows from 18(iv) and the construction of $trans_\varphi$.

(iv) "$\Rightarrow$": Let $\boldsymbol{r} \in C$ be phase-fixing. Choose φ such that $\boldsymbol{r}$ can be derived on a path $D_{\boldsymbol{r}}$ terminating in φ. Let $\boldsymbol{s} \in C$ be also phase-fixing. It follows from the construction of $trans_\varphi$ that $\boldsymbol{s}$ can be derived on a path $D_{\boldsymbol{s}}$ terminating in $\varphi_{\boldsymbol{s}}$. From the cyclicity of G_φ, it follows that there exists a word $\boldsymbol{t} \in C$ such that $\boldsymbol{str}$ can be derived on a path $D_{\boldsymbol{s}}D_{\boldsymbol{t}}D_{\boldsymbol{r}}$, which is a concatenation of three paths, the first of which is $D_{\boldsymbol{s}}$ and the last of which is $D_{\boldsymbol{r}}$. (i) says that $continue(\boldsymbol{s}) = \varphi_{\boldsymbol{s}}^+$. Use this and (iii) to concluded that $continue(\boldsymbol{str}) = \varphi^+$. Since $\boldsymbol{r}$ is phase-fixing, it holds that $continue(\boldsymbol{r}) = continue(\boldsymbol{str})$. Apply (i) and (ii) to conclude that $\varphi = \varphi_{\boldsymbol{r}}$.

"$\Leftarrow$": Let $\boldsymbol{r}$ be not phase-fixing. Then there exist two different phases $\varphi_{\boldsymbol{sr}}$ and $\varphi_{\boldsymbol{tr}}$. From "$\Rightarrow$" it follows that $\boldsymbol{sr}$ can be derived on a path ending in $\varphi_{\boldsymbol{sr}}$, and $\boldsymbol{tr}$ on a path ending in $\varphi_{\boldsymbol{tr}}$. Therefore, $\boldsymbol{r}$ can be derived on paths terminating in different local states.

(v) Let $\varphi = \varphi_{\boldsymbol{p}}$ be a phase such that φ^+ is maximal, i.e., there exists no other phase φ' such that $\varphi^+ \subset \varphi'^+$. Let $\eta(\varphi) = \varphi_{\boldsymbol{q}}$. From the definition of homomorphisms it follows that $continue(\boldsymbol{p}) \subseteq continue(\boldsymbol{q})$. Since φ^+ is maximal, this means that $continue(\boldsymbol{p}) = continue(\boldsymbol{q})$, i.e., $\eta(\varphi) = \varphi$. Apply (iii) to conclude that $\eta = id$.

(vi) "$\Rightarrow$": Assume $G = G_\varphi$. (a) follows from (iii). For (b), select for x an arbitrary phase φ, and for $\boldsymbol{s}$, some word that fixes φ. (c) is clear.

"$\Leftarrow$": First, we show

(1) $\forall\, y \in S\ \exists\, \varphi \in \Phi(C)$: $y^+ = \varphi^+$.

Let $x \in S$ be fixed by $\boldsymbol{s} \in C$ according to (b). Select some $y \in S$. Since G is cyclic, some $\boldsymbol{t} \in C$ exists such that $\boldsymbol{st}$ can be derived on a path terminating in y. Because G is deterministic, it follows that y is fixed by $\boldsymbol{st}$ in the sense of (b). From proposition 18(i) it follows that some $\boldsymbol{r} \in C$ exists, such that $\boldsymbol{rst}$ is phase-fixing for some phase $\varphi \in \Phi(C)$. Obviously, y is fixed by $\boldsymbol{rst}$. Therefore, $y^+ = continue(\boldsymbol{rst}) = \varphi^+$.

For $y \in S$, let φ_y denote the phase that corresponds to y according to (1). We now show

(2) $\forall\, \varphi \in \Phi(C)\ \exists^{=1}\, y \in S$: $\varphi = \varphi_y$.

Select some $\varphi \in \Phi(C)$. Let φ be fixed by $\boldsymbol{p} \in C$, i.e., $\varphi = \varphi_{\boldsymbol{p}}$. Select some derivation of $\boldsymbol{p}$ in G, which starts in some $x \in S$ and terminates in some $y \in S$. Analogously to the procedure in the proof of (1), select some $\boldsymbol{r} \in C$ that fixes x in G. Then, $\boldsymbol{rp}$ fixes y. It follows that $y^+ = continue(\boldsymbol{rp}) = continue(\boldsymbol{p}) = \varphi^+$, i.e., $\varphi = \varphi_y$. The uniqueness of y follows from (c).

(1) and (2) together yield an obvious one-to-one correspondence between S and $\Phi(C)$, which directly leads to an isomorphism between G and G_φ. □

Note that phases do not correspond to minimal sets of possible continuations. The case can occur that $\varphi^+ \subset \varphi'^+$. A related fact is that the phase generator of *C* need not have either the minimal possible number of local states or the minimal number of transitions. As an example consider the coherent language

$$C = \{r_1...r_n \in \{a, b, c\}^* \mid n \geq 1, r_i = r_{i+1} \Rightarrow r_i = a\} \cup \{\varepsilon\}.$$

I.e., *C* contains the words over {*a*, *b*, *c*} where only *a* may be directly repeated. There are three phases φ_a, φ_b, φ_c. It holds that φ_b^+, $\varphi_c^+ \subset \varphi_a^+$. Figure 3.3 depicts G_φ, and a generator G of *C* with two local states. The minimal number of states and transitions is obtained not in the phase generator but in G.

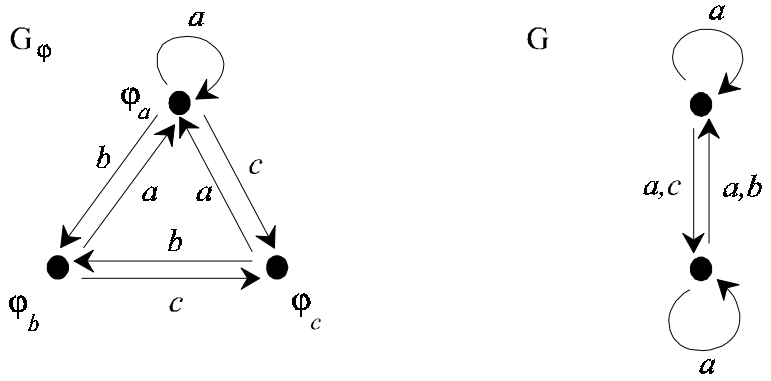


Fig. 3.3: Two generators of a coherent language.

Considering, once again, the background interpretation of words as local observations of a dynamic system, I ask now what is the *information* gained by such an observation. The term is generally used in two fashions. First, it can refer to the "semantic contents" of some observation (where, again, "observation" can be used in many fashions). For instance, when I "observe" that someone says, "the cat is on the mat", then the information conveyed by this statement might be that the cat is, indeed, on the mat. Second, the term can refer to a numerical value of the informativeness of an observation, which is usually given in bits. I will adopt the term *meaning* for the first usage, and the term *information* for the second.

There are several reasonable ways of how one might specify the meaning of a word from some coherent language *C*. I commit myself to a specification that, again, sees a word as an observation of a dynamic system. I define the meaning of a word (say, ***s***) as what is known about the internal state of the system, given the word as an observation that reaches from somewhere in the past up to the present. I.e, the meaning of ***s*** the set of phases which are reachable by derivations of ***s*** in G_φ,. The information, then, of ***s*** is maximal, when this set contains only one phase, i.e., when ***s*** is phase-fixing; it is 0 when ***s*** tells us nothing, i.e. when the set coincides with $\Phi(C)$. The value is computed, in the customary manner, as the negative

logarithm of the ratio of confirmed cases over possible cases. All in all, one obtains the following definition:

Definition 21: Let C be a coherent language, and $\boldsymbol{s} \in C$.

(i) The **meaning** of $\boldsymbol{s}$ with respect to C is the set
$\Phi(\boldsymbol{s}) = \{\varphi \in \Phi(C) \mid \varphi = \varphi_{\boldsymbol{rs}}$ for some phase-fixing $\boldsymbol{rs} \in C\}$.

(ii) The **information** of $\boldsymbol{s}$ with respect to C is the real number
$H(\boldsymbol{s}) = -\log_2 |\Phi(\boldsymbol{s})|/|\Phi(C)|$.

The information H will play an important role in accounting for self-organization in self-organizing scenes and streams.

Now I return to more practical matters. A coherent language will often be specified by some ad-hoc generator. I describe now an algorithm, which computes the phase generator from an arbitrary finite cyclic generator. Readers who are familiar with finite automata will find that the algorithm resembles somwhat to the customary power set construction of a deterministic finite automaton from a non-deterministic one.

Algorithm 22: Let G = (S, *trans*) be a finite cyclic generator of C. Then, the phase generator $(\Phi(C), trans_\varphi)$ can be computed in four steps. Figure 3.4 illustrates the procedure with an exemplary run.

Step 1: Construct from G a generator G' = (S', *trans*') of C, which is deterministic in the sense of proposition 20(vi/a), as follows. The local states of G' are certain elements of the power set of S. Construct S' and *trans*' incrementally by computing

$$S'_0 := \{S\},\ trans'_0 := \varnothing,$$
$$S'_{n+1} := S'_n \cup \{T \subseteq S \mid \exists\, R \in S'_n,\ s \in \Sigma\colon T = \{y \in S \mid \exists\, x \in R\colon xsy \in trans\}\},$$
$$trans'_{n+1} := trans'_n \cup \{RsT \mid R \in S'_n,\ T \in S'_{n+1}, \text{ and } \exists\, x \in R,\ y \in T\colon xsy \in trans\}.$$

Since S'_n and $trans'_n$ are monotonously increasing, and since the power set of S is finite, the sequence $(S'_n, trans'_n)$ eventually becomes stationary, i.e., some minimal k exists such that $(S'_k, trans'_k) = (S'_{k+l}, trans'_{k+l})$ for all $l \geq 0$. Put $G' = (S', trans') := (S'_k, trans'_k)$. It is easy to confirm that G' is a deterministic generator of C.

Step 2: Construct from G' a cyclic, deterministic generator G" = (S", *trans*") of C, as follows. Select some $T \in S'$ which is minimal, i.e., which is not a proper subset of some other $T^{\sim} \in S'$. Compute S" incrementally by putting

$$S''_0 := \{T\},$$
$$S''_{n+1} := S''_n \cup \{R \in S' \mid \exists\, U \in S''_n,\ s \in \Sigma\colon UsR \in trans'\}.$$

Again, this sequence becomes stationary for some minimal m. Put $S'' := S''_m$, and $trans'' := trans' \restriction S''_m$. This finishes the computation of step 2.

It remains to be shown that step 2 is correct. Since G" is a substructure of G', it is deterministic. In order to show that G" is, indeed, a cyclic generator of C, we select some $\boldsymbol{p} \in C$, such that $\boldsymbol{p}$ can be derived in G' on a path starting in S and terminating in T. We show that

(1) $\boldsymbol{p}$ fixes T in G' (in the sense of proposition 20(vi/b)).

Assume that ***p*** can be derived on a path starting in some $R_1 \in S'$ and terminating in some $R_2 \in S'$. We have to show that $R_2 = T$. The following general monotonicity law is easily confirmed: if $U_1 s U_2$ and $U_1' s U_2'$ are transitions in G', and $U_1 \subseteq U_1'$, then $U_2 \subseteq U_2'$. Clearly, $R_1 \subseteq S$. Apply the monotonicity law repeatedly for all transitions in both derivations of ***p*** in order to concluded that $R_2 \subseteq T$. Since T is minimal, this means that $R_2 = T$. In analogy to the argument used in the proof of proposition 20(vi), conclude from (1):

(2) A phase φ exists such that $T^+ = \varphi^+$.

Since $L_{G''} = \{\boldsymbol{s} \mid \boldsymbol{s}$ is a subword of some $\boldsymbol{t} \in T^+\}$ and $C = \{\boldsymbol{s} \mid \boldsymbol{s}$ is a subword of some $\boldsymbol{t} \in \varphi^+\}$, it follows that G'' generates *C*. In order to show that G'' is cyclic, select some $R \in S''$. It suffices to show that some path in G'' leads from R back to T. Some ***s*** exists which can be derived in G'' on a path starting in T and terminating in R. φ^+ contains some word of the form ***srp***. Use (1) to conclude that a derivation of ***rp*** leads from R back to T, as desired.

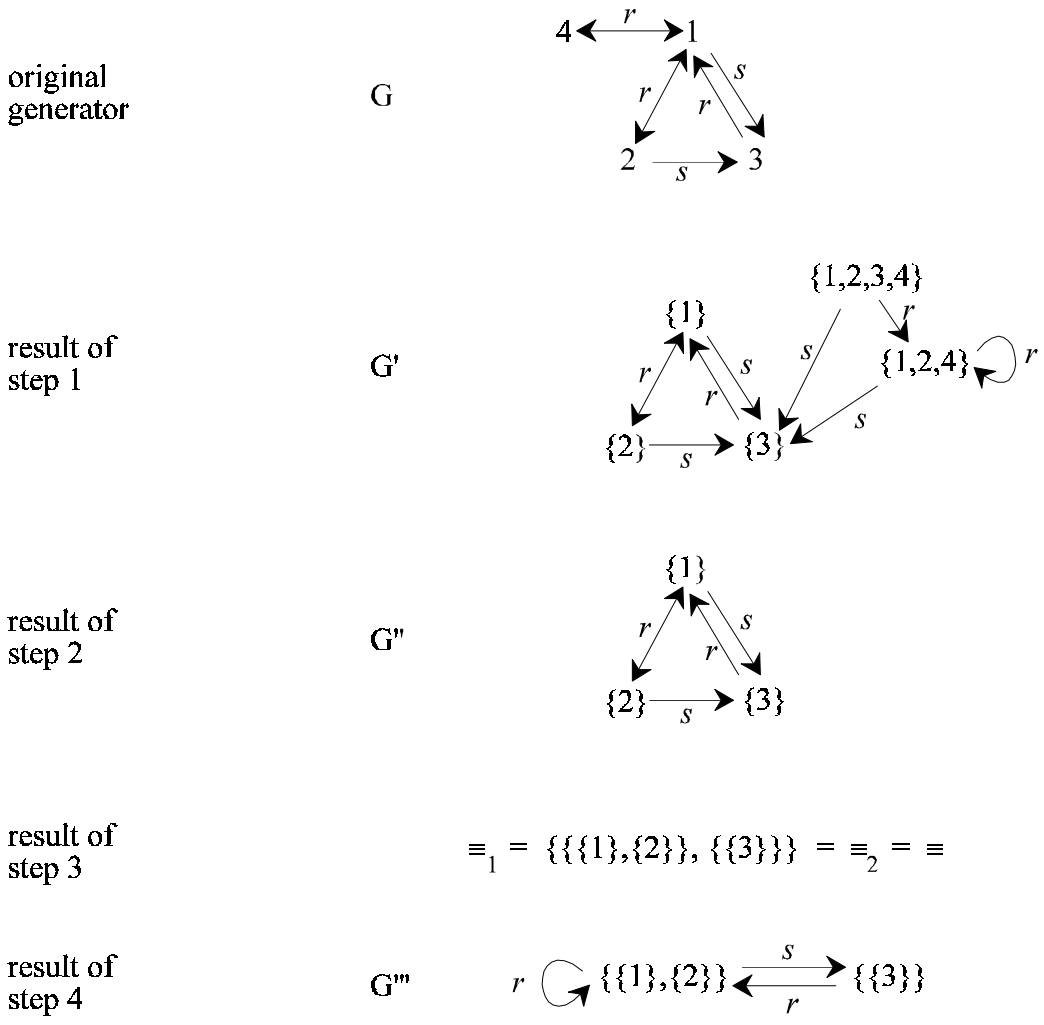


Fig. 3.4: An examplary application of algorithm 22.

Step 3: Compute the equivalence relation $\equiv$ on S'', defined by $T \equiv U$ iff $T^+ = U^+$, as follows. Define $\equiv_1$ such that

$$T \equiv_1 U \text{ iff } \forall\, s \in \Sigma\colon s \in T^+ \leftrightarrow s \in U^+.$$

For i = 2, 3, ..., define $\equiv_i$ inductively such that

$$T \equiv_{n+1} U \text{ iff } T \equiv_n U, \text{ and } \forall\, TsT', UsU' \in \mathit{trans}''\colon T' \equiv_n U'.$$

Effective procedures for computing $\equiv_1$ and $\equiv_{n+1}$ are obvious. Each $\equiv_{n+1}$ is a refinement of $\equiv_n$. Since each $\equiv_i$ can contain at most $|S''|$ equivalence classes, the sequence $(\equiv_i)_{i=1,2,..}$ must become stationary after some $i_0 \leq |S''|$. Put $\equiv := \equiv_{i_0}$.

For the correctness of step 3, we must show, for all $T, U \in S''$, that $T \equiv U$ iff $T^+ = U^+$. Define $T^{+n} := \{\boldsymbol{s} \in T^+ \mid |\boldsymbol{s}| \leq n\}$. It is easily confirmed that $T^{+n} = U^{+n}$ iff $T \equiv_n U$. Use this to conclude

$$\begin{aligned} T \equiv U &\text{ iff } T \equiv_{i_0} U \\ &\text{iff } \forall n\colon T \equiv_n U \\ &\text{iff } \forall n\colon T^{+n} = U^{+n} \\ &\text{iff } T^+ = U^+. \end{aligned}$$

Step 4: Construct $G_\varphi = (\Phi(C), \mathit{trans}_\varphi)$ by "$\equiv$-factorizing" G''. More precisely, construct a generator $G''' = (S''', \mathit{trans}''')$ by putting

$$S''' := \{T^{\equiv} \mid T \in S'', T^{\equiv} \text{ is the equivalence class of } T\},$$
$$T^{\equiv} s U^{\equiv} \in \mathit{trans}''' \text{ :iff } \exists\, U' \in S''\colon U' \equiv U \text{ and } TsU \in \mathit{trans}''.$$

Applying proposition 20(vi), it is easily confirmed that G''' is isomorphic to G_φ. □

The natural morphism for coherencies is not the homomorphism defined above, but a mapping which allows to "spool" one generator into another. The next definition specifies such mappings, called *simulations*. The term is borrowed and adapted from the theory of non-wellfounded sets (cf. Aczel 1988), where "bisimulations" are mappings between directed graphs, with very much the same intentions as here.

Definition 23: Let $G_1 = (S_1, \mathit{trans}_1)$, $G_2 = (S_2, \mathit{trans}_2)$ be coherencies generating C_1 and C_2. A mapping σ from S_1 to the power set $\boldsymbol{P}(S_2)$ is a **simulation** of G_1 in G_2 :iff

(i) $\sigma(x)$ is nonempty for all $x \in S_1$,
(ii) for all $x_1 r x_1' \in \mathit{trans}_1$, for all $x_2 \in \sigma(x_1)$ there exists $x_2' \in \sigma(x_1')$ such that $x_2 r x_2' \in \mathit{trans}_2$.

This is written as $\sigma\colon G_1 \to G_2$.

The next proposition lists two direct consequences of this definition:

Proposition 24:
(i) Simulations are transitive.
(ii) When $\sigma\colon G_1 \to G_2$ is a simulation of G_1 in G_2, then $C_{G_1} \subseteq C_{G_2}$. □

Two cyclic generators of a coherent language need not simulate each other. In the example shown in fig. 3.1, the generator G_1 cannot be simulated in G_2. However, every cyclic generator of a coherent language can be simulated in the corresponding phase generator:

Proposition 25: If G = (S, *trans*) is a coherency generating *C*, and G_φ is the phase generator of *C*, then there exists a unique simulation σ: G $\rightarrow$ G_φ.

Sketch of proof: Define σ by $\varphi \in \sigma(x)$:iff there exists a phase-fixing $\boldsymbol{p} \in C$ such that $\varphi = \varphi_{\boldsymbol{p}}$ and $\boldsymbol{p}$ can be derived in G on a path terminating in x. It is straightforward to show that σ: G $\rightarrow$ G_φ. To show that σ is unique, consider an arbitrary simulation σ': G $\rightarrow$ G_φ and some $x \in S$. Let $\boldsymbol{p}$ be phase-fixing such that in G there exists a derivation of $\boldsymbol{p}$ terminating in x. Use 23(ii) and 20(iv) to show that $\varphi_{\boldsymbol{p}} \in \sigma'(x)$, which means that $\sigma'(x) \supseteq \sigma(x)$. Now take an arbitrary $\varphi \in \sigma'(x)$ and a phase-fixing $\boldsymbol{p}$ for which there exists a derivation in G starting and ending in x. Use again 23(ii) and 20(iv) to show that $\varphi = \varphi_{\boldsymbol{p}}$, which means that $\sigma'(x) \subseteq \sigma(x)$. Thus, one has $\sigma'(x) = \sigma(x)$. Since x was selected arbitrarily, this implies $\sigma' = \sigma$. □

The next theorem is the key for any deeper analysis of coherencies and their interrelationships.

Simulation theorem 26: Let C_1, C_2 be coherent languages. Let $G_{1\varphi}$ = (S_1, $trans_1$) and $G_{2\varphi}$ = (S_2, $trans_2$) be their phase generators. Then $C_1 \subseteq C_2$ iff there exists a simulation σ: $G_{1\varphi} \rightarrow G_{2\varphi}$.

Sketch of proof:
"⇐": follows from 24(ii).
"⇒": This requires some work. Let $\boldsymbol{u} = u_1u_2...$ be a universal continuation in C_1. From $C_1 \subseteq C_2$ it follows that $\boldsymbol{u} \in C_2^\infty$. $\boldsymbol{u}$ can therefore be derived on an infinite path in $G_{2\varphi}$. Fix some such derivation $D_{G_{2\varphi}}(\boldsymbol{u}) = x_0u_1x_1u_2x_2...$. Since $G_{2\varphi}$ is finite, there exists some i_0 such that for all $i \geq i_0$, x_i appears infinitely often in $D_{G_{2\varphi}}(\boldsymbol{u})$. Since $\boldsymbol{u}$ is a universal continuation in C_1, there exists some j_0 such that for all $i \geq j_0$, $u_1...u_i$ is phase-fixing in C_1. Let m := max $\{i_0, j_0\}$. Let $D_{G_{1\varphi}}(\boldsymbol{u}) = y_0u_1y_1u_2y_2...$ be a derivation of $\boldsymbol{u}$ in $G_{1\varphi}$. Then y_i is uniquely determined for $i \geq m$ since $m \geq j_0$. Let G' = (S', *trans*') be the substructure of $G_{2\varphi}$ which consists of the local states x_m, x_{m+1}, x_{m+2}, ... and the transitions $x_mu_{m+1}x_{m+1}$, $x_{m+1}u_{m+2}x_{m+2}$, ... Define σ': $S_1 \rightarrow \boldsymbol{P}(S')$ by $x \in \sigma'(y)$:iff there exists $i \geq m$ such that $x = x_i$ and $y = y_i$. Observe that σ' is well-defined since every $y \in S_1$ appears (even infinitely often) in $y_mu_{m+1}y_{m+1}u_{m+2}y_{m+2}...$ due to the choice of j_0. In order to prove "⇒" it obviously suffices to show

(1) σ' simulates $G_{1\varphi}$ in G'.

Define another generator G" = (S", *trans*") by

$S'' := \{x_y \mid x \in \sigma(y)\}$,
$(x_y, r, x'_{y'}) \in trans''$:iff $(y, r, y') \in trans_1$ and $(x, r, x') \in trans'$.

G" has the following properties:

(2) G" is a cyclic generator of C_1.
(3) G" is deterministic in the sense of 20(iii), i.e. if $x_yrx'_{y'}$, $x_yrx''_{y''} \in trans''$, then $x'_{y'} = x''_{y''}$.

(2) follows from that on the one hand, $\boldsymbol{u}$ can be derived in G" in a fashion which "uses" every transition infinitely often due to the choice of $m \geq i_0$ (which implies that G" is cyclic and that in G" there can be derived at least the words from C_1), and that on the other hand, $\varphi(x_y) := y$

defines a homomorphism from G" to $G_{1\varphi}$ (which implies that in G" there can be derived at most the words from C_1). (3) is inherited from $G_{2\varphi}$.

Define σ": $S_1 \rightarrow \boldsymbol{P}(S")$ by $x_y \in \sigma"(z)$:iff z = y. In order to show (1) it obviously suffices to show that σ": $G_{1\varphi} \rightarrow G"$ is a simulation. It is easily confirmed that σ" is well-defined and that 23(i) holds. Thus 23(ii) remains to be shown, i.e.

(4) for all $zrz' \in trans_1$, for all $x_y \in \sigma"(z)$, there exists $x'_{y'} \in \sigma"(z')$ such that $x_y r x'_{y'} \in trans"$.

Due to the definition of G" this is equivalent to

(4') for all $yry' \in trans_1$, for all $x_y \in \sigma"(y)$, there exists $x'_{y'} \in \sigma"(y')$ such that $x_y r x'_{y'} \in trans"$.

Now comes the crucial argument. Define

$$k := \min \{n \in \mathbb{N} \mid \text{there exists } \boldsymbol{p} \in C_1 \text{ which is phase-fixing in } C_1 \text{ and whose possible derivations in G" terminate in n different } x_y \in S"\}$$

Let $yry' \in trans_1$ and $x_y \in \sigma"(y)$. It has to be shown that there exists $x'_{y'} \in \sigma"(y')$ such that $x_y r x'_{y'} \in trans"$. Let $\boldsymbol{q} = q_1...q_l \in C_1$ be phase-fixing in C_1 such that the possible derivations of $\boldsymbol{q}$ in G" terminate in k different local states from S". Since G" is cyclic, there exists a finite continuation $\boldsymbol{qt} = q_1...q_l t_{l+1}...t_n$ of $\boldsymbol{q}$, one of whose derivations in G" terminates in x_y.

The following holds:

(5) The possible derivations of $\boldsymbol{qt}$ in G" terminate at exactly k different local states from S".

In order to show (5), assume that there were k' > k local states from S" which are termination points of derivations of $\boldsymbol{qt}$. Use (3) to conclude that there must exist at least k' local states from S', which are possible termination points for derivations of $q_1...q_l t_{l+1}...t_{n-1}$. Iterate this argument to show that there exist at least k' local states from S" that are possible termination points for derivations of $q_1...q_l$. This is a contradiction.

Since $\boldsymbol{qt}$ can be derived in G" terminating in x_y, it can be derived in $G_{1\varphi}$ terminating in y (this follows from a simple argument involving φ). Since $yry' \in trans_1$, it holds that $\boldsymbol{qt}r \in C_1$. Assume that there exists no $x'_{y'} \in \sigma"(y')$ such that $x_y r x'_{y'} \in trans"$. Then no derivation of $\boldsymbol{qt}$ in G", which terminates in x_y, can be continuated by r. Use (3) and (5) to conclude that there exist at most k-1 local states in S" that are possible termination points for derivations of $\boldsymbol{qt}r$ in G". This contradicts the definition of k. Therefore, there exists $x'_{y'} \in \sigma"(y')$ such that $x_y r x'_{y'} \in trans"$. □

The simulation theorem is a remarkable result. "$C_1 \subseteq C_2$" is a brute extensional statement, which at a first glance does not appear to be technically well manageable. This impression is nourished, for instance, by a "density theorem" that I mention without proof: if $C_1 \subset C_2$, then there exist "interpolating" $C_1 \subset C \subset C_2$ with arbitrarily large phase generators. I.e., chains of the type $C_1 \subset C_2 \subset C_3$ ... are arbitrarily "erratic" with respect to the size of the corresponding phase generators. Yet, such chains correspond to chains of simulations.

The following is an ad hoc demonstration of the theorem's handiness:

Proposition 27: For two coherent languages C_1, C_2 with given generators G_1, G_2, it is decidable whether $C_1 \subseteq C_2$.

Sketch of decision procedure: Construct the phase generators $G_{1\varphi}$, $G_{2\varphi}$, and test whether there exists an involution σ: $G_{1\varphi} \rightarrow G_{2\varphi}$. This test can be carried out, e.g., by trying out all mappings from $\Phi(C_1)$ to $\boldsymbol{P}(\Phi(C_1))$. (Of course, since coherent languages are regular, one could also make use of decision procedures for inclusion of regular languages.)□

Summary of 3.1

- Interpreting words of formal languages as local observations of dynamic systems motivates consideration of segmentable languages, i.e., languages that are closed under subwords.
- In segmentable languages, words act as filtering contexts for their potential continuations.
- Segmentable languages are conveniently described by generators, i.e., edge-labeled directed graphs, where paths yield derivations of words.
- Regular segmentable languages are characterized by having finite generators.
- Coherent languages are regular segmentable languages that have cyclic finite generators, which are called coherencies.
- For coherent languages, there exist universal continuations, i.e., infinite words that contain all words of the language infinitely often. They can be interpreted as observations of a dynamic system that performs a random walk.
- When a coherency is an epimorphic image of another coherency, the latter can be interpreted as a symmetry breaking of the former.
- Phases are sets of words that can be interpreted as the semantic contents of maximally informative observations of an oscillation generating system. They yield the local states of a normal form of generators, i.e., phase generators.
- Phases are used to specify the notions of meaning and information of words, where the former concerns the contents of the knowledge provided by a word (interpreted, again, as an observation) about an observed system, and the latter a numerical value.
- Phase generators can be effectively constructed from arbitrary finite cyclic generators.
- The "natural" morphisms for coherencies are simulations. The simulation theorem states that a coherent language is a sublanguage of another coherent language iff there exists a simulation between their phase generators.

3.2 Dynamic symbol spaces

Dynamic symbol spaces are the DSS model of "long-term memory". The latter term is slightly misleading, since long-term memory is commonly associated with the conceptual level of an intelligent agent, whereas DSS assigns to every level on the periphery-centre axis its own "long-term memory". Unfortunately, there seems to be no better term available to denote the

fact that in some level or module, dynamic symbols are processed according to some long-term, stable "law".

Throughout this and the subsequent subsections, I rely on the conceptual level as a source of examples, because concepts are the most directly accessible type of dynamic symbols for human readers. Thus, the dynamic symbols to be used in examples will typically be rendered by formal symbols (i.e., symbols from the observer's theory) like `apple`, `paradise`, or `buy`. In making use of such examples, I tacitly assume that there exists a reliable observation procedure for the corresponding physical dynamic symbols in an agent, and that the external reference mechanism of these dynamic symbols is empirically explained (cf. sections 2.4, 2.5). For instance, the dynamic symbols in question might be neocortical activation patterns; however, empirical neuroscience is not yet in a state to record such patterns on-line with high resolution. A variety of conceptual-level dynamic symbols, which is easier to detect, are uttered words. This is, however, problematic for another reason: words that are uttered reflect only a fraction of what happens within the agent on a conceptual level, and they will do so in a sequential form. A third variety of conceptual-level dynamic symbols is electronic activation patterns in an artificial agent that is programmed symbolically on its conceptual level (in particular, a programming technique based on DSS might be used). These patterns can be reliably detected by their traces on a computer screen. The last kind of dynamic symbols that I want to mention here are the concepts at work in the researcher him- or herself. The observation procedure is introspection, a method whose value I hesitate to judge. Be this as it may, the important thing to note is that an `apple` symbol appearing in an example is not considered as a classical, "platonic" category name, but as the formal rendering of an observable informational entity that plays a dynamic role in an agent's information processing.

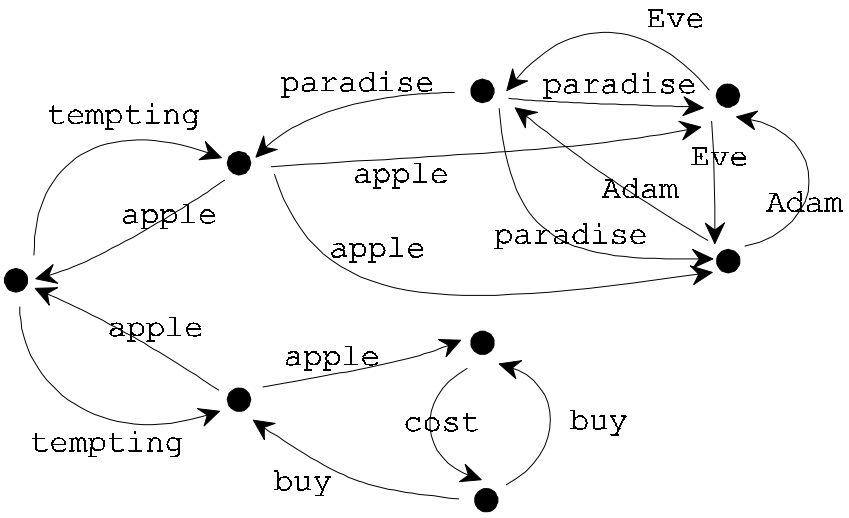


Fig. 3.5: A coherency as it might appear in a dynamic symbol space.

Dynamic symbols appear in a dynamic symbol space within coherencies. In subsection 3.1, coherencies have been treated in an abstractly formal way. To afford the reader with a less formal intuition, fig. 3.5 gives an example of how one of the coherencies in a dynamic symbol space (on the conceptual level) could look like.

The intuition behind this example is to see it as a generating device for words, i.e., sequences of dynamic symbols like `apple cost buy` or `tempting apple Eve`. In practice, such sequences might be observed, e.g., in free association tasks.

Observe, in particular, the filtering effects of contexts. For instance, the sequences `buy apple` and `apple Eve` are each derivable, whereas the concatenation `buy apple Eve` is not: the continuation `Eve` of `apple` is filtered out when `apple` lies in the context of `buy`. This simple mechanism for contextual influences on "associations" of dynamic symbols is one of the central features of DSS. It grows directly out of the notion of segmentable languages (cf. proposition 4), which in turn is derived directly from the background interpretation of words as local observations.

Furthermore, observe that any two dynamic concepts can be interlinked by a suitable intermediate sequence. For instance, `buy apple tempting apple Eve` is derivable. The context `buy`, which formerly ruled out `apple Eve`, is, one might say, "shadowed" by `tempting`, which serves as a "semantic bridge" between a supermarket and a garden-of-Eden setting. Using cyclic generators, i.e., coherencies, as the basic component of dynamic symbol spaces is generally motivated by the considerations made in the preceding subsection before definition 11. When the conceptual level is concerned in particular, an additional motivation comes from the cyclic closedness that is characteristic for conceptual systems (cf. section 2.4).

Dynamic symbol spaces are, however, more complex than a single coherency. They consist in a two-dimensional array of coherencies. Each of the dimensions contributes a particular aspect to what is classically called "abstraction".

One way of approaching dynamic symbol spaces is to consider them as rough equivalents of classical semantic networks for terminological knowledge, e.g., KL-ONE T-boxes. There are two basic differences.

The first concerns the *structure* of dynamic symbol spaces vs. semantic networks. In the latter, there exists a single abstraction dimension. The former, by contrast, are two-dimensional. Both dimensions are related to classical abstraction, but they distinguish two factors of it. The first dimension arises from the abstraction of dynamic symbols as observables, as discussed in 2.6. The second dimension accounts for the fact that a given dynamic symbol can interact with others in varying degrees of precision. This dimension is technically effected by a sequence of symmetry breakings. It can be interpreted in terms of a computational temperature: when it is "hot", the interactions are close to random, when it is "cool", they are specific and context-sensitive.

The second difference between classical semantic networks and dynamic symbol spaces concerns their *use*. A classical semantic network is accessed locally via the concepts contained in it, with information pertaining to the accessed concept being read out. It can in principle be accessed at several levels of abstraction simultaneously. A dynamic symbol space, by contrast, can rather be compared to the global state space of a thermodynamic system. Its two dimensions provide *global states* which are interpreted as global processing modes for the level or module whose "long-term memory" it is. A helpful metaphor is to view the module as a chemical reactor, and the global state as the temperature and pressure setting that controls the processes in the reactor. A dynamic symbol space, thus, is "accessed" not at arbitrary points, but, for some lapse of time, in a single point of its two-dimensional space, which characterizes such a processing mode.

So much for a first intuitive impression of the whole. Turning to the technicalities, I first specify an abstraction relation for dynamic symbols in terms of an *abstraction tree*. The definition is a formal reconstruction of abstraction hierarchies of the kind discussed in 2.6 (cf. fig. 2.7). Note that the term "dynamic symbol" appears in the definition, instead of "symbol", as in the preceding, formal-language-oriented subsection. Fig. 3.6 provides an example.

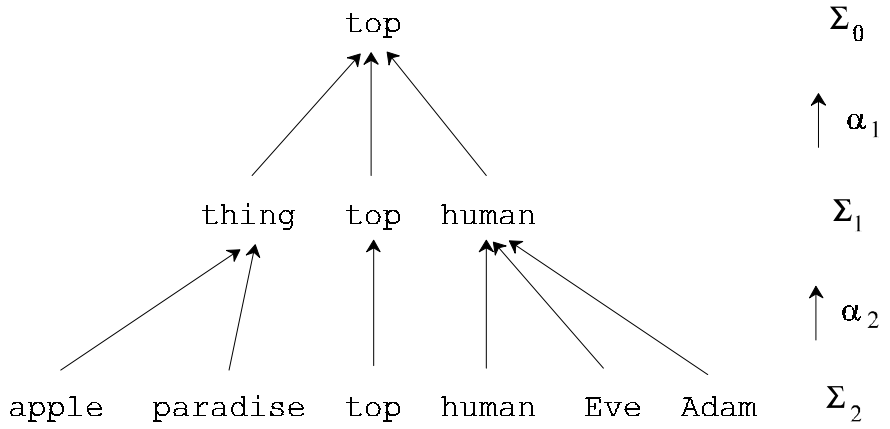


Fig. 3.6: An abstraction tree.

Definition 28: Let $\Sigma = \Sigma_n \cup \Sigma_{n-1} \cup ... \cup \Sigma_0$ be a set of dynamic symbols, and let α_j: $\Sigma_j \rightarrow \Sigma_{j-1}$ be a mapping. Then $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$ is an **abstraction tree** :iff

(i) each α_j is surjective,
(ii) if $r \in \Sigma_j \cap \Sigma_{j-k}$, then $r \in \Sigma_j \cap \Sigma_{j-1} \cap ... \cap \Sigma_{j-k}$ and $\alpha_l(r) = r$ for l = j, j-1, ..., j-k+1,
(iii) no α_j is the identity mapping.

The mappings α_j are called **abstraction mappings**. When $r = s$, or for some j, k it holds that $r \in \Sigma_j$, $s \in \Sigma_{j-k}$, and $s = \alpha_{j-k+1} \circ ... \circ \alpha_j(r)$, then s **subsumes** r. This is written as $s \geq r$.

Condition (i) implies that in intermediate Σ_j there cannot occur dynamic symbols that have no subsumees in the more specialized Σ_{j+k}. This reflects that an increase in resolution or differentiation cannot lead to the disappearance of an observable. Conversely, all dynamic symbols contained in Σ_j have subsumers in the more abstract Σ_{j-k}, since the abstraction mappings are totally defined. This reflects the assumption that poorer differentiation or resolution cannot make an observable disappear; at worst, still "something" is observed. This "something" is rendered in fig. 3.6 by `top`.

Condition (ii) ensures that a dynamic symbol cannot occur at distant levels of abstraction without occuring in all the intermediate levels, and that the abstraction mappings are the identity for dynamic symbols occuring in several of the Σ_j. This reflects an obvious "monotonicity" of observations of empirical entities: by sharpening resolution and/or differentiation, the information about an entity can only remain constant or increase. Note that the Σ_j need not be disjoint; this reflects that neither an increase in resolution, nor in differentiation, neccessarily yields a more informative observation of a particular dynamic symbol.

Furthermore, note that in this definition no mention is made of the particular resolution and differentiation contributions to abstraction. The formalism reflects only the net result of these factors. The notions of resolution and differentiation belong to the background interpretation of DSS.

The next two definitions prepare the specification of dynamic symbol spaces, which will be given in definition 32. Definition 29 describes how a generator that uses dynamic symbols from Σ_n can be stepwise abstracted via an abstraction tree.

Definition 29: Let $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$ be an abstraction tree, and $G_n = (S, trans_n)$ a generator, where $trans_n \subseteq S \times \Sigma_n \times S$. For j = n-1, n-2, ..., 0, let $\beta_j := \alpha_{j+1} \circ \alpha_{j+2} \circ \ldots \circ \alpha_n$. For j = n-1, n-2, ..., 0, define $G_j = (S, trans_j)$ by

$$trans_j := \{x\beta_j(r)y \mid xry \in trans_n\}.$$

This is more briefly written as $G_j = \beta_j(G_n)$ or, equivalently, as $G_j = \alpha_{j+1}(G_{j+1})$. Then, $(G_j)_{j=n,...,0}$ is an **abstraction sequence of** G_n **with respect to** $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$.

Abstraction sequences of generators are reflected in the generated languages in an obvious fashion:

Proposition 30: Let $(G_j)_{j=n,...,0}$ be an abstraction sequence of G_n with respect to $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$. Then, $L_{G_j} = \{\beta_j(r_1)...\beta_j(r_k) \mid r_1...r_k \in L_{G_n}\}$. □

The next definition prepares the other dimension of dynamic symbol spaces.

Definition 31: Let $(G_i)_{i=m,...,0}$ be a sequence of finite cyclic generators, such that G_{i+1} is a symmetry breaking of G_i. Then $(G_i)_{i=m,...,0}$ is a **symmetrification sequence** of G_m.

The "space" spanned by an symmetrification sequence and an abstraction sequence is a *dynamic symbol space*:

Definition 32: Let G_{mn} be a finite cyclic generator, $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$ an abstraction tree, and $(G_{in})_{i=m,...,0}$ a symmetrification sequence of G_{mn}. Define for each i = m, m-1, ..., 0 an abstraction sequence $(G_{ij})_{j=n,...,0}$ of G_{in} with respect to $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$. Then $(G_{ij})_{i=0,...,m\ j=0,...,n}$ is a **dynamic symbol space**. The generators G_{ij} are called the **global states** of the dynamic symbol space.

A dynamic symbol space is, thus, a matrix (G_{ij}) of generators. In definition 32, this matrix is constructed from the rightmost lowest element G_{mn} by first establishing the rightmost column as an symmetrification sequence, and then by specifying the rows from their rightmost element via an abstraction sequence. Fig. 3.7 provides an example. Derivation paths in its global states should be interpreted as temporal associations of dynamic symbols, which could be observed, e.g., on instructing a subject to utter words that come into his or her mind while thinking about human emotions.

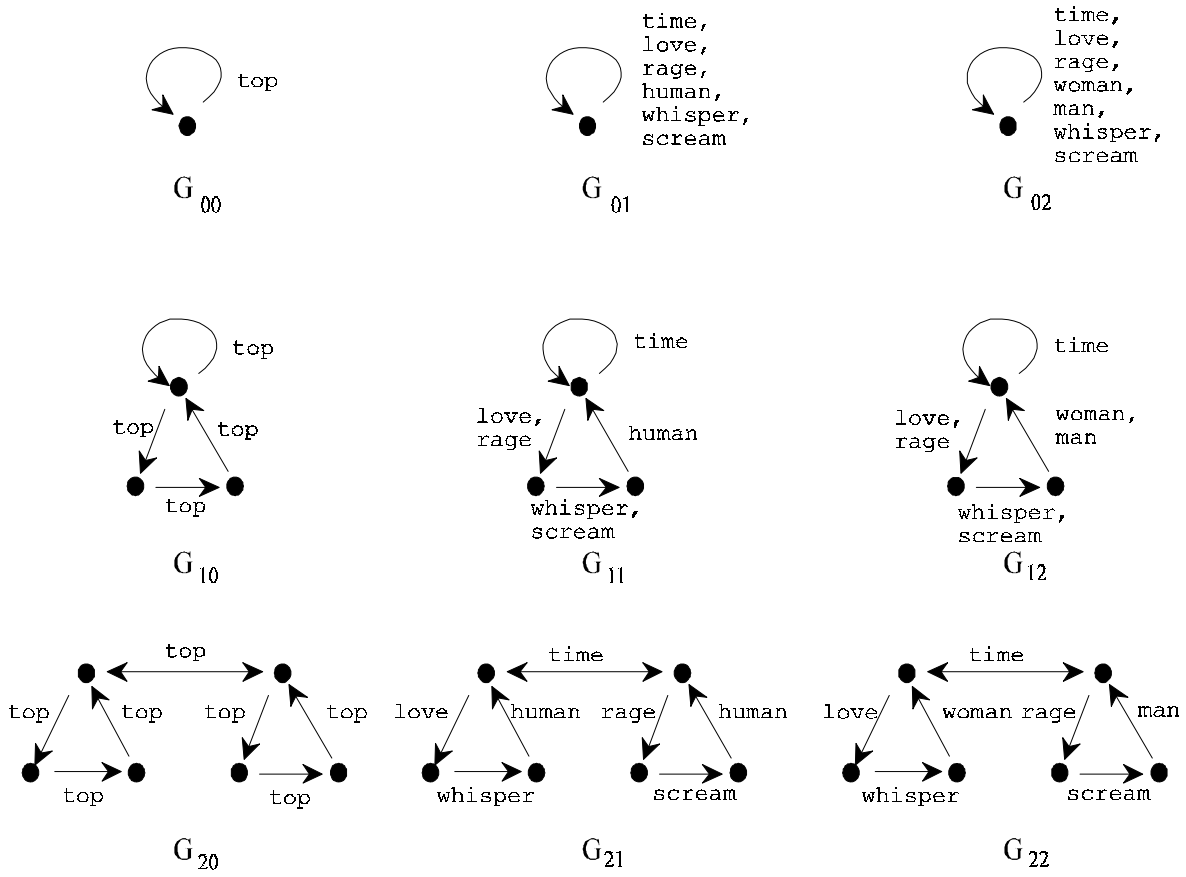


Fig. 3.7: A dynamic symbol space. The horizontal dimension is induced by abstraction of dynamic symbols (from right to left), the vertical dimension to symmetry breakings (downward).

In definition 32, only the rightmost column of (G_{ij}) is required to be a symmetrification sequence. However, it is straightforwardly confirmed that all other columns are symmetrification seqences, too. Thus, going down in (G_{ij}) means symmetry breaking, going right means dynamic symbol specialization:

Proposition 33: Let (G_{ij}) be a dynamic symbol space. Then $G_{ij} = \alpha_{j+1}(G_{ij+1})$, and G_{ij} is a symmetry breaking of G_{i-1j}. □

Both the abstraction and symmetrification sequence dimension carry aspects of what is understood outside DSS as "abstraction". This should be clear for the abstraction dimension of dynamic symbol spaces. As the symmetrification dimension is concerned, shifting down on the symmetry breaking axis means that although the same dynamic symbols are used, they are so in a more context-sensitive manner. For instance, in G_{12} (fig. 3.7) the sequence `woman rage` can be derived, whereas in G_{22} the dynamic symbol `rage` is filtered out by the context `woman`. In other words, the *use* of `rage` is more general in G_{12} than in G_{22}. In classical semantic networks, a generalization of usability is connected with classical abstraction. For instance, in applications that rely on a KL-ONE terminologic knowledge base, it is always permissible to substitute a concept by any of its superconcepts, but not vice versa; i.e., superconcepts are more generally usable than subconcepts. In classical systems, this generalization of use is always coupled to a generalization of the very concepts concerned, whereas in DSS these aspects are treated separately in the two dimensions of dynamic symbol spaces.

Working with dynamic symbol spaces needs some habituation, when one is used to classical semantic networks. In the remainder of this subsection, I comment on several points which help to get a firmer grasp on these spaces.

Global states as "processing modes"

In the perspective of DSS, information processing is a collective process, in which many dynamic symbols interact. The site where such interactions occur are self-organizing scenes or streams (introduced in subsequent subsections). They are the DSS model of a conceptual-level "working memory", or another information processing module elsewhere on the periphery-centre axis. Each such module comes with its own dynamic symbol space.

A self-organizing stream is, in the DSS view, more like a chemical reactor than like an algorithm or a calculation machine. Staying in the metaphor, information processing is assumed to be influenced by global parameters, just like a chemical reaction is influenced by pressure and temperature. Such global parameters could be, for instance, tiredness, performance pressure, or somatic arousal.

DSS offers several parameters which can be set to control the global processing conditions of a self-organizing stream. Two of these parameters are the dimensions of (G_{ij}). A self-organizing stream is modeled as a collective process for which a global state G_{ij} acts as a global "processing mode". While processing is going on, the global state can be shifted, thereby driving the collective process in its entirety along a trajectory in the corresponding dynamic symbol space.

A very brief note on learning

A dynamic symbol space can always be expanded by adding new columns at its right side, and (when G_{mn} is not a single cycle graph) new rows can be added at the bottom. This means that a dynamic symbol space can be further elaborated. Since a dynamic symbol space is analogous to long-term memory, an elaboration of it is a kind of learning. However, learning mechanisms are not included in the present state of the DSS formalism, which focusses on short-term dynamics.

Balance of the two dimensions

It seems plausible to assume that in many natural or artificial systems, the two dimensions of (G_{ij}) are not entirely decoupled. Specialized dynamic symbols will, as a tendency, be used in a specific manner. In classical semantic networks, this coupling is strict. In dynamic symbol spaces, global states that lie near the diagonal in (G_{ij}) are particularly "balanced" in this sense. The diagonal can be seen as the region in (G_{ij}) that most closely resembles classical abstraction hierarchies. When the coherencies from the diagonal are isolated from the rest, and written one

below the other, what one gets is quite similar to a classical semantic network. Note, however, that (G_{ij}) need not be quadratic. Thus, "diagonal" is an approximate notion.

The strongest possible interdependence occurs when the abstraction sequence corresponds exactly to the symmetrification sequence. This happens when the contextual filtering effects, which are governed by the symmetrification dimension, correspond exactly to abstraction. Such special dynamic symbol spaces are called *balanced*.

For an illustration, assume that `apple` occurs in the contexts `paradise` and `buy` in some global state that lies left from the diagonal (say, in G_{21}), i.e., the words `paradise apple` and `buy apple` are derivable in G_{21}. This "contextual promiscuity" of `apple` in an off-diagonal global state would, in a balanced dynamic symbol space, disappear in G_{22}. There would be two different specializations of `apple`, say, `apple_of_Eden` and `apple_as_merchandise`, corresponding to the two contexts. Generally, a balanced dynamic symbol space is quadratic, and in a coherency on the diagonal, there is a one-to-one correspondence between dynamic symbols and local states. Transitions leading into a particular local state are all labeled by the same dynamic symbol, and every dynamic symbol occurs as a label of transitions that lead into a unique local state. The "breaking" of local states in going down the symmetry breaking dimension corresponds exactly to specializations of dynamic symbols. Figure 3.8 sketches the example.

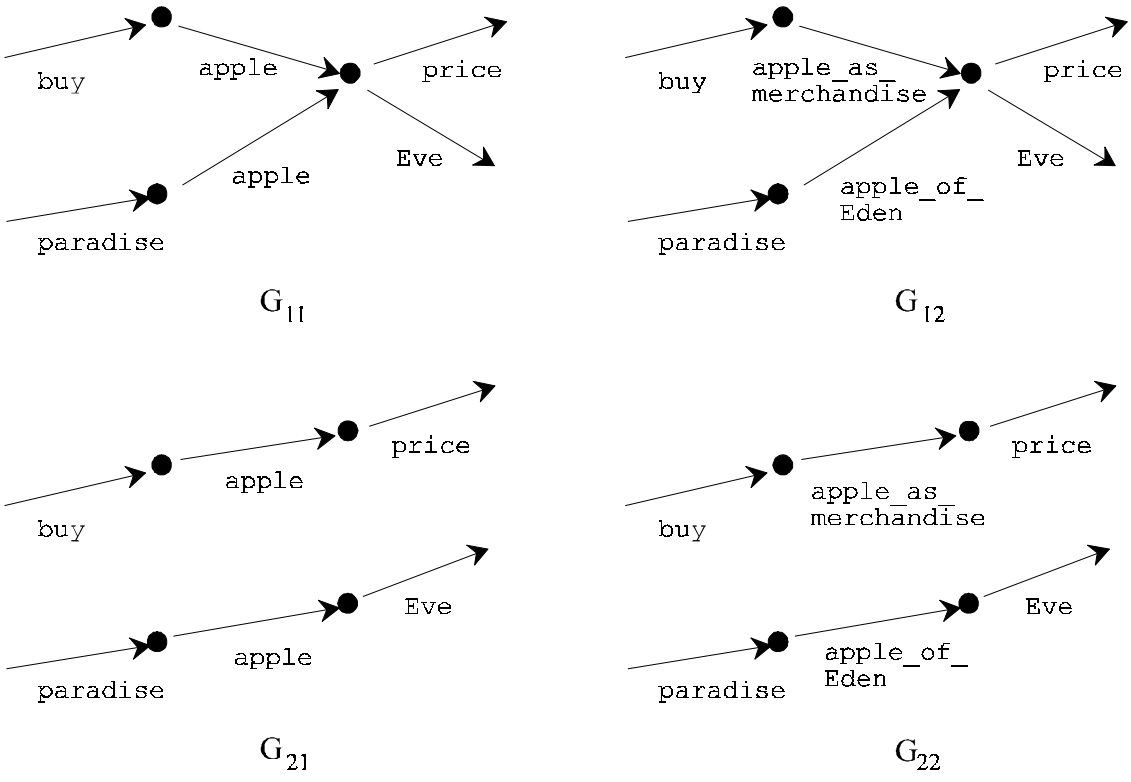


Fig. 3.8: A portion of a balanced dynamic symbol space. Parts of G_{11}, G_{12}, G_{21}, and G_{22} are shown. In the transition from G_{11} to G_{22}, the local state characterized by the `apple` transition that leads into it becomes broken into two local states. Thus, `apple` becomes specialized into two versions, namely, `apple_of_Eden` and `apple_as_merchandise`. The other local states and dynamic symbols are not subject to symmetry breaking or specialization in this example.

The definition of a balanced dynamic symbol space is now obvious:

Definition 34:

(i) A generator G = (S, *trans*) is **state-transition-balanced** :iff
$\forall$ x*s*x', y*r*y' $\in$ *trans*: *s* = *r* $\leftrightarrow$ x' = y'.

(ii) A dynamic symbol space (G_{ij}) is **balanced** :iff it is quadratic, and the generators on the diagonal are state-transition-balanced.

The following proposition is easily verified:

Proposition 35: A balanced dynamic symbol space has the following properties:

(i) It is already determined by either G_{nn} and the abstraction sequence, or by G_{nn} and the symmetrification sequence alone. It is thus essentially one-dimensional.

(ii) For $j \geq i$, G_{ij} features no contextual filtering, i.e., *continue*(***s***) = *continue*(***rs***) for all ***rs*** $\in C_{G_{ij}}$.

(iii) For $j \geq i$, in $G_{ij} = (S_j, trans_{ij})$ it holds that for all x*s*x', y*r*y' $\in$ $trans_{ij}$, if *s* = *r*, then x' = y'. A local state z is thus uniquely determined by any of the dynamic symbols *r* that occur as labels of transitions leading into the local state. This can be written as $z = z_r$.

(iv) For $i \geq j$, in $G_{ij} = (S_j, trans_{ij})$ it holds that for all x*s*x', y*r*y' $\in$ $trans_{ij}$, if x' = y', then *s* = *r*. A dynamic symbol *s* is thus uniquely determined by any of the local states z into which it can lead in a transition. This can be written as $s = s_z$.

(v) For every dynamic symbol specialization in the left triangular submatrix of (G_{ij}) there exists a corresponding "context specialization" in the right triangular submatrix and vice versa. I.e., if a dynamic symbol *r* from G_{ij} becomes specialized to r_1, ..., r_k in G_{ij+l} (both G_{ij} and G_{ij+l} in the left triangular submatrix), then the local state x_r in G_{ji} becomes broken into x_{r_1}, ..., x_{r_k} in G_{j+li}, and vice versa. □

It is debatable to what extent natural dynamic symbol spaces (on the conceptual level) are balanced. The phenomenon seems common enough: a subconcept typically corresponds to a contextual differentiation of usage of its superconcept.

Cognitive interpretation of the two dimensions

In which situations will an intelligent agent be in processing modes represented by global states from different areas of (G_{ij}), and what will induce a shift of these modes? Some relevant observations can be gleaned from everyday experience.

- A factor that influences the position on the horizontal (i.e., abstraction) axis is the agent's familiarity with the situation at hand. When a situation is relatively incomprehensible, specialized dynamic concepts from the agent's stock will be inapplicable. The processing mode will thus be pushed towards the left in (G_{ij}), where dynamic concepts are more general.
- Another factor concerning this dimension springs from communicative demands. When the agent communicates with a relative novice, it is forced to recur to general categories shared by both parties. For instance, a scientist explaining his or her approach to a grant-deciding government executive will take care to use simple concepts. Note that even with abstract dynamic concepts, it is possible to articulate specific information, by working in

global states from the lower region of (G_{ij}), where the use of dynamic concepts is highly context-sensitive. One can express subtle things in simple terms.

- On the lower levels of signal processing, the horizontal axis reflects the quality of signals. When signals are noisy, it is difficult to extract highly specific features. A module that processes these features will be shifted towards the less specialized left region of its (G_{ij}).
- On a conceptual level, the vertical axis reflects effects of mental acuity. When a human is in a smalltalk situation, or simply tired, s/he will tend to use concepts in an undifferentiated manner, i.e. s/he will be driven towards the upper regions in (G_{ij}). A similar effect can be expected from time pressure or other stress factors.
- In dreaming, concepts are often used in a fashion where contexts "cross over". This again is an instance of modes from the upper regions of (G_{ij}). The famous nonsense example found in many linguistic textbooks, "colorless green ideas sleeping furiously", is of this dream kind; it obviously has been invented in a processing mode far up in (G_{ij}).
- Low-level modules processing signal features will be shifted upwards in their mode when the features are, for themselves, precise, but mutually contradictory, such that they cannot be interpreted in a conclusive manner.

"Thermodynamic" interpretation of the two dimensions

DSS, being a well-defined model of a particular brand of complex dynamic systems, should be equipped with a principled interpretation in terms of statistical mechanics and information theory. Only then will effects of self-organization be properly understood (cf. section 6).

Two preliminary observations on this behalf are suggestive. First, it seems that both dimensions of a dynamic symbol space contribute to the entropy of self-organizing streams. This is, of course, not a precise statement, since an entropy remains to be defined. It is, however, reasonable to expect that an entropy measure will reflect the average information of words occuring in streams. In the average, the information of a word grows when the global state is shifted towards the bottom (symmetry breaking). Similarly, when it is shifted to the right (specialization of dynamic symbols), the specialized version of a word will be, in the average, more informative. At its present state, the formalism is not mature enough to prove whether (or when) the increase of word information is strictly monotonous with global state shift. The reason for this unsatisfying situation is that the information of a word is defined in terms of phase generators, whereas a dynamic symbol space is made from arbitrary generators.

Second, the vertical dimension (i.e., the symmetry breaking dimension) seems to correlate with temperature. Again, this remains a metaphorical statement as long as there is no well-defined temperature measure. A closer look at this dimension reveals, however, that this choice of metaphor seems the right one.

Consider two global states G_{ij} and G_{i+kj}. A word $\boldsymbol{r}$ derivable in G_{ij} is a concatenation $\boldsymbol{r} = \boldsymbol{r}_1...\boldsymbol{r}_n$ of shorter words $\boldsymbol{r}_1$, ..., $\boldsymbol{r}_n$ from G_{i+kj}. Seen the other way round, $\boldsymbol{r}_1...\boldsymbol{r}_n$ can be regarded as being the result of an "extraordinary" derivation procedure in G_{i+kj}. In such extraordinary derivations, it is allowed to "jump" from certain local states to certain other ones. It is suggestive to ascribe this jumping to some "local excitatory energy" available to the derivation procedure (again, "energy" remains to be defined). In statistical thermodynamics, temperature is interpreted by the average energy of micro-entities. By analogy, this would support an interpretation of G_{i+kj} as being "hotter" than G_{ij}. This also conforms with the common usage in

emergent computation approaches of calling the superposition of noise on computations a "computational temperature".

With due reserve, such observations indicate a route that a future analysis of DSS in terms of statistical thermodynamics might take. In section 6, a related approach (Smolensky's "harmony theory") is examined, where such an analysis is fully carried out. Harmony theory is, on this behalf, a model for DSS.

Monotonic concept specialization

Monotonic concept specialization is mainly achieved by two mechanisms in classical semantic networks. First, information can be *added* to a concept in order to specify a subconcept. Second, information can be *passed down* from concepts to subconcepts, becoming specialized on the way. Examples are role restrictions and role differentiations in KL-ONE (Brachman & Schmolze 1985). All these mechanisms have counterparts in dynamic symbol spaces.

The inheritance of information, which becomes specialized itself, is a direct consequence of abstraction and symmetrification. For instance, the word (i.e., the sequence of dynamic symbols) `apple pick` might be abstracted to `fruit harvest` by going left on the abstraction dimension. The subsumee `apple` of `fruit` inherits the continuation `harvest`, which becomes specialized to `pick` in the process. This corresponds directly to KL-ONE role differentiation. For an anlogue of role value restriction, compare `fruit harvest plant` and `apple harvest apple-tree`. The former word can occur in a dynamic symbol space by virtue of an abstraction of the latter (i.e., going left), or by virtue of a combined abstraction and symmetrification (i.e., going left and up). Interpreting `harvest` as a role that relates `fruit` to `plant`, it is restricted in its second argument from `plant` to `apple-tree`.

The other mechanism of specialization, i.e. the introduction of new information, is not so easily accomplished. Suppose, for example, that `fruit` (in some G_{ij}) becomes specialized to `apple` (in G_{ij+1}), and `apple` is to be equipped with the continuation `fall-of-Man`. There seems to be no likely subsumer of `fall-of-Man` which could serve as a continuation of `fruit` in G_{ij}, so `fall-of-Man` has to be newly attached to the `apple` context in G_{ij+1}.

A technique for doing so is to exploit the `top` concept as a source for `fall-of-Man`. One would require that there exist a dynamic symbol `top`, which subsumes all others. Furthermore, one would demand that there exist an expansion `fruit top` in G_{ij}. The desired continuation `apple fall-of-Man` in G_{ij+1} is then achieved by the canonical abstraction mapping α_{j+1}(`apple`) = `fruit` and α_{j+1}(`fall-of-Man`) = `top`. I.e., the "addition" of new information is not really an addition out of the blue; technically, it is a specialization of the already available, completely unspecific, `top`. To make this mechanism generally applicable, one has to require that for every dynamic symbol *r* in every global state there exists the continuation *r* `top`. A further natural generalization leads to the notion of *conservative* abstractions:

Definition 36:

(i) An abstraction tree $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$ is **conservative** :iff $\Sigma_{j-1} \subseteq \Sigma_j$ for j = n, ..., 1, and $\Sigma_0 = \{\texttt{top}\}$.

(ii) $(G_j)_{j=n,...,0}$ is a **conservative** abstraction sequence of G_n with respect to $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$:iff

(1) $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$ is conservative,

(2) if $r_1...r_x \in C_{G_j}$, $r_i' \geq r_i$, then $r_1...r_i'...r_x \in C_{G_j}$.

(iii) A dynamic symbol space is **conservative** :iff it is constructed from a conservative abstraction sequence of G_{mn}.

In a conservative dynamic symbol space, "spontaneous" new continuations can be introduced in G_{ij+1} through a specialization of `top` in G_{ij}, or of other "generic" dynamic symbols in G_{ij} (like `thing` or `event`).

Conservative dynamic symbol spaces can be written down economically, without explicitly carrying all the abstract dynamic symbols through the entire abstraction dimension. For instance, when the dynamic symbol space from fig. 3.7 were taken as a shorthand notation of a conservative dynamic symbol space, then, e.g., G_{22} would tacitly be assumed to be, in fact, of the form depicted in fig. 3.9:

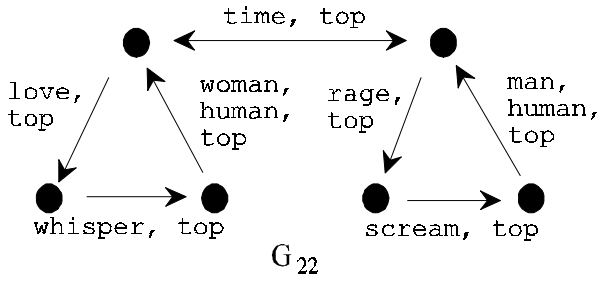


Fig. 3.9: G_{22} from fig. 3.7, as it would be tacitly interpreted if the dynamic symbol space were conservative.

From the perspective of the background interpretation of dynamic symbols as observables, conservative abstraction mirrors a situation where, regardless of the resolutive powers of the observation procedure, arbitrarily abstract findings will occur. This implies that in the observed system arbitrarily de-differentiated entities are present. For dynamic symbol spaces as models of "long-term memory" this means that even when the processing module is in a processing mode from the bottom right area of (G_{ij}), arbitrarily abstract dynamic symbols can occur. In a cognitivistic interpretation of these dimensions, this corresponds to the assumption that even in a state of mental acuity, one uses, amongst more specialized concepts, their abstract superconcepts as well, which is plausible. Conservative dynamic symbol spaces appear natural at a conceptual level of information processing; in this sense, the monotonic inheritance mechanism that exploits conservativism is natural, too.

Birds fly, Tweety is a bird, Tweety flies, penguins don't fly, Tweety is a penguin!, Tweety folds its wings and swims. Figure 3.10 shows how this can be modeled in a dynamic symbol space.

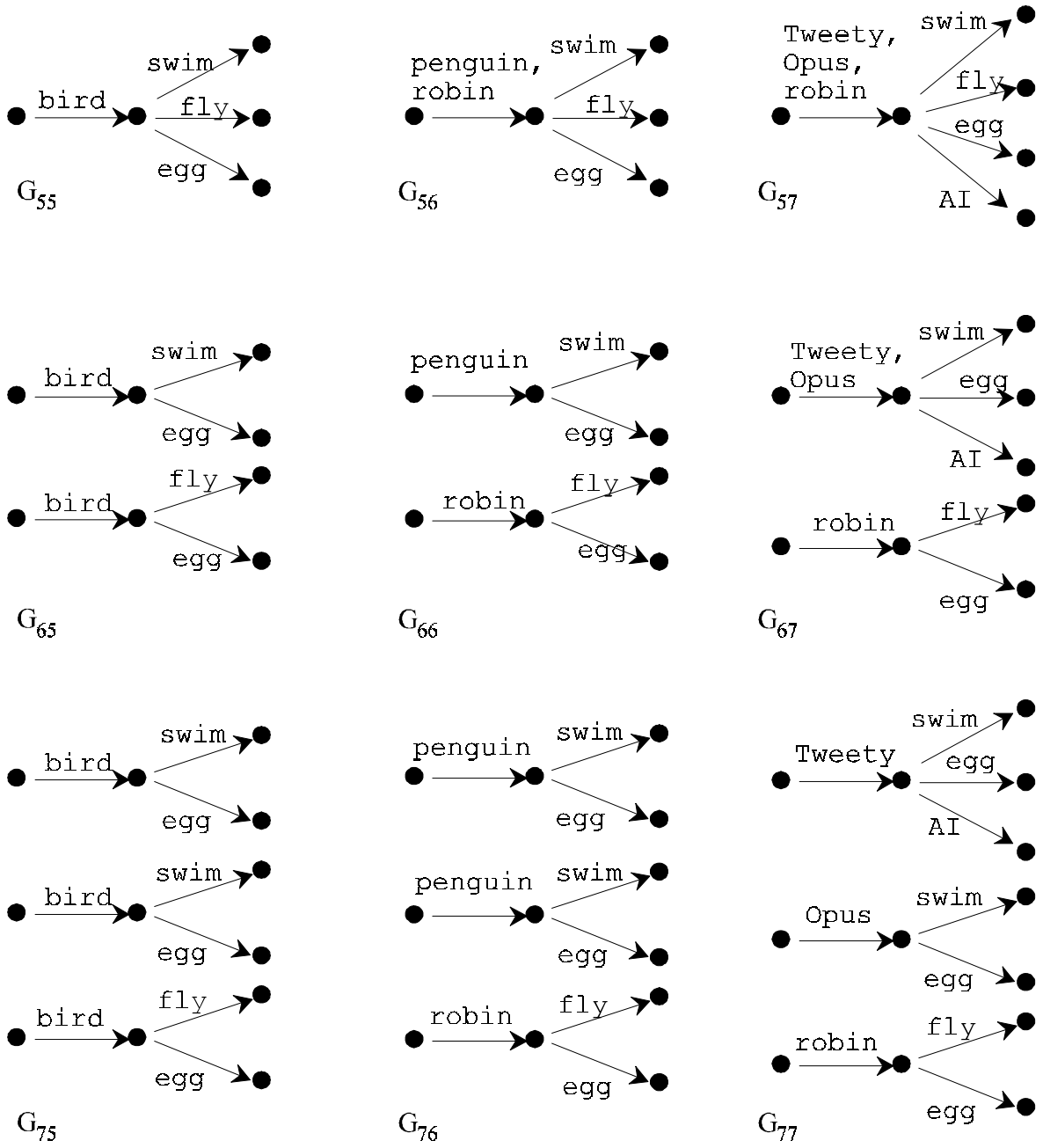


Fig. 3.10: Flightless waterfowl in a dynamic symbol space. Small portions of some of the G_{ij} are shown. `top` transitions are assumed to accompany other transitions; they are not shown.

Several points concerning this example might need some explanation:

- `Tweety` and `Opus` are dynamic symbols just as well as `bird`. DSS has no extensional semantics; thus, there are no individuals or instances in the classical sense (more about this in the next subsection).
- The `AI` concept appearing in column 7 has been monotonically added by a specialization of suitable `top`-transitions in column 6. The dynamic symbol space is assumed to be conservative; the economic presentation mentioned above is adopted, which hides abstract dynamic symbols in specialized global states.
- Nonmonotonicity is a combined effect of symmetrification and abstraction. Penguins are construed as flightless derivates of birds by first breaking the `bird` transition into two variants, one of which swims and one of which flies, in the shift from G_{55} to G_{65}. These variants are then allocated to different dynamic symbols, i.e., to `penguin` and `robin`, in the step from G_{65} to G_{66}.

Summary of section 3.2

- Dynamic symbol spaces are the DSS model of "long-term memory".
- Every processing module or level in a multi-module or multi-level architecture is equipped with its own dynamic symbol space.
- A dynamic symbol space's basic components are coherencies. They yield a fundamental mechanism for context-sensitivity, which is one of the central aspects of DSS.
- Roughly, a dynamic symbol space resembles a classical semantic network. But, the single classical abstraction dimension gets split into two. Formally, a dynamic symbol space is a two-dimensional array of coherencies. One dimension is due to abstractions of dynamic symbols, the other to symmetry breakings of coherencies.
- DSS focusses on fast self-organization phenomena. In its present state, it omits long-term learning.
- The coherencies in a dynamic symbol space yield global processing modes for self-organizing streams.
- In a cognitivistic interpretation, such modes should be interpreted to mental states like acuity or tiredness.
- In a thermodynamics-oriented interpretation, they correspond to global thermodynamic states. Symmetry breakings can be interpreted in terms of computational temperature.
- In the special case of balanced dynamic symbol spaces, both dimensions essentially collapse into one. Dynamic symbols are, then, essentially characterized by their mutual relationships.
- Monotonic and nonmonotonic inheritance can be reconstructed in dynamic symbol spaces.

3.3 Self-organizing scenes and streams

In this subsection, first a simple kind of a parallel dynamic system is described, namely, *self-organizing scenes.* They are closed systems without output or input. Obviously, closed systems are not suited as a model of an agent's subsystems. I introduce them mainly for expository reasons, in order to fix the basic dynamics mechanisms first, without having to bother from the outset about the complications afforded by input and output. A generalization to open systems (self-organizing *streams*) is derived afterwards, by way of constructing a (not quite trivial) input/output mechanism.

A self-organizing scene is always equipped with a dynamic symbol space, which serves as its "long-term memory". This dynamic symbol space describes how certain "orderly" sequences of dynamic symbol can be derived. However, such derivations of orderly sequences are *not* the dynamics in self-organizing scenes. In self-organizing scenes, the information is represented (just like in dynamic symbol space) by directed graphs with edges labeled by dynamic symbols. Again, just like in the dynamic symbol space, sequences of dynamic symbols can be derived in these graphs. But, these sequences are typically "disordered". They are concatenations of mixed sequence fragments from the underlying dynamic symbol space. The dynamics of a scene shuffles, duplicates, deletes, completes, abstracts or specializes these fragments in a fashion whose net effect is that the "order" in the derivable sequences increases. I.e., increasingly long and interconnected sequences form, which "correctly" correspond to sequences derivable in the underlying dynamic symbol space. This is likely to be a rather fast effect, and the scene is likely to equilibrate swiftly to a stadium where it locally resembles a coherency from its dynamic symbol space. In a nutshell, self-organization consists in the local convergence of a disordered graph towards the order prescribed by long-term memory.

So much for a first intuition. More specifically, self-organizing scenes rest on the following ideas:

- A self-organizing scene is a directed graph whose edges are labeled by dynamic symbols from a dynamic symbol space (G_{ij}).
- A self-organizing scene develops in time through self-organizing, collective processes, which are incessantly active.
- The informational entities that interact in an self-organizing scene are words which essentially come from a particular global state in (G_{ij}). They are called *associations*. The interaction of associations is stochastic and asynchronous. It is effected by a single kind of local operation, *microchanges*.
- Global "working parameters" of a self-organizing scene are set by selecting a global state from the corresponding dynamic symbol space. The selected global state rules certain details of microchanges. It can be "shifted" in (G_{ij}), which yields *macrodynamics*. Macrodynamic change is externally imposed on an self-organizing scene. It is typically slow in comparison with microdynamic change. A suitable analogy from physical systems is the temperature and pressure control of a chemical reactor.

This frame is now worked out in detail. I begin with some preparatory definitions centered around the notion of associations.

For the remainder of subsections 3.3 and 3.4, let $(G_{ij})_{i=0,...,m\ j=0,...n}$ be a dynamic symbol space with an underlying abstraction tree $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$, where $\Sigma = \Sigma_0 \cup \Sigma_1 \cup ... \cup \Sigma_n$, and let $G_{ij} = (S_j, trans_{ij})$.

Definition 37:

(i) Let $r_1...r_k$, $s_1...s_k \in \Sigma^k$. Then $s_1...s_k$ **subsumes** $r_1...r_k$:iff $s_i \geq r_i$ for i = 1, ..., k.

(ii) Let $r \in \Sigma_{j+k}$, $s \in \Sigma_j$, $s = \alpha_{j-1} \circ ... \circ \alpha_{j+k}(r)$. Then s is the Σ_j-**abstraction** of r.

(iii) Let $r \in \Sigma$, $0 \leq j \leq n$. Then $r^{(j)}$ denotes the Σ_j-abstraction of r, in case that $r \in \Sigma_i$ for some $i \geq j$. In other cases, put $r^{(j)} := r$.

Comment: It is easily confirmed that this definition of $r^{(j)}$ is independent from the choice of a particular $i \geq j$. Intuitively, the $^{(j)}$-operation "lifts" dynamic symbols belonging to some abstraction level that is more specific than level j, to the level j. For technical conveniance, the $^{(j)}$-operation is also defined when r cannot be subsumed by any $s \in \Sigma_j$. Dynamic symbols can only become abstracted by the $^{(j)}$-operation, not specialized. Using the abstraction tree from figure 3.6, examples are `thing`$^{(2)}$ = `thing`, or `Eve`$^{(1)}$ = `human`.

(iv) $s_1...s_k \in \Sigma^k$ is a G_{ij}-**association** :iff $k \geq 1$, and $s_1^{(j)}...s_k^{(j)}$ subsumes some $r_1...r_k \in C_{G_{ij}}$. $r_1...r_k$ is called a G_{ij}-**interpretation** of $s_1...s_k$.

Comment: Intuitively, $s_1...s_k$ is a G_{ij}-association if it can be made a word from $C_{G_{ij}}$ by abstracting the s_i which are too special for Σ_j, and by specializing the s_i which are too abstract for Σ_j. Note that the empty word does not count as an association.

(v) Let $s_1...s_k$ be a G_{ij}-association. Then the G_{ij}-**meaning** of $s_1...s_k$ is the set

$$\Phi_{ij}(s_1...s_k) = \{\varphi \mid \varphi \text{ is a phase in } C_{G_{ij}}, \text{ and } \varphi \text{ is a terminal point in some derivation of some } G_{ij}\text{-interpretation } r_1...r_k \text{ of } s_1...s_k \text{ in the phase generator of } C_{G_{ij}}\}.$$

(vi) Let $s_1...s_k$ be a G_{ij}-association. Then the G_{ij}-**information** of $s_1...s_x$ is

$$H_{ij}(s_1...s_k) = -\log_2 |\Phi_{ij}(s_1...s_k)|/M,$$

where M is the total number of phases in $C_{G_{ij}}$.

Comment: (v) and (vi) essentially recall definition 21, with technical modifications to account for the fact that a G_{ij}-association can correspond to several words from $C_{G_{ij}}$, i.e., that it can have several G_{ij}-interpretations.

(vii) Let $s_1...s_k t$, $s_1...s_k t'$ be G_{ij}-associations, where $t \geq t'$. Then t' is a G_{ij}-**specialization** of t **in the context** $s_1...s_k$:iff

$t' = t$, or
$t > t'$, and for all $t'' \in \Sigma$, where $t'' \neq t'$, $t \geq t''$, and $s_1...s_k t''$ is a G_{ij}-association, it holds that $t'' > t'$ or $t' > t''$.

t' is a **maximal** G_{ij}-**specialization** of t **in the context** $s_1...s_k$:iff there does not exist some $t'' < t'$ which is a G_{ij}-specialization of t in the context $s_1...s_k$.

Comment: It can be straightforwardly shown that a maximal G_{ij}-specialization of t in the context $s_1...s_k$ exists and is uniquely determined. It is simply called *the* G_{ij}-specialization of t in

the context $s_1...s_k$. In less technical terms, the G_{ij}-specialization t' of t in the context $s_1...s_n$ is the most specific subsumee of t which preserves $s_1...s_nt'$ as a G_{ij}-association, and which is still uniquely determined. In connection with a microchange, the G_{ij}-specialization of t in the context $s_1...s_n$ will be used to model the phenomenon that if some abstract t comes into the scope of a context $s_1...s_n$, the context can induce a specialization of t. For instance, when `fruit` comes into the context of `paradise Eve`, it should automatically be specialized to `apple`. Another example is provided when G_{12} from fig. 3.7 is used. There, e.g., `human` is the G_{12}-specialization of `top` in the context `time love top`. Note that `human` could be further specified: `time love top man` and `time love top woman` are also G_{12}-associations. But this further specialization destroys uniqueness and is, therefore, excluded in (vii).

Associations occur abundantly in the sequel. The next proposition collects some easily verified properties:

Proposition 38:

(i) If $s_1...s_k \in \Sigma^k$ subsumes $r_1...r_k$, and $r_1...r_k$ is a G_{ij}-association, then $s_1...s_k$ is a G_{ij}-association. I.e., abstraction preserves G_{ij}-associations.

(ii) For all indices ij, it holds that every word of length 1 from Σ^*, i.e., every $s \in \Sigma$, is a G_{ij}-association.

(iii) If $r_1...r_k$ is a G_{ij}-association, and $i' \leq i$, $j' \leq j$, then $r_1...r_k$ is a $G_{i'j'}$-association.

(iv) If $r_1...r_k$ is a G_{ij}-association, then every subword of $r_1...r_k$ is a G_{ij}-association.

(v) If G_{ij} contains only `top`-transitions (as in the leftmost column in fig. 3.7), or if it is a single-node, single-loop graph (as in the uppermost row in fig. 3.7), then every word from Σ^* is a G_{ij}-association. □

A self-organizing scene develops in time. This is technically managed by conceiving a scene as a sequence of (*closed*) *configurations*. A closed configuration is a phase coherency whose transitions are labeled by dynamic symbols from Σ:

Definition 39: Let Σ be a set of dynamic symbols. Let G be an arbitrary coherency whose transitions are labeled by dynamic symbols from Σ. Let C be the phase generator of the language generated by G. Then C is a **closed Σ-configuration**. When no misunderstandings are expected, C will be simply called a **configuration**.

Note that C is required to be a *phase* generator. The reasons for this will be discussed later in this subsection.

The term "configuration" is adopted from the theory of cellular automata (Wolfram 1986), which has, to some extent, co-influenced the DSS formalism. Both in cellular automata and in self-organizing scenes, a configuration is a structured pattern of discrete informational entities, which develops in time by virtue of local interactions. A crucial difference is that in self-organizing scenes, the very topology of the pattern is arbitrarily inhomogenous and changes in time (as will shortly become clear), whereas in cellular automata, the pattern is a temporally stable, homogenous grid.

In a configuration C, words from Σ^* can be derived that need have nothing to do with the languages $C_{G_{ij}}$ that are generated by the coherencies in the underlying dynamic symbol space. However, for each $G_{kl} \in (G_{ij})$, the words derivable in C can be interpreted piecewise as G_{kl}-associations:

Proposition 40: Let (G_{ij}) be a dynamic symbol space with an underlying set of dynamic symbols Σ, and let $G_{kl} \in (G_{ij})$. Let C be a Σ-configuration. Then, every word derivable in C can be segmented into subwords that are G_{kl}-associations.

This is clear, since the "trivial" segmentation into subwords of length 1 yields G_{kl}-associations (cf. 38(ii)). □

A configuration C can thus be perceived, for every $G_{kl} \in (G_{ij})$, as a network of interconnected, and possibly overlapping, G_{kl}-associations. At worst, these G_{kl}-associations all have length 1. When G_{kl} is shifted upwards or to the left in (G_{ij}) to become $G_{k'l'}$, then 38(iii) implies that words from C, which have been found to be G_{kl}-associations, will be re-established as $G_{k'l'}$-associations. When a derivation path in G_{kl}, which yields a G_{kl}-association ***r***, is of maximal length (i.e., it is not a proper subpath of another derivation of a G_{kl}-association), and when G_{kl} is shifted upwards or to the left in (G_{ij}) to become $G_{k'l'}$, then it is possible that the derivation can be properly extended and still yields a $G_{k'l'}$-association. In the extreme, when $G_{k'l'}$ is of the "degenerate" kind mentioned in 38(v), every path in C yields a $G_{k'l'}$-association. In sum, shifting G_{kl} upwards or to the left only increases the length of associations that can be found in C.

An illustrative way to see configurations is to consider them as snapshots of a working memory. A working memory is typically loaded with only a small subset of the concepts available in long term memory. Analogically, the set of dynamic symbols appearing in a configuration should be a small subset of Σ.

I want to emphasize again, in order to preclude a suggestive misunderstanding, that the generation of words, either in C or in some G_{ij}, is *not* what makes the dynamics of a self-organizing scene. The dynamics stems from an altogether different kind of operation, *microchanges*, which locally modify C, giving rise to a *history* $C_0, C_1, C_2, \ldots$. Each of these configurations is formally a generator of a language, but again, their "generating power" is not to be taken as a dynamic process but rather as an implicit description of a language. The right way to see $C_0, C_1, C_2, \ldots$ is not to look for the generation of words, but to watch the evolution of a language. The self-organization aspect of the history is that the languages $C_{C_0}, C_{C_1}, C_{C_2}, \ldots$ converge towards a sublanguage of $C_{G_{ij}}$, where G_{ij} is the coherency from the underlying (G_{ij}), which serves as the global processing mode for the history.

Microchanges are the crucial element in the whole affair. Before describing an algorithm for their computation, I sketch an intuitive picture of how a microchange works.

Assume that C_i is a configuration in a history, which is currently under the "control" of a global state $G_{kl} \in (G_{ij})$. A microchange essentially consists in "moving" a G_{kl}-association along a transition that branches from the association. Fig. 3.11 sketches such a move, which transforms C_i into C_{i+1}. The association that moves is marked by bold arrows, the transition selected as a "railway" for the move is shaded:

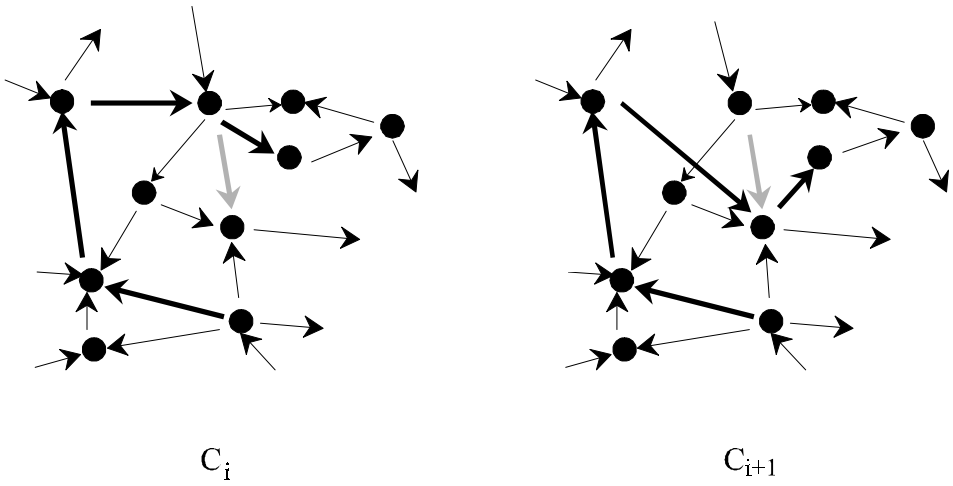


Fig. 3.11: A microchange. Only a portion of configurations is shown, with edge labels omitted.

The general intuition behind microchanges is that when many of them are executed, they thoroughly "mix" the configuration. Associations "migrate", using each other as "railways". Mere shuffling, however, would be of little use. Important side-effects are connected with microchanges. When a moving association hits a transition that can serve as a continuation of the association, the association grows in length. Associations, so to speak, "hunt" through the configuration (more correctly, through a sequence of configurations), "in search" of suitable continuations. They can also compete with each other, "catching" pieces from each other. Furthermore, labels of transitions can become specialized or abstracted to fit better with passing associations. All these effects are accounted for in the definition of a microchange, which, therefore, is not a simple one. The following definition gives the basic version of the microchange algorithm.

Algorithm 41: Let C = (S, *trans*) be a closed Σ-configuration. Let $G_{kl} \in (G_{ij})$. Then a G_{kl}-**microchange** of C is computed as follows (compare fig. 3.12, where G_{kl} is G_{22} from fig. 3.7):

Step 1: Select a derivation *assoc* = $x_0 r_1 x_1 ... x_{n-1} r_n x_n$ in C which yields a G_{kl}-association $r_1 ... r_n$.

In fig. 3.12a, 5`rage5top`1 is selected. This is a G_{22}-association since `rage top` subsumes the $C_{G_{22}}$-word `rage scream`. The transitions concerned are marked by bold arrows in fig. 3.12a.

Comment: *assoc* can be selected at random. The selected association need not have maximal length. However, in the comments below I argue that self-organiziation is fostered by preferring "informative" associations for selection.

Step 2: Compute the direct continuations of *assoc* in C, i.e., compute the set *cont* of transitions that continuate *assoc*:

$$cont = \{x r x' \in trans \mid x = x_n\}$$

In fig. 3.12a, *cont* = {1time2, 1love3, 1human3, 1man4} (dotted and broken arrows).

Step 3: Split *cont* into $cont_1$ and $cont_2$, where $cont_1$ contains those transitions which make G_{kl}-compatible continuations of *assoc*, (i.e., *assoc* continuated by these transitions yields again a G_{kl}-association), and where $cont_2$ contains the G_{kl}-incompatible continuations. When $cont_2$ is empty, then stop. The microchange, in this case, is an "empty change", which changes nothing.

In fig. 3.12a, $cont_1$ = {1human3, 1man4} (broken arrows) and $cont_2$ = {1time2, 1love3} (dotted arrows). I.e., 5rage5top1human3 and 5rage5top1man4 yield G_{22}-associations rage top human and rage top man, whereas 5rage5top1time2 and 5rage5top1love3 yield rage top time and rage top love, which are no G_{22}-associations.

Comment: In subsequent steps of the algorithm, the transitions contained in $cont_1$ will be "moved" along together with *assoc*, as indicated in fig. 3.11. The transitions from $cont_2$ yield the "railways" for the move. The association cannot move along a transition that would itself be suitable as a continuation of the association. Once an association "locks" to continuations, they cannot be cut off by moving the association. However, they can be cut off by *other* associations that pass by in another microchange, and which *compete* for these continuations.

Step 4: Put

$$S_1 := \{x \in S \mid \text{there exists some } x_n r x \text{ in } cont_1\}, \text{ and}$$
$$S_2 := \{y \in S \mid \text{there exists some } x_n r y \text{ in } cont_2\}.$$

Modify *trans* to make *trans*':

$$\begin{aligned} trans' := trans \; &- \{x_{n-1} r_n x_n\} \\ &- cont_1 \\ &\cup \{x_{n-1} r_n y \mid y \in S_2\} \\ &\cup \{y r x \mid y \in S_2, x_n r x \in cont_1\} \end{aligned}$$

In the example of fig. 3.12, S_1 = {3, 4} and S_2 = {2, 3}. The effects of computing *trans*' are shown in fig. 3.12b.

Comment: In intuitive terms, *trans*' is computed from *trans* by "moving" the terminal end of *assoc* along $cont_2$ (*assoc* "disperses" during the move when $cont_2$ contains more than one element, since then the former single transition $x_{n-1} r_n x_n$ is replaced by several $x_{n-1} r_n y$). The G_{kl}-compatible continuations of *assoc* are carried along with the terminal end of *assoc* (thus, they also may disperse).

Step 5: Select all transitions which directly continuate the moved *assoc* and which yield G_{kl}-associations with *assoc*. Compute for the dynamic symbols *r* occuring in these transitions the G_{kl}-specializations in the context of $r_1 ... r_n$, or their Σ_l-abstractions.

More precisely, consider the set $cont^+ := \{yrz \in trans' \mid y \in S_2, r_1 ... r_n r \text{ is a } G_{kl}\text{-association}\}$. All of these transitions are one-step continuations of *assoc* (after the move of the latter). For each of the *r* ocurring in transitions from $cont^+$, compute *r*' as follows:

If $r > s$ for some $s \in \Sigma_l$, compute *r*' as the G_{kl}-specialization of *r* in the context $r_1 ... r_n$. In other cases, put $r' := r^{(l)}$.

Now compute

$$trans'' := trans' - cont^{+} \cup \{yr'z \mid yrz \in cont^{+}\}$$

and change the configuration accordingly. Intuitively, step 5 adapts the abstraction level of concepts which come into the contextual scope of the moved association.

Comment: In this step, *assoc* exerts some influence on the local environment where its "head" has arrived after the move. This influence afflicts the level of abstraction of dynamic symbols in its vicinity. They are specialized insofar as a specialization is uniquely determined by *assoc* (in terms of yielding continuations of *assoc*); they are abstracted when their original level of abstraction is more special than the one of the underlying global state G_{kl}.

Step 6: If in *trans*" there are no transitions left that lead into x_n, then delete x_n and the transitions that lead away from it. The result will be a cyclic generator.

Fig.3.12c depicts the result of steps 5 and 6. The continuations `top` and `human` have become specialized to `man` at several places. No abstraction of dynamic symbols occurs in the example. The local state 1 is deleted together with its attached transitions.

Comment: The move of *assoc* may destroy the cyclicity of C. The local state x_n, when it has no transitions leading to it after the move, has become "inaccessible": no further move of another association in another microchange can arrive at x_n. Step 6 restores cyclicity. As a side effect, this prevents C from breaking apart into disjoint substructures.

Step 7: "Renormalize" the generator, as constructed so far, to a phase generator C' (when there is no "dispersion", a recomputation will usually be unnecessary). More precisely, if C^{+} is the generator constructed in steps 1-6, then compute the phase generator of $C_{C^{+}}$. C' is the result of the microchange.

A recomputation is necessary in the example. The result is shown in fig. 3.12d.

Comment: The reason for insisting on phase generators will be discussed in detail further below.

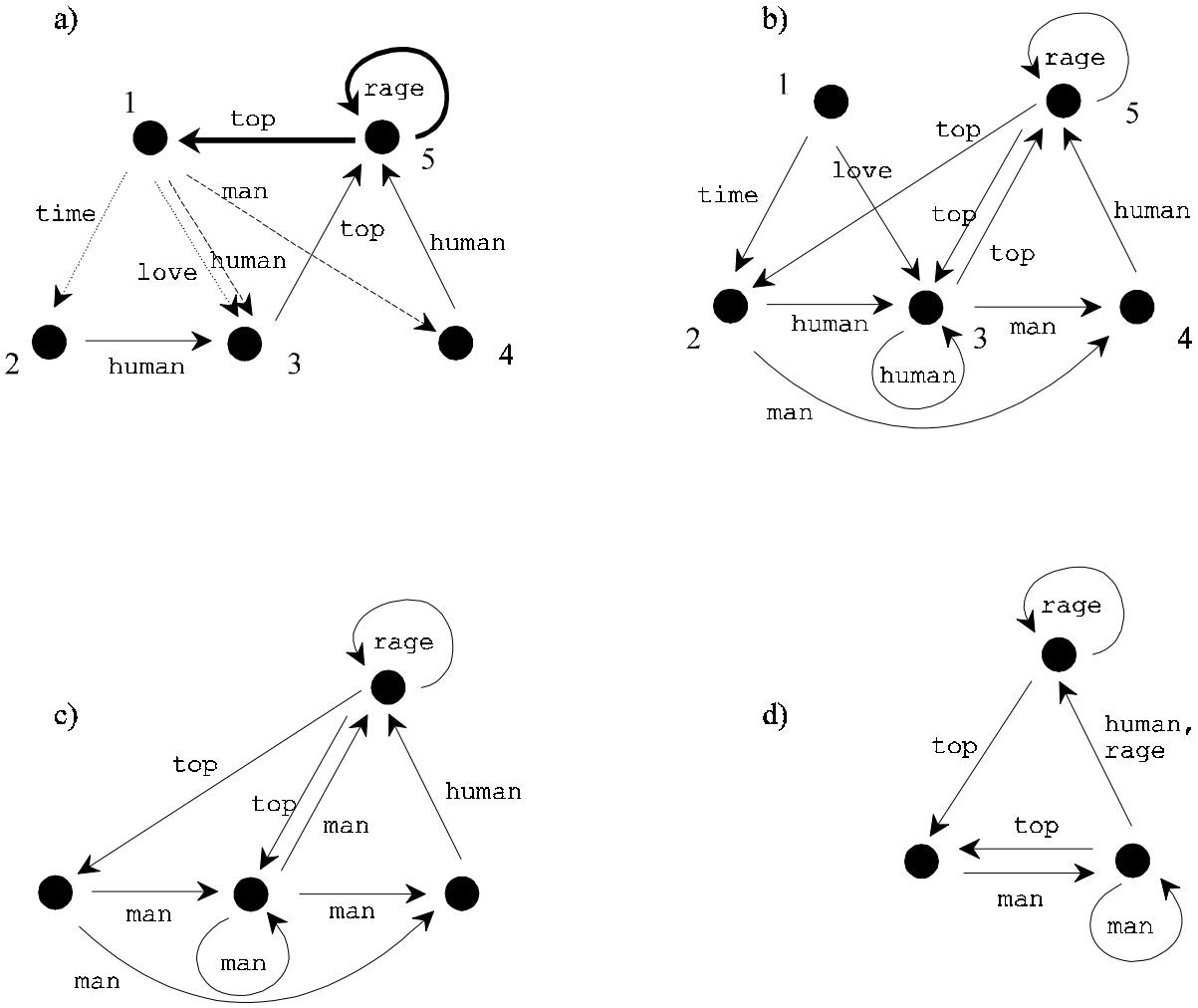


Fig. 3.12: The execution of a microchange according to algorithm 41.

Definition 42:

(i) A **self-organizing scene** (more specifically, a self-organizing (G_{ij})-scene) is a finite or infinite, alternating sequence C_0, $G_{i_0j_0}$, C_1, $G_{i_1j_1}$, C_2, ... of Σ-configurations and global states from (G_{ij}), which starts with a Σ-configuration and (in the finite case) ends with a Σ-configuration, and where C_{n+1} is derived from C_n by a $G_{i_nj_n}$-microchange.

(ii) If C_0, $G_{i_0j_0}$, C_1, $G_{i_1j_1}$, C_2, ... is a self-organizing scene, then the sequence $(C_i)_{i=1, 2, ...}$ is its **history**. A subsequence of a history, in which the global state is constant ≡ G_{kl}, is a G_{kl}-**passage**.

The intentions behind the DSS approach suggest that the global state remain fixed for comparatively long intervals, i.e., that a history consists of relatively long G_{kl}-passages, and that changes in global states are "smooth", i.e., that when a global state shift occurs at some point in the history, then it is a minimal alteration from G_{kl} to $G_{k\pm1l}$ or $G_{kl\pm1}$. In other words, macrodynamics should be slow in comparison with microdynamics. In cognitivistic terms, a global mental state, like the degree of tiredness, has a slower dynamics than the dynamics of reasoning itself; in thermodynamic terms, the global thermodynamic state of a system changes slowly, compared to local interaction timing on the microlevel.

The remainder of this subsection is devoted to a discussion of several points of interest concerning microchanges and self-organizing scenes. References to "steps" mean the steps in algorithm 41.

A closer look at microchanges

The basic idea of microdynamics is to enable associations to wander about in a self-organizing scene, and, by meeting with suitable interaction partners, connecting into longer associations, or competing for sub-associations. Microchanges lead to such a behavior. But the specific details of algorithm are by no means cogent. There are many "knobs to turn" in the algorithm. Some variations will have considerable impact on ergodic phenomena in self organizing scenes. I hint at four variations of microchanges, which are of particular interest. Let them be called type 1 - 4 microchanges, with the original algorithm yielding type-0-microchanges.

Type 1: In step 1, do not select associations at random but prefer informative associations, i.e., associations with a high H_{kl} value according to definition 37(vi).

"Preferring" more informative associations does not imply that one has to compute H_{kl} for all associations in a configuration in order to select the most informative one. One of the many alternatives to this catch-all approach is to volunteer a small random collection of associations for the computation of H_{kl}, and then select the most informative of these candidates.

Preferring informative associations increases the average informativeness of associations in long-term runs of self-organizing scenes. In cognitive terms, more informative associations should be interpreted as being more salient. Preferring them means that salient parts in a self-organizing scene are microdynamically more active than parts that are poor in information.

Type 2: Type-0-microchanges have an undesirable effect. Associations that "belong together", in that they lead into a common local state and have there a common continuation, can become separated. Fig. 3.13a,b shows how the (length 1) association `Adam` becomes separated from `Eve` due to a type-0-microchange.

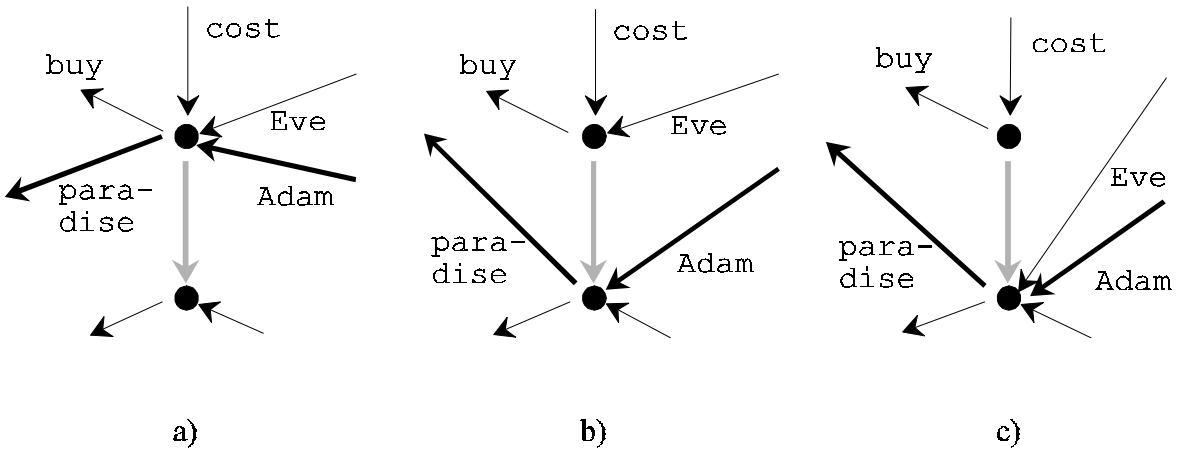


Fig. 3.13: An undesired side effect of a G_{kl}-microchange (from a to b) and its remedy (from a to c). The coherency from fig. 3.5 is taken for G_{kl} in this example.

Figure 3.13 also indicates how this counterintuitive side effect can be mended by a variation of step 4. The basic task is to define which other transitions $yr'x_n$ leading into x_n, besides $x_{n-1}r_nx_n$, have a "related meaning" to *assoc* = $x_0r_1x_1...x_{n-1}r_nx_n$ such that they should move together with $x_{n-1}r_nx_n$. A natural approach is to use the G_{kl}-meaning of $s_1...s_n$, as defined in 37(v). More specifically, define r' to have a **related meaning** to $r_1...r_n$:iff $\Phi_{kl}(r') \subseteq \Phi_{kl}(r_1...r_n)$ or $\Phi_{kl}(r') \supseteq \Phi_{kl}(r_1...r_n)$. Note that it would not suffice to require that r' and $r_1...r_n$ possess a common continuation, as witnessed by the counterexample r' = `paradise`, $r_1...r_n = r_1$ = `buy`. They share the continuation `apple` but should not be considered as having a related meaning.

With a notion of related meaning available, it is easy to adjust the definition of a microchange, as indicated in fig. 3.13c.

Type 3: In step 4 (i.e., the "move" step), do not delete from *trans* all the transitions $\{x_{n-1}r_nx_n\} \cup cont_1$ that are deleted in the original version of this step. I.e., select a subset $trans^* \subseteq \{x_{n-1}r_nx_n\} \cup cont_1$ and define

$$\begin{aligned} trans' := trans \;-\; trans^* \\ \cup \{x_{n-1}r_n y \mid y \in S_2\} \\ \cup \{yrs \mid y \in S_2,\ x_n r x \in cont_1\} \end{aligned}$$

The net effect of this variant is less competition between associations, since a "moving" association can now *copy* parts of other associations instead of *cutting* them off. This fosters the parallel development of associations that share subsequences, which might be helpful when an exploratory type of processing is desired. Too little cutting, however, will be harmful. In the extreme, when there is no cutting at all, a G_{kl}-passage is likely to degenerate to a stationary, trivial single-phase configuration which looks like G_{02} in fig. 3.7. This happens because the passage converges to the language $(\Sigma_{C_0})^*$, where Σ_{C_0} is the set of dynamic symbols that are present in C_0. The phase generator of this language is the trivial single-phase generator.

Type 4: Use only a subset of S_2 in step 4. This reduces the degree of "spreading" of associations. In the extreme, use only one element of S_2 for step 4, thus suppressing spreading altogether. An intuitive metaphor for this alternative is to think of step 4 as "wave spreading" (in type-0-microchanges) vs. "particle movement" (in the extreme case of type 4).

These variations (and others that remain to be invented) can be dynamically tuned during a history: the first, by changing the stringency of preference, the second, by restraining or relaxing the notion of semantic relatedness, the third, by adapting the percentage of transitions that become deleted, the fourth, by adapting the percentage of S_2 used for moving an association. Such tuning offers a great amount of flexibility in the control of self-organizing scenes - and makes their management a subtle affair. This should not come as a surprise. Complex, collective processes are intrinsically difficult to guide (compare Forrest & Miller 1990 for analog remarks concerning classifier systems).

Resonances

It may happen that in the course of a G_{kl}-passage, an association meets with itself and finds itself a continuation of itself. A cyclic association forms. More generally, cyclic substructures in a configuration, which are generators of a sublanguage of $C_{G_{kl}}$, may occur. Such substructures are a conspicuous and important phenomenon. Definition 43 specifies such *resonances*:

Definition 43: A cyclic substructure R of a configuration is a G_{kl}-**resonance** :iff C_R contains only G_{kl}-associations.

I have chosen the term "resonance" for its obvious associations with physical phenomena. Carpenter and Grossberg (1990) use the same term in a related sense. Their "adaptive resonance theory" (ART) describes recurrent, distributed neural network architectures for real-time pattern recognition. The input can consist in single patterns, or sequences thereof. When a pattern is found to match with a stored goal pattern, the system turns into a "resonant" state, where activation can persist even after the pattern is removed from the input field. Thus, input-free intervals can be bridged, and the interpretation of successive input is influenced. In DSS, resonances likewise allow a self-organizing stream to decouple its activity from input timing (cf. 3.4).

An obvious precondition for resonances to be formed in a history is that dynamic symbols in C_0, which can become interconnected in "cyclic" associations, exist. That taken as granted, the likeliness of G_{kl}-resonance generation in a G_{kl}-passage and their stability, further depend on many factors. It is promoted, for instance, by the following tendencies:

- There is a low degree of competition between associations. I.e., when a subset of all available dynamic symbols can form several associations, then these associations are prone to interconnect instead of competing destructively. As an example for "co-operative" dynamic symbols, take {`apple`, `paradise`, `Adam`, `Eve`}; as an example for destructive competition, take {`apple`, `Eve`, `paradise`, `buy`, `cost`} (assuming the example from fig. 3.5 for G_{kl}). In the latter set, `Eve` and `paradise` compete with `buy` and `cost` for `apple`.
- The configurations concerned are small. The chance of an association's closing back in itself is then increased for purely combinatorial reasons.
- Informative associations are preferred for microchanges (i.e., type 1 microchanges). Informativeness correlates, to a certain degree, with association length. Long associations have better chances of crossing themselves than short ones have.
- Associations with related meaning are coupled in a move (i.e., type 2 microchanges). This fosters association growth by reducing competition and thus, again, increases the combinatorial chances of associations to loop into themselves.
- There is much copying and little cutting (i.e., type 3 microchanges). This again promotes association growth.

A resonance formation event typically induces a sudden increase in the information value H_{kl} of the associations that are contained in the resonance. When an association $r_1...r_x$ loops back into itself, the initial sub-association r_1 will abruptly become equipped with a (cyclically infinite) context $...r_{x-1}r_x$. Other elements of $r_1...r_x$ will likewise gain such an infinite context where they had only a finite one before. These additional contexts yield an increase in information. This increase renders resonances particularly interesting when there is a preference for informative associations (type 1 microchanges). Then, resonant associations (i.e., associations occurring in a resonance) will be preferred over non-resonant associations in

the selection for microchanges. Since an association selected for a microchange cannot deteriorate in the microchange (it can only grow), a resonance will be markedly more stable than the associations partaking in it had been before the resonance formed; microchanges will favor the resonance's accreting further continuations over losing them to competing associations. In complex systems terminology, a resonance acts as an attractor. In cognitive terms, it is the DSS model of gestalt formation.

It may occur as a special case that at a point in the history, the entire configuration is a resonance. Then, nothing more will happen, since within such a total resonance the set S_2 (step 3) is always empty, i.e., microchanges do not effect any further change, i.e., the history is stationary. To put this the other way round, effective change can occur only as long as the language generated by a configuration is not a proper sublanguge of the "goal language" $C_{G_{kl}}$. The temporal development is, in this sense, conflict-driven.

Self-organization I: fast equilibration

The cumulative effect of microchanges is hard to predict analytically. This situation is characteristic for collective computation. Computer simulations are indispensable for a deeper understanding of self-organizing scenes. Under the simplifying assumption that the global state G_{kl} remains constant throughout the history, some theoretical prognoses are, however, on the safe side:

- The abstraction level of dynamic symbols is likely to converge towards Σ_l, when there are enough concepts apt to form associations. This is mostly due to step 5 in the microchange algorithm, which drives the abstraction level of association continuations toward Σ_l. Another effect to the same end is that "singular" dynamic symbols, which do not associate with others (and are therefore not susceptible to step 5), are prone to be deleted in step 6.
- When there exists a set Σ_{res} of dynamic symbols in C_0, from which it is combinatorially possible to form a G_{kl}-resonance R, and whose elements are not competed for by other dynamic symbols in C_0, then a resonance made from the dynamic symbols from Σ_{res} is likely to form. Its formation would be almost certain, were it not for the potential deletion of local states in step 6. The outcome of the race between deletion and spreading/copying is impredictable. It can, however, be influenced by tuning microchanges as outlined above. Once a resonance is established, without there being competing associations outside it, it is immune to the deletion of local states in step 6.
- A pathological case: When C_0 contains only disparate dynamic symbols, which cannot form G_{kl}-associations of length greater than 1, the history is likely to degenerate into a stationary, trivial single-phase configuration $C_n = C_{n+1} = C_{n+2} ... = (\{x\}, \{xr_1x, ..., xr_kx\})$, where $\{r_1, ..., r_k\}$ is a subset of the dynamic symbols occuring in C_0. The reason is that the languages generated by the configurations in the history will converge to some $(\Sigma_{survive})^*$, where $\Sigma_{survive}$ is a subset of the dynamic symbols originally present in C_0.
- When C_0 contains mainly `top`-transitions, then either C_0 is already a resonance in its entirety, or it will very likely soon become so. As a consequence, the history becomes stationary.

- More generally, a possible stable outcome of a G_{kl}-passage is the generation of several mutually non-competitive resonances (assuming type 2+4 microchanges). These resonances move along (or rather, "through") each other without affecting each other. Furthermore, each of these resonances is likely to acquire a maximal degree of internal connectivity.

The effects mentioned in this list are *fast*. To understand what this means, one has to examine the notion of time in self-organizing scenes more closely.

Microchanges can, to some extent, be computed in parallel at different places in an self-organizing scene. At least this holds when they are truly local, i.e., when the renormalization in step 7 induces only a local change in the vicinity of the moved association, which will often be the case. Therefore, when C_0, C_1, ..., C_n is a G_{kl}-passage, there are typically many other routes C_0, C_1', C_2', ..., C_{n-1}', C_n, which also lead from C_0 to C_n, and which are, therefore, essentially equivalent to the first development. The moral is that a sequence of configurations is only a technical contrivance to manage temporal development. It should not be misunderstood as reflecting time proper. As a first approximation, one might define a subsequence of arbitrarily permutable microchanges as a global system time step. The issue of time will be discussed at greater detail below. At this point, it suffices to observe that a natural measure of "system time" should be considerably coarser than "configuration sequence time".

For their full expression, the self-organization effects mentioned above apparently need system time steps in the order of magnitude of the number of local states in a self-organizing scene. In more suggestive terms, a G_{kl}-passage equilibrates in many respects during the interval of time it takes for local effects to spread once over the width of configurations.

Such equilibration effects can be compared in their rapidity, e.g., with the development of structure in self-organizing cellular automata (Wolfram 1984), with equilibration in many kinds of neural networks in a single pattern recognition episode, with the equilibration of pressure gradients in a reactor containing a gas, or with the interpretation of a conflict-free sentence in humans. These fast processes have to be contrasted with long-term structure-forming processes as, e.g., genetic adaptation of classifier systems, learning in neural networks, biochemical adaptation of a cell to the chemistry of its environment, or long-term memory learning in humans.

Unfortunately, the above considerations are essentially only plausibility considerations. This situation can only be mended by extensive computer simulations, which yield statistical support, or by a far-reaching mathematical analysis with methods from probability theory. Both approaches are beyond what I can offer in this thesis.

Self-organization II: effects on a medium time scale

Equilibration leads to a rapid stabilization of only some characteristics of a G_{kl}-passage. Other parameters settle to stability only in extended passages, or might even continue to fluctuate indefinitely. Again it is difficult to predict what will happen without the experience of simulations, but the following points are safe:

- The set of dynamic symbols that appears in a G_{kl}-passage after an initial equilibration towards Σ_l can only get smaller in an extended G_{kl}-passage. This is obvious, since in step 6 dynamic symbols can vanish from the scene, but there is no input mechanism for the introduction of new dynamic symbols.
- Interesting medium-term activity can be expected when several, competing resonances form. Then, there exists no clear winner attractor state for the self-organizing scene. The resonances can persist potentially for an indefinite time, growing and shrinking due to cutting portions out of each other. In the long run, however, almost certainly a population of non-competing resonances "survives".

Cognitive candidates for such medium-range phenomena are conflict-loaden processes, e.g., the interpretation of garden path sentences, or ambiguous pattern recognition tasks.

A crucial parameter is the very size of configurations. Uncertainties arise from the renormalization to phase generators in step 7. The reasons for step 7 will be discussed below. A microchange can lead both to an increase and to a diminuation in the size of the phase generator. In case there is a persistent net growth of configurations, the growth velocity is crucial for the likeliness of resonance formation. If it is high compared with the migration velocity of associations, the latter will persistently be "stretched" and thereby hampered in looping back into themselves.

Size development is affected by all of the variations of microchanges mentioned above, and by the choice of G_{kl}. The latter is particularly effective for shrinking configurations: in the extreme, shifting l to 0, i.e., abstracting dynamic symbols to `top`, rapidly yields trivial single-phase configurations.

In many cases, a self-organizing scene will resolve initial competition conflicts by deletion and resonance formation and become essentially uninteresting, if not entirely stationary. Global state shifts may then become relevant as a means for reviving activity. However, in order to arrive at interesting long-term developments, it seems more natural to allow external input (cf. 3.4).

Why phase generators?

There are several reasons for the renormalization step 7. One is that the renormalization is apt to simplify the generator constructed in steps 1-6, as is indeed the case in the example of fig. 3.12.

More subtle, and much more important, reasons concern the question of what exactly, in fact, is modeled in a self-organizing scene. It would be suggestive to interpret a self-organizing scene (or stream) as a model of the physical realization of an information processing module. The self-organizing scene's structure would, then, mirror physical structure. E.g., dynamic symbols that occur at different transitions could be interpreted to correspond to local neural activation patterns, and the topological structure of a configuration could be interpreted to reflect the geometrical distribution of these patterns on a cortical surface.

However, this interpretation is *not* intended. There are better models of physical information processing structures than configurations - to wit, connectionist models.

A self-organizing scene *is* intended, much more abstractly, to model the state of a self-organizing information processing module in purely informational terms. A self-organizing scene's only property of interest are the words and associations derivable in it, i.e., the languages generated by its configurations. These words are the only observables that lie in the focus of the DSS approach, and of which a formal account is attempted. In this sense, all other generators that yield the same language as some given C_i in a history are equivalent, since they state the same about the observables.

But, different generators can lead to incommensurable developments of a self-organizing scene. When C_i and C_i' are two equivalent configurations (i.e., $C_{C_i} = C_{C_i'}$), and some microchange leads from C_i to C_{i+1}, it can happen that there is no microchange which can lead from C_i' to a generator equivalent to C_{i+1}. Structural differences between equivalent C_i, C_i' can thus act as "hidden variables". The reason for using phase generators, then, is to resort to the "purest" representation format possible for the technical handling of the informational content of a self-organizing scene. Phase generators represent, in their very structure, nothing but what is essential for the interaction of associations, namely, contextual influences. There is no contingent additional structural information in a phase generator that might induce artifacts to bear on the development of languages in a history.

Another, equally fundamental reason for phase generators is that they provide a principled answer to the question of what constitutes an "instance" of an informational entity. On the conceptual level, for instance, the question arises of how to model one's ability to think of two distinct apples simultaneously. In classical symbolic representation formalisms, this presents no difficulties: simply introduce two instance variables `apple_1` and `apple_2`, assert `apple_1` $\neq$ `apple_2`, and let model theory take care of making sure that there are, in fact, two apples. In DSS, there is no difference between individuals and classes. Indeed, this distinction does not make any sense here; there are neither individuals nor classes. The theory is all about observables; both `Eve` and `apple` are observables, namely, activation patterns in a brain (for instance); and there is nothing more to be said about it. Then, how can an agent think of two distinct apples?

The answer is that two apples occur in one's mind if and only if the two are distinguished informationally. In DSS terms, two apples are indeed two `apple`s when the latter occur in separable contexts and give rise to separable continuations. This can be conveniently expressed using phases:

Definition 44: An **occurence** of a dynamic symbol r in a coherency G is a phase φ of C_G such that there exists some transition $\varphi' r \varphi$ in G_φ.

Thus, there are as many distinct apples in a configuration as there are phases which can be reached by an `apple`-transition. This is the other reason for phase generators.

An intriguing side effect of the renormalization to phase generators in step 7 is that sometimes a large and complex self-organizing configuration can "collapse" into a much simpler one. This phenomenon is accompanied by a merge of dynamic symbol occurences. Intuitively, this models an "Aha!" effect, where disparate notions suddenly unify.

For the same fundamental reasons as the ones just mentioned, one should require that the generators G_{ij} used in a dynamic symbol space be phase generators. The only reason for not doing so is the imperfect present state of the formalism. In order to use phase generators in (G_{ij}), the abstraction and symmetrification operations have to be tuned to the special restrictions of phase generators. I have started to work in this direction, but it appears that this requires a considerably deeper understanding of coherent languages, and morphisms between generators, than is presently within my reach.

Time and structure

I have already mentioned that the succession of configurations in a history is a technical contrivance rather than a natural model of time. Arriving at a satisfying account of time is difficult. Several disparate observations and requirements must be reconciled, among them the following:

- It is desirable to interpret time not as an extraneous parameter, which orders and measures change, but as change itself. Such a *qualitative*, non-parametric account of time is readily provided by discrete dynamic systems like cellular automata or finite automata (and by self-organizing scenes, too, but not as convincingly). Here, time steps are identical to qualitative changes.
- It is, however, also desirable to have a *quantitive* measure for time. One would like to be able to state that some process is slower than another. The common method to make this possible is to use a parametric time as a universal reference. An alternative is to define systems of interacting subsystems, each of which comes with its own qualitative time. These subsystems could then provide relative time measures for each other. How this idea becomes effective for DSS will be indicated in 3.4.
- In many physical systems, as well as in self-organizing scences, change is caused by *local* mechanisms. It is unnatural to assume a global clock for the coordination of parallel local changes. Technically, this leads to some of the well-known difficulties in the coordination of parallel processes. On a theoretical level, this leads to *relativistic* notions of time.
- One will often find a *duality* between time and structure. When a particular association is traced through a history, it can be observed how it grows at its "head" and dissolves from "behind". An association, thus, can be considered a transient trace of an ongoing process. In this perspective, a configuration is a blurry image of an ongoing development rather than a sharp standstill picture. The directedness of a transition can be interpreted temporally and structurally. It is temporal insofar as it reflects the direction of growth and dissolution of associations (incidentally, Sandewall (1993) arrives at a strikingly similar picture of head-growing/tail-dissolving memory elements in a logic-oriented description of situated reasoning). A transition is passively structural in that it yields (oriented) "railways" for moves of other associations (cf. the dotted arrows in fig. 3.12). But then, again, these "railways" also have a temporal aspect: when an association moves along it, then the association is *earlier* at the beginning of the transition than at its end.
- A more global dualistic aspect of associations is that they are interlinked with each other, thereby forming a grid, which serves as a frame of reference for the coordination of interaction between associations. The cognitive phenomenon thus captured is that of a relatively stable mental image. For instance, when one reasons about a task with a given goal, the goal is present in mind as a relatively persistent image, while considerations about

how to achieve the goal form an ongoing process. Note, the stability of such images is not absolute. They are made from the same mental stuff as the ongoing considerations themselves. The roles of structure vs. process are prone to become exchanged; mental images are "frozen processes" which can unfreeze (the term is borrowed from Ipke Wachsmuth). This becomes particularly apparent in informational structures that are intrinsically temporal, as melodies or rhythms. Lashley (1951) lists many other examples from all levels of the periphery-centre hierarchy. He claims that the question of how "spatial" representations are transformed into "serial" ones (and vice versa) lies at the bottom of understanding the brain. The procedural vs. declarative knowledge debate in AI, which has exhausted itself rather than having resolved the question, is a more recent witness of the problem's impact. When massively parallel processing of symbolic information gains in influence, the question will predictably resurface.

The basic idea of a dynamics that locally changes the topology of a graph is partially inspired by Zuse's (1975) proposal to investigate "net automata". Net automata are a generalization of cellular automata, where the topology of the cell grid is modified by local rules. Zuse views net automata as a contribution to theoretical physics, aiming at "computational" models of space-time at a very small scale. He examines from various angles how such a conception of space-time might correlate with insights from relativity theory and quantum mechanics. Though Zuse's aims can hardly be compared with the intentions underlying DSS, it is interesting to note that in that case, too, a topology-changing dynamics directly leads into fundamental questions concerning time and structure.

This is all more enigmatic than one would like. An integration of these issues would be a great achievement. It is not surprising that I did not yet find a way to accomplish this task. No presently available theory of complex systems - physical, abstract mathematical, computational, or psychological - offers a comprehensive account of local interactions vs. global time, qualitative vs. parametric time, different time scales, or time vs. structure. Before this is done, our understanding of complex processes in general, and of self-organizing information processing in particular, remains incomplete. DSS might be an interesting starting point for further investigations in this area, since one of its basic constructs, namely, associations, has dual temporal/structural characteristics.

Self-organizing streams

I shall now describe how self-organizing scenes can be generalized to account for input and output, arriving at *self-organizing streams.*

Equipping self-organizing scenes with I/O-facilities could be done ad hoc by defining an *open configuration,* which has transitions leading out of it, and into it (fig. 3.14a). Input would be effected by appending new transitions to the outgoing transitions, output by deleting or copying ingoing transitions (fig. 3.14b).

Note that *in*put works via *out*going transitions since this is the continuation direction, i.e., the direction in which words are derived. This needs some habituation, since one is accustomed to represent input into a system graphically by arrows directed into a system, rather than by arrows pointing outward. However, the unusual direction is quite correct. The directed

transitions in DSS generators can be interpreted temporally (cf. 3.3). Traversing a path of transitions can be considered as going with time. In this view, the *c'*-transition in fig. 3.14b comes *later* than the *c*-transition. This correctly reflects the fact that *c'* is inputted later than *c*.

The problem with the simple picture of fig. 3.14b is that configurations should be phase generators. But, a phase generator cannot be "open", since it is by definition a cyclic structure. This difficulty can be overcome by introducing a special dynamic symbol ex (from *ex*ternal, *ex*terior), which does not belong to the underlying dynamic symbol space, and fully cross-connecting the terminal local states of the open configuration with the initial local states by ex-transitions (fig. 3.14c). The resulting generator is cyclic and can be transformed to a phase generator. In this phase generator, a unique ex-transition appears (fig. 3.14d), which serves as the generator's I/O-port. The step from Fig. 3.14d to 3.14e replays the I/O-events from fig. 3.14b in the phase generator.

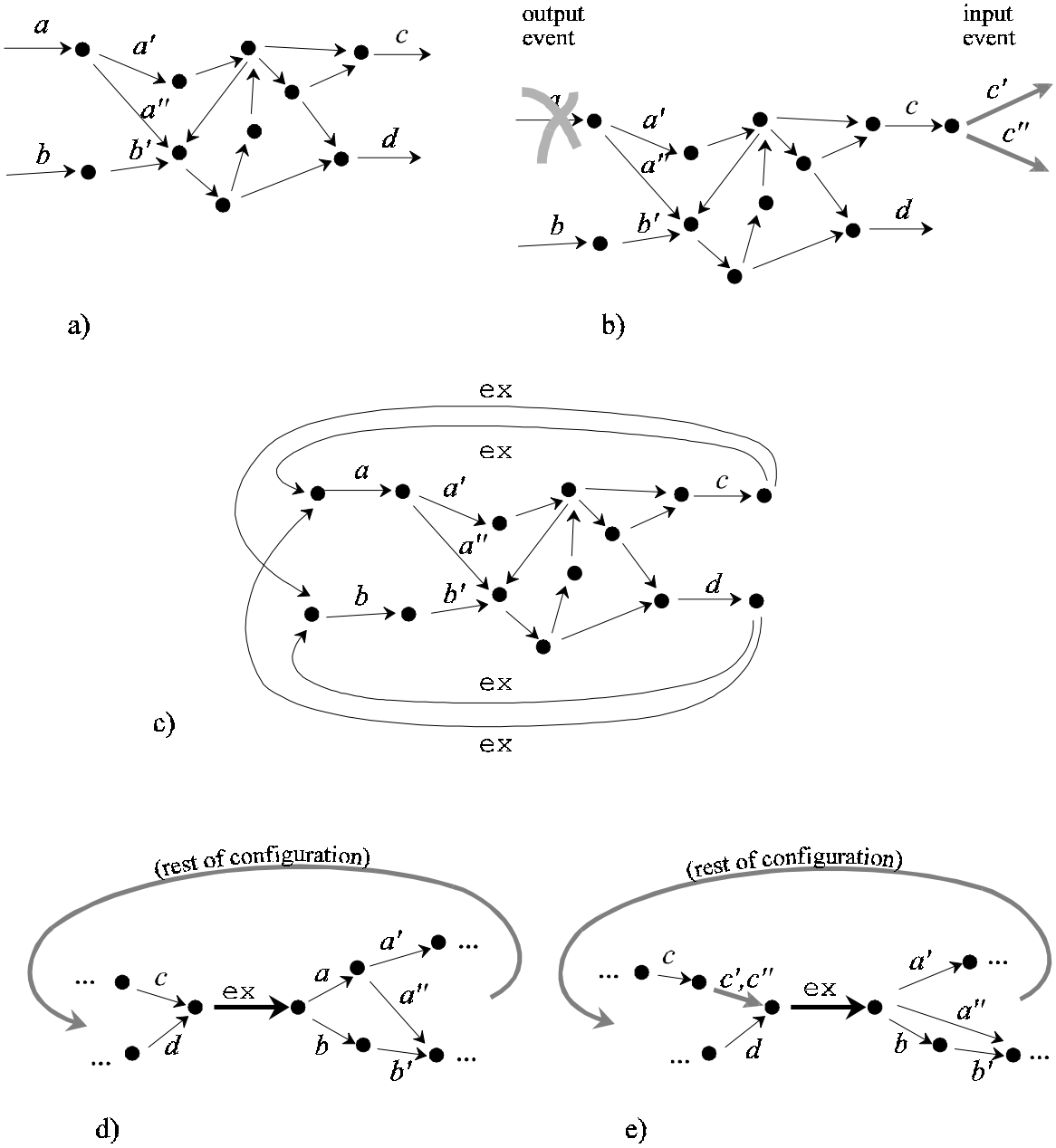


Fig. 3.14: How to construct an open configuration while leaving it formally closed.

I call ex a "dynamic symbol" for the sake of terminological simplicity, although it seems questionable whether one should consider the "exterior" of an information processing module as an observable.

For the remainder of this subsection, let $(G_{ij})_{i=1,...,m\ j=1,...n}$, where $G_{ij} = (S_j, trans_{ij})$, be a dynamic symbol space with a Σ-abstraction tree $((\Sigma_j)_{j=n,...,0}, (\alpha_j)_{j=n,...,1})$.

The formal definition of an open configuration should now cause no difficulties:

Definition 45: Let C = (S, *trans*) be a phase generator with dynamic symbols from $\Sigma \cup \{ex\}$, where $ex \notin \Sigma$. Then C is an **open Σ-configuration** :iff (i) there exists exactly one transition $(x, ex, x') \in trans$ that is labeled by ex, and (ii) this is the only transition between x and x', and (iii) $x \neq x'$. More precisely: C is an open Σ-configuration :iff (i) $\forall$ (x, ex, x'), (y, ex, y') $\in$ *trans*: x = y and x' = y', (ii) $\forall$ (x, *r*, x'), (x, *s*, x') $\in$ *trans*: $r = ex \rightarrow s = ex$, (iii) $\forall$ (x, ex, x') $\in$ *trans*: $x \neq x'$.

The definition of a microchange must be slightly adjusted in order to prevent moves which transgress the "exterior" represented by ex.

Algorithm 46: A G_{kl}-microchange for an open configuration is defined as in algorithm 41, with the following modification of step 2:

Step 2': Compute the direct non-ex continuations of *assoc* in C, i.e. compute the set $cont = \{xrx' \in trans \mid x = x_n, r \neq ex\}$ of transitions that continuate *assoc* and that are not the ex-transition.

For open configurations, microdynamics is augmented by local input and output operations:

Algorithm 47: Let C = (S, *trans*) be an open configuration, and xexx' the ex-transition. Let $input = \{r_1, ..., r_n\} \subseteq \Sigma$. Select some set $\{y_1r_1x, ..., y_mr_mx\} \subseteq trans$. Introduce a new local state y'. Put $intrans := \{y'r_ix \mid r_i \in input\}$. Put

$$trans' := (trans - \{y_1r_1x, ..., y_mr_mx\}) \cup \{y_1r_1y', ..., y_mr_my'\} \cup intrans.$$

Compute the phase generator C' that corresponds to the generator $(S \cup \{y'\}, trans')$. The entire operation leading from C to C' is a **micro-input**. The set *input* is the **input** of the micro-input.

Output events are a variety of microchanges. Adapt algorithm 41 as follows:

Algorithm 48:

Step 1: Instead of *assoc*, use the transition xexx' in the remainder of the algorithm. *Step 2* remains unchanged. *Step 3*: Instead of splitting *cont* into $cont = cont_1 \cup cont_2$, split it arbitrarily into two nonempty subsets $cont = cont_{readout} \cup cont_{keep}$. Use $cont_{readout}$ instead of $cont_1$, and $cont_{keep}$ instead of $cont_2$ in the rest of the algorithm. The other steps remain unchanged with the exception of step 5, which is ignored.

The entire operation leading from C to C' in this variant of algorithm 41 is a **micro-output**. The set

$$output := \{r \in \Sigma \mid x'ry \in cont_{readout}\}$$

is the **output** of the micro-output.

It is straightforward to show that all microdynamic operations leave the unique existence of the ex-transition intact:

Proposition 49: Microchanges, micro-inputs and micro-outputs transform open configurations into open configurations. □

A self-organizing stream is defined in perfect analogy to self-organizing scenes:

Definition 50:

(i) A **self-organizing stream** (more specifically, a self-organizing (G_{ij})-stream) is a a finite or infinite, alternating sequence C_0, $G_{i_0j_0}$, C_1, $G_{i_1j_1}$, C_2, ... of open Σ-configurations and global states from (G_{ij}), which starts with an open Σ-configuration and (in the finite case) ends with an open Σ-configuration, and where C_{n+1} is derived from C_n by a $G_{i_nj_n}$-microchange, a micro-input, or a micro-output.

(ii) If C_0, $G_{i_0j_0}$, C_1, $G_{i_1j_1}$, C_2, ... is a self-organizing stream, then the sequence $(C_i)_{i=1, 2, ...}$ is its **history**. A subsequence of a history, in which the global state is constant $\equiv G_{kl}$, is a G_{kl}-**passage**.

The remainder of this subsection is devoted to a discussion of several topics of interest.

A closer look at input and output

Input can come from sources of different nature. The selection step should be adapted to the particulars of the situation. I treat two different cases without going much into detail.

First, input can come in singular events "out of the blue"; different micro-inputs don't have much to do with each other at first sight. This is characteristic, e.g., for early stages of difficult diagnosis tasks, where input consists of symptoms which are seemingly disparate.

The selection of the $y_i r_i x$ in algorithm 47 should, in these cases, maximize information gain in the sense of definition 37(vi). In addition, one should augment the input procedure by adapting the abstraction level of the *input* elements analogous to step 5 in algorithm 41.

Second, in contrast to coming in disparate "out of the blue"-events, input can be already organized. This is characteristic, e.g., for signal processing. Input should, in such cases, be considered as coming in a *band*, where a band itself is a non-cyclic generator. Fig. 3.15 sketches an example:

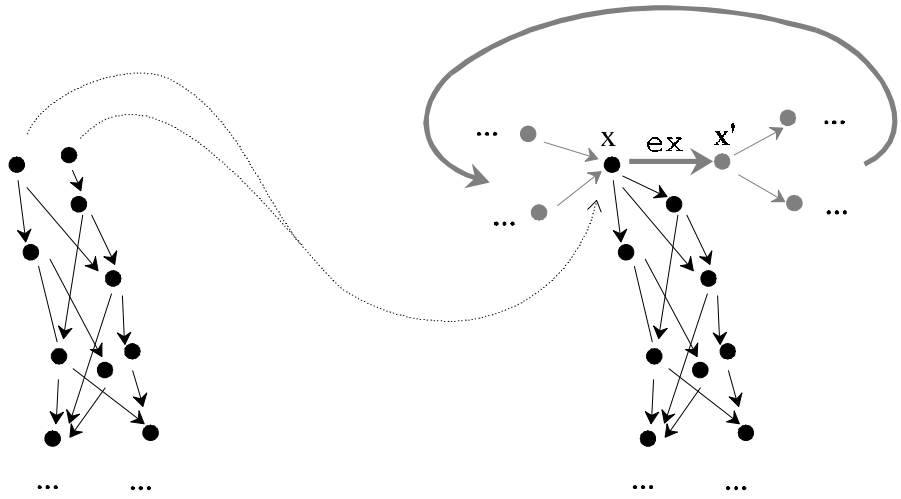


Fig. 3.15: An input band, and how it is coupled to the input port of an open configuration.

In a micro-input that reads from an input band, the direct successors of a local state of the input band are fed into the open configuration, yielding by their labels the *input* set required in algorithm 47.

In direct analogy to input bands one can define output bands. The band conception of I/O-organization is particularly helpful when several self-organizing (C_{ij})-streams are to be coupled together (see further below). They can be linked by bands of finite length.

I have called the ex-transition a "port". Elaborating on this computer terminology a bit, the transitions leading to and from this transition in an open configuration could be called input and output "channels". An open configuration thus accepts input, and generates output, on several channels in parallel. Their number can change in time.

However, the channel metaphor is not quite satisfying, since all input "channels" have access to the same input information. This is a result of the input band's being "slimlined" in order for it to pass through the "needle's eye" of the unique local state x. The same proviso holds for the output side. As a consequence of the band's "slimlining", self-organizing streams of the type described above cannot properly deal with truly parallel input, which would be required, e.g., for a retina-like module. One can, however, elaborate the basic form of self-organizing streams in order to arrive at a processing of truly parallel input and output bands. I hint at two possibilities:

- The input band's original topology can be reconstituted within the self-organizing stream, after it has passed through the "needle's eye" x. The reconstitution can be achieved by a suitable selection of the set $\{y_1 r_1 x, ..., y_m r_m x\}$ in algorithm 47. This leaves the definitions from above as they are, but requires an extra, somewhat unnatural memory mechanism in order to reconstruct the band topology after its passage through x.
- The definition of a self-organizing stream can be generalized by dropping the requirement that there is only one ex-transition. Essentially this means that in fig. 3.14b the terminal states are not fully cross-connected by ex-transitions with the initial states. The resulting set of local states $\{x \in S \mid$ there exists an ex-transition $x ex x' \in trans\}$ acts, then, as a kind

of "input retina" of the self-organizing stream. Analogically, an "output surface" exists. The details of this approach have to be spelled out with some care, since it must be guaranteed that the open configuration does not fall apart into disjoint structures.

Self-organization III: "dissipative" self-organizing streams

An interesting perspective on input is to consider it not as a source of "meaningful" material, but as a source of "energy" that helps to maintain the self-organizing stream in a state"far from equilibrium". Analogically, output can be used not only for reading out "results", but also as a sink for "used-up" informational entities. In thermodynamic terms, this corresponds to a *dissipative* process. Examples for dissipative processes is the functioning of a car engine or a biological cell. Energy-rich substances (gasoline or carbohydrates, fats, and proteins) enter the device; heat and low-energy compounds leave. Such a flow of energy, which is coupled to a "degradation" of the energy form, is a thermodynamic precondition for interesting (in particular, adaptive) forms of self-organization (cf. Prigogine & Stengers 1980).

In the case of self-organizing streams, the "energy-rich" substance consists in `top`-transitions (or more generally, in transitions of relatively abstract dynamic symbols). What happens when micro-inputs grant a continual supply of "fresh" `top`-transitions, and when there is no other kind of input? For a discussion, assume that the global state G_{kl} is comparatively specific, i.e., that it comes from the lower right region of (G_{ij}). Then, the most notable effect of `top`-transitions is that they are likely to become specialized as soon as they are hit by moving associations (in step 5 of the microchange algorithm), thereby adding new continuations to the association. For instance, taking the coherency from fig. 3.5 for G_{kl}, when an association `apple cost` hits `top`, the `top`-transition will become specialized to a `buy`-transition, and the association `apple cost buy` will form.

The "low-energy substances", which leave the system, are *any* transitions that happen to be destructively read out in micro-outputs in a random fashion. There may be `top`-transitions among them, but on the average, the abstraction level of outputted transitions will be lower than the maximally abstract level Σ_0 of `top`.

The net effect of such a "dissipative mode" for a self-organizing stream is that the associations contained in it can *wander* through the underlying G_{kl}. Since new dynamic symbols can be made from `top` raw material, the languages generated by open configuarations can successively encompass symbolic material that is not initially present in C_0. On a conceptual level, such a history resembles dream-like, "free" associations. Fig. 3.16 captures the essentials of such a dissipative "dream machine".

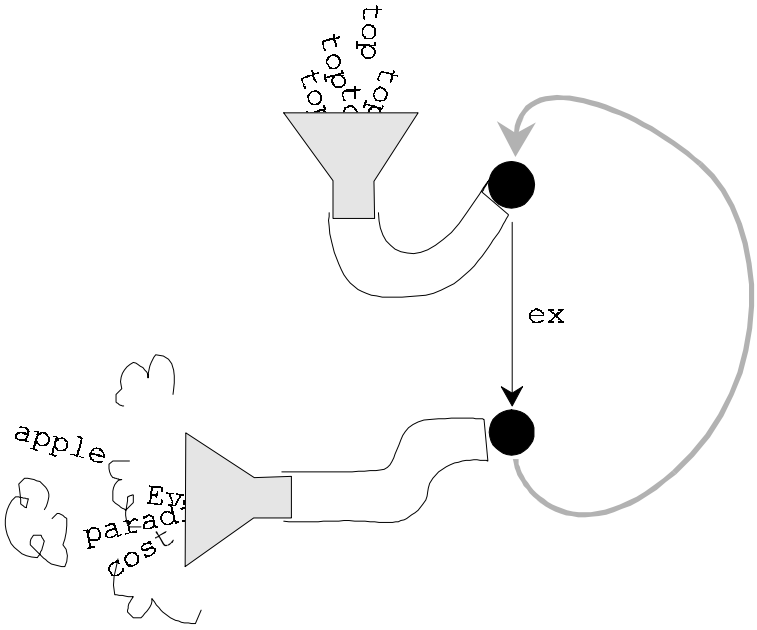


Fig. 3.16: A dissipative dream engine.

Supplying fresh `top`-transitions can (even should) be used as an additional mechanism in combination with another, "meaningful" input band. The self-organizing stream will then be able to exploit a constant supply of `top`-transitions to fill gaps in the dynamic symbol material provided by the input band, thereby increasing its powers to derive long associations and resonances.

Self-organization IV: equilibration and resonance formation in an open system

The remarks made in 3.3 concerning equilibration and self-organization largely carry over to self-organizing streams. The general picture, then, of a self-organizing stream is one of an open system with some fast self-organization effects, which will, however, not reach a long-term equilibrium due to incessant input and output. Since micro-inputs and micro-outputs are essentially independent from microchanges, a self-organizing stream is an *anytime-algorithm*, which (self-)organizes its input. When the frequency of microchanges is high compared to micro-inputs and micro-outputs, the degree of organization achieved is high. When it is small, the stream essentially degenerates to a mere transmission line, which does not change the input band much before it is again outputted.

This general picture is, however, a simplification. Due to the `ex`-transgression prohibition, some notable special effects arise. For a discussion of these effects, the notion of an open configuration's *body* is helpful. The body is simply the open configuration with the `ex`-transition removed:

Definition 51: The **body** of an open configuration C = (S, *trans*) is the generator (S, *trans*'), where *trans*' = *trans* - {xexx'}.

A first observation is that while an open generator is by definition cyclic, its body can, in the extreme, be entirely cycle-free, or even consist of separate linear paths which start in x' and terminate in x (fig. 3.17a). When all of these derivation paths yield G_{kl}-associations, microchanges will be of the empty type, i.e., they do not effectively change the open configuration. When such impasses occur, further interesting microchange activity can only occur after input has introduced again some "fresh disorder" (fig. 3.17b).

However, the fresh disorder might soon be absorbed by microchanges, and the stream might again turn stationary (fig. 3.17c). The way out of such a situation cannot solely lie in more input. The self-organizing stream is, in such cases, not really self-organizing; at best, each inputted transition runs through a few equilibrative microchanges, before it is either deleted or linked into a stationary association.

More interesting and comprehensive self-organization phenomena can only arise when the body contains one or several cyclic substructures which interconnect its "laminae" (fig. 3.17d). Such substructures are a precondition in a self-organizing stream for associations from different parts of the body to meet and interact. Of course, such substructures can themselves be modified by microchanges.

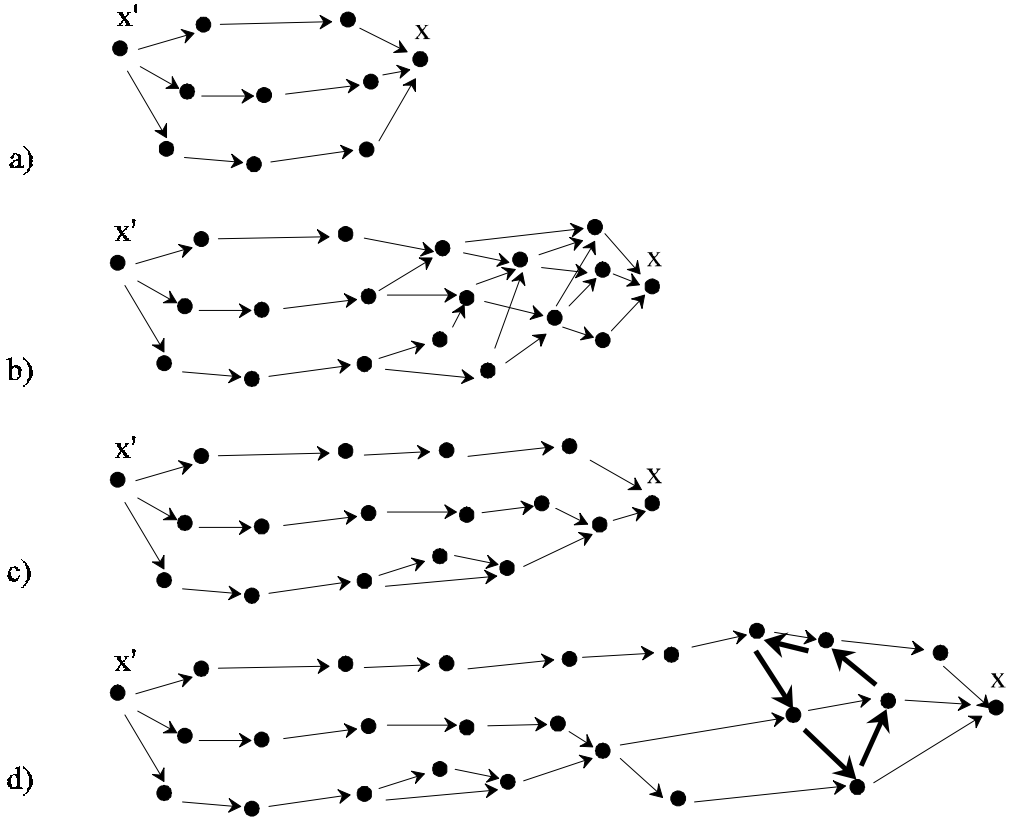


Fig. 3.17: Things that can happen to bodies. a) - d) are snapshots from an extended passage. For graphical clarity, the passage contains no micro-outputs. Micro-outputs would "eat up" bodies from left to right.

Since cyclic substructures are important to overcome impasses in cycle-free bodies, it is important that they can form spontaneously in a cycle-free body. Fortunately, microchanges *can* generate cycles in a cycle-free body. The mechanism is geometrically similar to the shear-induced formation of vortices in a fluid. Figure 3.18 shows a simple example:

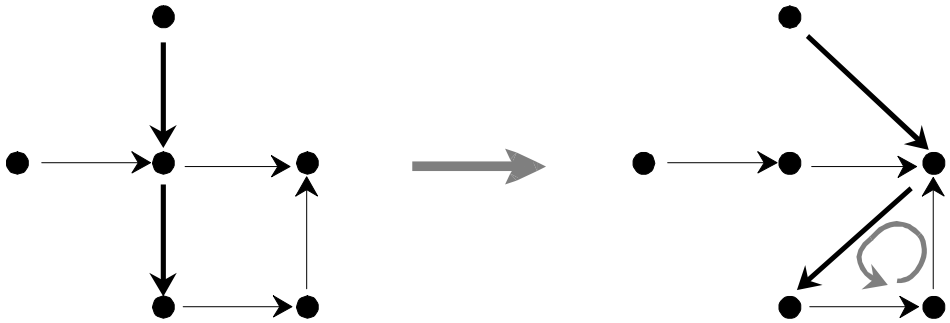


Fig. 3.18: The formation of a cycle as a by-product of a microchange. Only a portion of a body is shown.

Cyclic substructures in the body can or cannot be resonances. When the frequency of micro-changes is high compared to the frequency of micro-inputs, the considerations made in the case of self-organizing scenes carry over to streams. This means that cyclic, but non-resonant substructures are likely to produce resonances, provided that suitable dynamic symbols are present. When no potentially resonance-supporting dynamic symbols are present, the considerations from self-organizing scenes suggest that the cyclic substructure decays to a trivial single-state subgenerator. Then again, the body becomes stationary with respect to microchanges. In sum, only when resonances form in the body, can it become relatively autonomous in the sense that interesting activity can persist independently from micro-inputs.

These observations lead to a refined general picture of a self-organizing stream. When there are no cycles in the body of an open configuration, there is but a superficial equilibration-type of self-organization, which is essentially coupled to input time. Only when feedback loops giving rise to resonances appear within the body, can the internal microdynamics become temporally decoupled from input.

This corresponds to a basic introspective experience: one can detach one's mental processing from input flux only when relatively stable mental images are formed. When this is not the case, as in a roller coaster situation or in viewing agressively short cut video clips, one is "breathlessly" coupled into the stream of experience provided by sensory input, without much chance of cognitive detachment.

When a resonance exists in the body, it can be read out repeatedly. This is manifest, e.g., in that one can talk about mental images without erasing them, or perform repetitive movements. However, in talking about a mental image, it is also "rethought". This is accounted for by the active microchange nature of micro-outputs, which can induce modifications on the material that remains in the body.

Summary of section 3.3

- A self-organizing scene is the basic version of the DSS model of an information processing module. It is a closed dynamic system with no input or output.
- Each self-organizing scene is equipped with a dynamic symbol space, which yields its "long-term memory".
- A self-organizing scene essentially is a sequence of configurations (a history). A configuration is a cyclic directed graph whose edges are labeled by dynamic symbols from the underlying dynamic concept space. It is itself a generator of a language.
- Self-organization manifests itself in a history in that the languages generated by the configurations converge to a sublanguage of a global state from the underlying dynamic symbol space.
- The dynamics of a history is effected by local operations (called microchanges) in configurations. Microchanges operate on associations, i.e., on words derivable in a configuration that essentially belong to the language of the global state.
- Microchanges essentially let associations migrate along each other. An association that moves can grow at its front end, and induce specialization and abstraction effects in its vicinity. Such side-effects are the basis for the desired self-organiziation.
- A conspicuous phenomenon is the formation of resonances during a history. A resonance is a cyclic substructure of a configuration, which generates a sublanguage of the current global state. They are relatively stable, and they are the DSS analogue of gestalt phenomena.
- When a history starts from a disordered state, it will rapidly equilibrate in many cases such that there are only few (or one) non-competing resonances left, the history essentially becoming stationary. When the initial material contained in a self-organizing scene is ambiguous or conflicting, i.e., when there is a considerable degree of competition between associations, an equilibration is likely to be reached only after a longer interval. Generally, however, the ergodic behavior of self-organizing scenes is hard to predict analytically, and there exist other potential long-term outcomes of a history besides an equilibration to resonances.
- For two fundamental reasons, configurations are required to be phase generators. First, phase generators avoid "hidden variables". Second, phases yield a natural account of what is modeled by instances in logic-oriented AI.
- Associations have a dual process/structure character. Thus, DSS might be a suitable frame to investigate fundamental questions of self-organizing information processing. However, DSS cannot as yet provide a satisfying, comprehensive account of time and structure. But then, such an account does not currently exist anywhere else.
- Self-organizing streams are a generalization of self-organizing scenes, which allows input and output. Like the latter, they are described in terms of a sequence of configurations.
- For fundamental reasons, configurations are required to be phase generators, i.e., they are cyclically closed. In order to "open" them for I/O, a formal trick is used. A special `ex`-transition is introduced, which represents the *ex*terior within the configuration.
- Micro-inputs and micro-outputs are added to microchanges in the account of micro-dynamics.
- The input and output format can be specified by bands. A band is itself a (typically non-cyclic) generator. In the simplest case, a band is a sequence of dynamic symbols.
- When input is provided that consists of `top`-transitions, a self-organizing stream behaves like a dissipative system, using these transitions as a "fuel", which augments its self-organization capabilities.

- Being an open system, a self-organizing stream typically does not reach a stable state in its history. However, resonances can form in it, which are likely to persist for some time. Mental images and other gestalt phenomena are the intended cognitive correlates of resonances.

3.4 Associeties

In section 2, I have outlined a general view on agent architectures in terms of processing levels that are ordered on a periphery-centre axis. This general outline is concretely realized in DSS. In the remainder of this section, I describe two methods by which self-organizing streams can be coupled, thus giving rise to complex multi-stream architectures, *associeties*. The first method consists in making the output band generated by one self-organizing stream the input band of another. This technique seems to be suited for coupling modules that are "lateral" to each other in the periphery-centre dimension. The other method consists in re-interpreting a lower-level self-organizing stream directly, in terms of a higher-level self-organizing stream. This corresponds to grounding/emergence relations between adjacent levels (cf. 2.2, 2.7). Fig. 3.19 sketches an architecture made from many self-organizing streams by means of these two techniques.

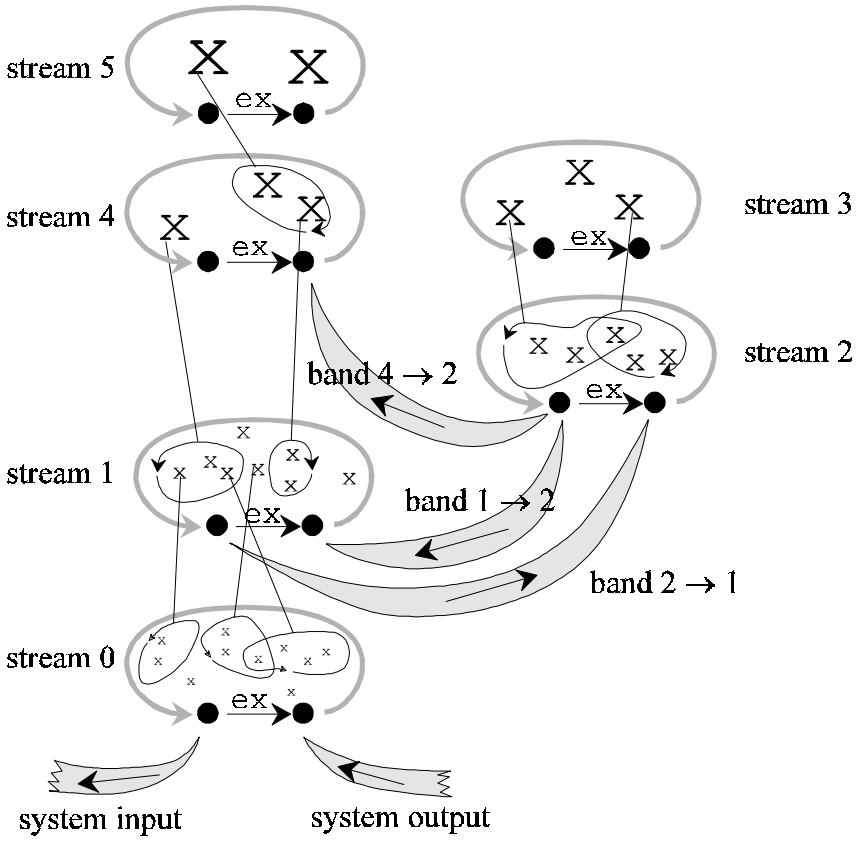


Fig. 3.19: An associety. Coarse-grained dynamic symbols are rendered by large X's, fine-grained by small x's.

The following comments will help to clarify the architecture sketched in the figure.

- Streams are coupled by two mechanisms: I/O bands (banana-shaped segments), and by grounding/emergence relations, which couple higher-level dynamic symbols to lower-level composites (streams $0 \leftrightarrow 1 \leftrightarrow 4 \leftrightarrow 5$ and $2 \leftrightarrow 3$). The latter mechanism will be explained later in the subsection.
- Higher-level streams are assumed to be coarser grained, i.e., they contain less dynamic symbols and develop slower. This is indicated in the schematic figure by the size of the local state dots that appear in the streams.
- All streams are assumed to be of the dissipative kind sketched further above. Thus, even when there are no I/O bands leading to/from a stream (streams 3 and 5), they are made from open configurations, which have an ex-transition.
- Streams 1 and 2 exchange information bidirectionally, whereas the communication from stream 4 to stream 2 is one-way only (note again that the direction of arrows reflects time, not communication direction!).
- The level topology is defined by emergence/grounding relations. The example has no simple linear level topology. There are two leveled "columns", which are interconnected by "lateral" bands. A band connection between two streams does not imply that they are on the "same" level. Bands can connect streams of different "granularities". In the extreme, a band could connect two streams that are also connected by emergence/grounding. This is, however, rather not intended from the view of dynamic symbol structures. Bands should be used to connect modules that don't stand in a (transitive) emergence/grounding relation with each other. In modeling biological brains, bands might be an appropriate model for the coupling of differently specialized cortical regions (e.g., auditory and visual), which is biologically achieved by transversal fibres.
- In the example, output and input from/to the environment is conveyed through a single, lowest-level stream (stream 0). This is an extremely simple architecture. In concrete applications, it will rather occur that external output is managed by a different module than external input, and there may be several external I/O channels leading to/from different streams.

The term "associety" is a free invention. It is intended to emphasize the dominant role of associations in DSS multi-stream architectures, and it reminds one of Minsky's (1985) "society of mind". Minsky's society of mind metaphor characterizes a perspective on agent architectures, where many symbolic information processing devices ("agents") interact on many hierarchic levels. This generally resembles the perspective of dynamic symbol structures.

I shall only provide local specifications of the two coupling mechanisms, not a global definition of entire associeties. The reason for refraining from fixing the term precisely is that I wish to keep the term open in its meaning, such that it can cover any agent architecture made from self-organizing streams. Architectures like the one from fig. 3.19 are what I have in my mind presently. Others might be invented, for instance, exploiting the "topological" variants indicated in fig. 2.1.

The term "dynamic symbol system", which gives name to the thesis, is intended to be even more general and open. It is meant to refer to any information processing system with a microchange dynamics controled by dynamic symbol spaces. Self-organizing scenes, self-organizing streams, and associeties are the presently available instances of dynamic symbol systems.

So much for a general impression. I shall treat now the two coupling mechanisms in more detail.

Coupling streams by bands

Different self-organizing streams are typically equipped with different dynamic symbol spaces. Interstream communication thus requires that information from some dynamic symbol space $(G_{ij})_{i=0,...,m\ j=0,...,n}$ is *translated* into information pertaining to another concept space $(G'_{ij})_{i=0,...,m'\ j=0,...,n'}$.

There are many reasonable ways of specifying a translation between (G_{ij}) and (G'_{ij}). I will opt for a fairly general version, which treats translations as a relation between associations.

Definition 52: Let $(G_{ij})_{i=0,...,m\ j=0,...,n}$, $(G'_{ij})_{i=0,...,m'\ j=0,...,n'}$ be dynamic symbol spaces with underlying sets of dynamic symbols Σ, Σ'. Let $A((G_{ij}))$ denote the set of all associations which can be sampled accross (G_{ij}), i.e.

$$A((G_{ij})) = \{r_1...r_n \in \Sigma^n \mid n \geq 1,\ r_1...r_n \text{ is a } G_{kl}\text{-association for some } G_{kl} \in (G_{ij})\}.$$

Let $\tau \subseteq A((G_{ij})) \times A((G'_{ij}))$. Write, as a shorthand, $r_1...r_n \leftrightarrow_\tau r'_1...r'_m$ for $(r_1...r_n, r'_1...r'_m) \in \tau$. Then τ is a **translation** between (G_{ij}) and (G'_{ij}) :iff

(i) τ is total, i.e., the range of τ in its first [second] component is $A((G_{ij}))$ [$A((G'_{ij}))$, respectively].

(ii) τ respects abstraction in the first component, i.e., if $r_1...r_n \leftrightarrow_\tau r'_1...r'_m$, and $s_1...s_n \in A((G_{ij}))$, $s_1...s_n \geq r_1...r_n$, then there exists $s'_1...s'_m \in A((G'_{ij}))$, $s'_1...s'_m \geq r'_1...r'_m$, such that $s_1...s_n \leftrightarrow_\tau s'_1...s'_m$.

(ii') τ respects abstraction in the second component (analogous).

(iii) τ respects specialization in both components (analogous).

(iv) τ is respects concatenations, i.e., if $r_1...r_n \leftrightarrow_\tau r'_1...r'_m$, $s_1...s_p \leftrightarrow_\tau s'_1...s'_q$, $r_1...r_n s_1...s_p \in A((G_{ij}))$, and $r'_1...r'_m s'_1...s'_q \in A((G'_{ij}))$, then $r_1...r_n s_1...s_p \leftrightarrow_\tau r'_1...r'_m s'_1...s'_q$.

(v) τ respects subsequences in the first component, i.e., if $r_1...r_n \leftrightarrow_\tau r'_1...r'_m$, and $s_1...s_p$ ($p \geq 1$) is a subsequence of $r_1...r_n$, then there exists a subsequence $s'_1...s'_q$ ($q \geq 1$) of $r'_1...r'_m$, such that $s_1...s_p \leftrightarrow_\tau s'_1...s'_q$.

(v') τ respects subsequences in the second component (analogous).

Note that (iv) comes not in the following, stronger form: if $r_1...r_n \leftrightarrow_\tau r'_1...r'_m$, $r_1...r_n s_1...s_p \in A((G_{ij}))$, then there exists $s'_1...s'_q \in A((G'_{ij}))$, $s_1...s_p \leftrightarrow_\tau s'_1...s'_q$, such that $r_1...r_n s_1...s_p \leftrightarrow_\tau r'_1...r'_m s'_1...s'_q$ (plus an analogical requirement for the other direction). I.e., it is not guaranteed that when a translation of $r_1...r_n s_1...s_p$ into $A((G'_{ij}))$ has got so far as to translate $r_1...r_n$ by $r'_1...r'_m$, the translation can be finished to cover all of $r_1...r_n s_1...s_p$. This is a kind of "garden path" phenomenon, which reflects that translations can be ambiguous with respect to contexts. The "first attempt" to translate $r_1...r_n$ by $r'_1...r'_m$ pins down a context in (G'_{ij}) which cannot be further continuated by any translation of $s_1...s_p$. If one would not admit this kind of continuation conflict, one would essentially require that the topological structures of the phase generators of the G_{ij} are identical to those of the G'_{ij}.

Two trivial cases of translations are the following:

Proposition 53:

(i) For any (G_{ij}) and (G'_{ij}), $\tau = \{(r_1...r_n, r'_1...r'_n) \mid n \geq 1, r_1...r_n \in A((G_{ij})), r'_1...r'_n \in A((G'_{ij}))\}$ is a translation.

(ii) The identity on $A((G_{ij}))$ is a translation between $A((G_{ij}))$ and $A((G_{ij}))$. □

When a band couples two streams S and S' with different underlying $A((G_{ij}))$ and $A((G'_{ij}))$, it must be translated after it leaves S and before it enters S'. This translation usually cannot be a simple symbol-to-symbol rewriting, since associations that are translated to each other need not have the same length, and since continuation conflicts can arise. There are many practical solutions for the rewriting task. One of them is sketched in the following algorithm.

Algorithm 54 (sketch): Let B = (S, *trans*) be a generator (intention: B is a band) that uses dynamic symbols from (G_{ij}), and let τ be a translation between (G_{ij}) and (G'_{ij}). Then rewrite B to arrive at a generator B', which uses dynamic symbols from (G'_{ij}), as follows:

Step 1: Identify in B derivations of associations from $A((G_{ij}))$, such that all transitions in B are covered by these associations. Let *COVER* be the set of these derivations.

Step 2: For every derivation D in *COVER*, apply τ to the association $\boldsymbol{a}$ derived by D, to get some association $r'_1...r'_n \in A((G'_{ij}))$. Construct for $r'_1...r'_n$ a linear generator

$$G'_D = (\{x^D_0, ..., x^D_{n_D}\}, \{\{x^D_0 r'_1 x^D_1\}, ..., \{x^D_{n-1} r'_n x^D_{n_D}\}\}).$$

The set of all the simple generators such constructed is *COVER'*.

Step 3: Interconnect the generators from *COVER'* as follows. For all D_1, D_2, for which it holds that the initial local state of D_2 is a local state that also occurs in D_1, merge G'_{D_1} with G'_{D_2} by identifying the last local state $x^{D_1}_{n_{D_1}}$ of G'_{D_1} with the initial local state $x^{D_2}_0$ of G'_{D_2}. The net result of all these merges is the desired B'.

This algorithm is "quick and dirty". Many topological details of the original B will get lost. However, when one works with self-organizing, collective processes, one can be generous with respect to detail. Straightforward modifications of the algorithm allow the rewriting to be executed in an incremental fashion, such that a band B can be continually rewritten in the measure as it is produced by micro-outputs.

Communication via bands is a simple method. In many cases, this method will also work to couple a self-organizing stream into an agent architecture, where different modules are realized by different techniques, e.g., connectionist and classical ones. There are two directions to be considered: output of a non-DSS module serves as input for a self-organizing stream, and vice versa. Implementing the first direction is likely to be easy, since any serial symbolic output essentially is already a (linear) band, and since self-organizing streams impose no preconditions on the "syntax" of the input band. The other direction is less likely to be easy. A module that inputs a band generated by a self-organizing stream must be suited to use this input format, which has no other syntax besides being a labeled directed graph, or more simply, being a sequence of symbols. In many cases, such a type of input will be used not as the only source of input, but rather as an additional clue for "soft" tasks. Generally, information provided by self-organizing streams will be useful for tasks which have some or all of the following characteristics:

- The task can be interpreted in terms of gestalt formation.
- Some portions of the information processed in the task can be interpreted as a context for other portions.
- The task does not consist of clearly defined subtasks with a precisely timed control.
- There is no hard success criterium; suboptimal results are acceptable when they come fast.
- The task is of a stream processing nature.
- The quality of solutions can be judged in terms of a degree of specialization, or precision, of the output.

This list characterizes many of the pattern classification and generation tasks that occur in situated agents. Therefore, integrating self-organizing streams into such architectures seems generally well-motivated.

Coupling streams by emergence/grounding

In the dynamic symbol structure framework (cf. section 2), dynamic symbols ground in dynamic composites from the next lower level. The latter's concrete nature is left open in the abstract framework. In the DSS formalism, the basic idea is to take *resonances* for dynamic composites. Intuitively, this means that a "gestalt" made from fine-grained dynamic symbols can be perceived from a higher level as a unit, and be referenced by a single dynamic symbol there. A desirable side-effect is that higher-level dynamics are thus made typically slower than lower-level dynamics, since resonances are relatively persistent phenonema.

Technically, I define grounding as a relation between dynamic symbol spaces, i.e., between the "long-term memories" of level-adjacent streams. Therefore, I consider two dynamic symbol spaces (G_{ij}) and (E_{ij}), where the second is made from "coarse" dynamic symbols that ground in resonances found in the first (G ~ *G*rounding level, E ~ *E*mergent level). Since the concrete form of global states $G_{kl} \in (G_{ij})$ and $E_{kl} \in (E_{ij})$ is essentially arbitrary, a generally satisfying account of emergence/grounding should not depend on the concrete G_{kl}, E_{kl}, but rather on the generated languages $C_{G_{kl}}$ and $C_{E_{kl}}$. For the remainder of the section, I therefore take the array $(G_{ij}) := (C_{G_{ij}})$ of languages of a dynamic symbol space as a starting point, instead of the dynamic symbol space proper.

I start by considering a single coherent language G and explain what it means for another coherent language E to emerge from it. This requires some work. The first thing to do is to define resonances in terms of languages (the G_{kl}-resonance definition in subsection 3.3 relies on generators).

Definition 55: A coherent sublanguage $R \subseteq G$ of a coherent language G is a **resonance** in G.

When R is a resonance in G, and an infinite continuation $\boldsymbol{s} = s_1 s_2 ... \in G^\infty$ is observed, a question of interest is to detect points in $\boldsymbol{s}$ where this continuation "passes through" R. This leads to the notion of *characters* of R. A character of R is a word $\boldsymbol{r}$ from R, which indicates that, wherever $\boldsymbol{r}$ is observed as a subsequence in $\boldsymbol{s}$, the latter is, at that place, "actually in R". I.e., from that place onward, $\boldsymbol{s}$ could be able to continue indefinitely in R:

Definition 56:

(i) For $R \subseteq G$, $\boldsymbol{r} \in R$, $continue_R(\boldsymbol{r}) := \{\boldsymbol{t} \in R \mid \boldsymbol{rt} \in R\}$.

(ii) A word $\boldsymbol{r} \in R$ of a resonance R in G is a **character** of R in G :iff for all $\boldsymbol{sr} \in G$, $continue_R(\boldsymbol{r}) \subseteq continue_G(\boldsymbol{sr})$.

Proposition 57: Every resonance $R \subseteq G$ has characters.

Sketch of proof: Let $\boldsymbol{r} \in R$ be phase-fixing in R, $\varphi_{\boldsymbol{r}}$ the phase of $\boldsymbol{r}$ in R. Let R_φ, G_φ be the phase generators of R and G. Consider all simulations σ: $R_\varphi \rightarrow G_\varphi$ (the simulation theorem warrants that such simulations exist). Let $\Phi_{\boldsymbol{r}}$ denote the set of all phases in G that are $\varphi_{\boldsymbol{r}}$-images with respect to one of the these simulations, i.e. $\Phi_{\boldsymbol{r}} = \{\varphi \in \Phi(G) \mid \varphi \in \sigma(\varphi_{\boldsymbol{r}})$ for some σ: $R_\varphi \rightarrow G_\varphi\}$. Define for $\varphi \in \Phi(G)$

$$continue_{G_\varphi}(\varphi) := \{\boldsymbol{s} \in G \mid \boldsymbol{s} \text{ can be derived in } G_\varphi \text{ on a path starting in } \varphi\}.$$

It holds that $continue_R(\boldsymbol{r}) \subseteq continue_{G_\varphi}(\varphi)$ for all $\varphi \in \Phi_{\boldsymbol{r}}$. Let $\Phi^*_{\boldsymbol{r}}$ denote the set of all phases in G such that $\boldsymbol{r}$ can be derived in G_φ on a path terminating in one of these phases. It holds that $\Phi_{\boldsymbol{r}} \subseteq \Phi^*_{\boldsymbol{r}}$. If $\Phi_{\boldsymbol{r}} = \Phi^*_{\boldsymbol{r}}$, $\boldsymbol{r}$ is a character of R in G, and nothing remains to be shown. If $\Phi_{\boldsymbol{r}} \subset \Phi^*_{\boldsymbol{r}}$, then there exists some $\boldsymbol{rr}_1 \in R$, such that $\boldsymbol{r}_1 \notin continue_{G_\varphi}(\varphi)$ for some $\varphi \in \Phi^*_{\boldsymbol{r}}$. Define $\Phi_{\boldsymbol{rr}_1}$ and $\Phi^*_{\boldsymbol{rr}_1}$ analogous to $\Phi_{\boldsymbol{r}}$ and $\Phi^*_{\boldsymbol{r}}$. It holds that $|\Phi_{\boldsymbol{r}}| = |\Phi_{\boldsymbol{rr}_1}|$, $|\Phi^*_{\boldsymbol{r}}| > |\Phi^*_{\boldsymbol{rr}_1}|$, and $\Phi_{\boldsymbol{rr}_1} \subseteq \Phi^*_{\boldsymbol{rr}_1}$. Iterate until for some $n \geq 1$ it holds that $\Phi_{\boldsymbol{rr}_1 \ldots \boldsymbol{r}_n} = \Phi^*_{\boldsymbol{rr}_1 \ldots \boldsymbol{r}_n}$. Then $\boldsymbol{rr}_1 \ldots \boldsymbol{r}_n$ is a character of R in G. □

Proposition 58: If $\boldsymbol{r}$, $\boldsymbol{r}_0\boldsymbol{rr}_1 \in R$, and $\boldsymbol{r}$ is a character of R in G, then $\boldsymbol{r}_0\boldsymbol{rr}_1$ is a character of R in G. I.e., words in R, which contain a character as a subword, are themselves characters.

Sketch of proof: That $\boldsymbol{r}_0\boldsymbol{r}$ is a character of R in G, follows from $continue_R(\boldsymbol{r}_0\boldsymbol{r}) \subseteq continue_R(\boldsymbol{r}) \subseteq continue_G(\boldsymbol{sr}_0\boldsymbol{r})$. Thus, it remains to be shown that if $\boldsymbol{r}$, $\boldsymbol{rr}_1 \in R$, and $\boldsymbol{r}$ is a character of R in G, then $\boldsymbol{rr}_1$ is a character of R in G. Assume that this is not true, i.e., there exists $\boldsymbol{s} \in G$, such that there exists $\boldsymbol{rr}_1\boldsymbol{r}_2 \in R$, where $\boldsymbol{srr}_1\boldsymbol{r}_2 \notin G$. This is a contradiction to $\boldsymbol{r}$ being a character of R in G. □

When lower-level resonances are interpreted via grounding/emergence by higher-level dynamic symbols, it would be nice if one could "recognize" the lower-level resonances by their characters. I.e., each of the resonances that give rise to higher-level dynamic symbols should be discernible from the others by at least one of its characters. This motivates the following definition:

Definition 59: Let $Res = \{R_1, ..., R_k\}$ be a set of resonances in G. Then Res is **separable by characters** :iff for all R_i, there exists a character of R_i which is not a character of any of the other R_j.

The following proposition gives a sufficient condition for separability by characters:

Proposition 60: $Res = \{R_1, ..., R_k\}$ is separable by characters if none of the R_i is a sublanguage of another R_j.

Sketch of proof: In order to construct, e.g., a character of R_1, which is not a character of any of the other R_2, ..., R_n, take a character $\boldsymbol{r}$ of R_1, select words $\boldsymbol{r}_2$, ..., $\boldsymbol{r}_n \in R_1$ which are not

contained in R_2, ..., R_n, and construct a word from R_1 which contains $\boldsymbol{r}$ and all $\boldsymbol{r}_i$ as subwords. Then apply proposition 58. □

Now the scene is prepared to define E as an *emergent* language over G, or, equivalently, G as a language in which E *grounds* (the letters E, G are motivated by **e***mergent* and **g***rounding*). The alphabet of E corresponds to a set *Res* of resonances of G.

Definition 61: Let G be a coherent language, and $Res = \{R_1, ..., R_k\}$ be a set of resonances in G. Let $\Sigma_{Res} = \{a_1, ..., a_k\}$ be a set of dynamic symbols. Define the **emergence mapping** ε: $Res \rightarrow \Sigma_{Res}$ by $\varepsilon(R_i) := a_i$, and the **grounding mapping** $\gamma := \varepsilon^{-1}$. Then, E **emerges over** G (more precisely, it emerges over G by ε), if the following condition holds:

$s_1...s_m \in E$ iff there exists a word $\boldsymbol{r}_1...\boldsymbol{r}_m \in G$, such that $\boldsymbol{r}_i$ is a character of $\gamma(s_i)$.

When E emerges over G, then, the other way round, E **grounds in** G (by γ).

Extending the scope of the term "grounding", we say that a_i grounds in R_i, when E emerges over G by ε, and $\varepsilon(R_i) := a_i$. Furthermore, when $\boldsymbol{r}_i$ is a character of R_i in G, we also say that a_i grounds in $\boldsymbol{r}_i$.

Proposition 62: Let G be a coherent language. If E emerges over G, then E is segmentable and regular.

Sketch of proof: It is clear that E is segmentable. By proposition 8, regularity is now equivalent to the existence of a finite generator for E. Let $G = (S, trans)$ be a coherency generating G. Define a finite generator E for E by $E = (S, trans')$, where $trans' \subseteq S \times \Sigma_{Res} \times S$ is given by $xsy \in trans'$:iff there exists a character $\boldsymbol{r}$ of $\gamma(s)$, which can be derived in G on a path starting in x and ending in y. □

I have not required in the preceding definitions that E be coherent. However, as a candidate for a global state language in a "coarse" dynamic symbol space, E should be coherent. The following definition and proposition gives a sufficient condition for E being coherent.

Definition 63: Let G be a coherent language, and $Res = \{R_1, ..., R_k\}$ be a set of resonances in G. *Res* **covers** G :iff there exists a word $\boldsymbol{r}_1...\boldsymbol{r}_m \in G$, where $\boldsymbol{r}_i$ is a character of some resonance from *Res*, such that $\boldsymbol{r}_1...\boldsymbol{r}_m$ can be derived in some generator of G on a path that visits every transition of the generator.

Proposition 64: If *Res* covers G, then E is a coherent language.

The proof is straightforward. □

The first part of the task is thereby completed: it is now clear how a coherency can give rise to a coarser-grained, higher-level coherency. Now assume that a dynamic symbol space $(G_{ij})_{i=0,...,m\ j=0,...n}$ with the corresponding array of languages (G_{ij}) is given, and that E_{mn} emerges over G_{mn}, where E_{mn} is a coherent language. What one would like is to find an array $(E_{ij})_{i=0,...,m\ j=0,...n}$ of coarse-grained, coherent languages, such that E_{ij} emerges over G_{ij}, and that the E_{ij} are generated by some coherencies E_{ij}, which give rise to a dynamic symbol space (E_{ij}). I cannot offer such a construction as yet. It is possible (albeit tedious) to construct languages E_{ij}, which emerge over the given G_{ij}, and which satisfyingly mirror the abstraction

and symmetrification series that characterize the original (G_{ij}). However, the good news ends at that. I cannot yet show that, or formulate conditions when, a dynamic symbol space for the array (E_{ij}) actually exists.

This present shortcoming of the theory mainly lies in the fact that dynamic symbol spaces are defined via *generators*, and the emergence relation is formulated for *languages*. It is easy to go from an array of generators (G_{ij}) to an array of languages (G_{ij}) and onwards to emergent languages (E_{ij}), but the last step back to an array of generators (E_{ij}) is barred, since the original conditions concerning abstraction and symmetrification sequences are lost on the way, due to their being generator-dependent. I leave the issue at rest for the time being, since I believe it is a more productive strategy to search for a generator-independent definition of dynamic symbol spaces, instead of patching the inconveniences afforded by the present, imperfect, generator-dependent definition.

Taking what I have positively got, I shall now assume for the remainder of the section a fine-grained dynamic symbol space $(G_{ij})_{i=0\ j=0}$, which consists of a single global state G_{00} =: G. I.e., $(G_{ij})_{i=0\ j=0}$ = (G). G generates the coherent language *G*; *E* emerges over *G* via ε: *Res* → Σ_{Res}, where *Res* = $\{R_1, ..., R_k\}$ and Σ_{Res} = $\{a_1, ..., a_k\}$. Furthermore, I assume that *E* is coherent, and that *Res* is separable by characters. Let E be some coherency generating *E*, and (E) the corresponding single-global-state dynamic symbol state.

In order to facilitate the matter further, I shall treat the emergence/grounding coupling only for closed systems, i.e., self-organizing *scenes*. Let C^G be a closed configuration made with dynamic symbols from *G*, and C^E a configuration made with dynamic symbols from *E*. How does ε lead to a coupling between C^G and C^E?

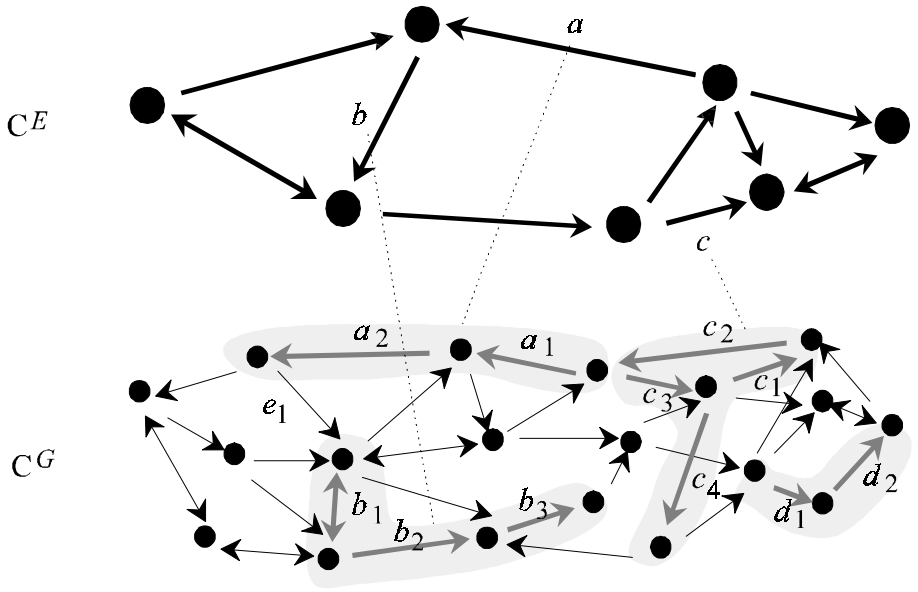


Fig. 3.20: A higher-level configuration C^E, coupled to a lower-level configuration C^G by grounding/emergence. Shaded areas in C^G are characters of resonances, broken lines denote χ-links between lower-level dynamic composites (i.e., characters) and higher-level dynamic symbols. For instance, the derivation path for a_1a_2 in C^G is χ-linked to the *a*-transition in C^E.

There are many ways to specify such a coupling. I will examine a version which builds on weak requirements. The idea is to use characters of resonances for the dynamic composites in C^G in which dynamic symbols from C^E ground. Figure 3.20 shows an example and definition 65 provides the particulars.

Definition 65: Let $C^G = (S^G, trans^G)$, $C^E = (S^E, trans^E)$ be configurations with underlying dynamic symbol spaces (G) and (E), where the language *E* generated by E emerges over the language *G* generated by G via ε: $Res \rightarrow \Sigma_{Res}$.

(i) A derivation $x_0 s_1 x_1 ... x_{n-1} s_n x_n$ in C^G is a **dynamic *E*-composite** :iff $s_1...s_n$ is a character of some resonance $R \in Res$. The set of dynamic *E*-composites in C^G is denoted by $dyncomp_E(C^G)$.

(ii) A partial, injective function χ: $dyncomp_E(C^G) \rightarrow trans^E$ is an **emergence coupling** between C^G and C^E, if it holds that

$$\chi(s_1...s_n) = xsx' \Rightarrow \text{there exists } R \in Res \text{ of which } s_1...s_n \text{ is a character, and } \varepsilon(R) = s.$$

This is quite a weak specification of an emergence coupling. Stronger versions could be derived by additional requirements, e.g., that dynamic *E*-composites be cyclic (i.e., resonant) substructures of C^G, or that χ is totally defined, or that the characters used by χ separate the resonances concerned.

The emergence coupling χ between C^G and C^E alone is a static relation without much use. In order to make it effective for bottom-up and top-down interactions, it must influence micro-changes in C^G and C^E. I indicate a preliminary list of methods for achieving this (referring to the steps of algorithm 41):

- Prefer, in C^G, for selection in step 1 associations which are characters mapped by χ into C^E. Intuitively, this means that lower-level associations which are "recognized" in the higher level are more salient and more active than unrecognized associations. This will favor the former's growth and stability relative to the latter's. Often a character will be "long enough" to loop back into itself, forming a resonance. Preferring characters for microchanges, therefore, enhances the likeliness of resonance formation.
- Prefer, in C^E, for selection in step 1 associations whose derivations contain χ-images. Intuitively, this favors grounded associations over such that appear "ad hoc" in C^E.
- Assume that type 4 microchanges are employed in C^G (i.e., a moving association does not spread homogenously but can select a direction). Assume further that in C^E there exists a derivation for some association *ab*, where *a*, *b* are χ-coupled to characters ***a***, ***b*** in C^G, and where in C^G ***a***, ***b*** are not connected (as in fig. 3.20). Then move, if possible, ***a*** into the direction of ***b*** in C^G (in fig. 3.20: move ***a*** along the e_1-transition). This strategy drives the topology of C^G towards the topology of C^E (modolo granularity). Analogically, microchanges in C^E can be biased to approximate the topology of C^G in C^E.
- Assume that the underlying dynamic symbol spaces for C^G and C^E have more than one global state, featuring abstraction of dynamic symbols and associations. Assume further that χ is suitably generalized to couple dynamic symbols in C^E with characters in C^G in a fashion that allows for some variance in the abstraction of the coupled entities. Assume that a microchange in C^G concerns an association ***a*** that is interpreted in C^E by *a*, where *a* is more special than ***a***. Modify step 5 (the generalization/abstraction step) such that the specialization information surplus given by *a* is exploited for a stronger specialization of

cont^{+} than would be effected by ***a*** alone. This is a realization of the "abstraction gradient" mentioned in section 2.6. The mechanism can be analogically exploited for bottom-up induced specializations in C^E.

Such mechanisms foster self-organization in histories by exploiting higher-level information on the lower level and vice versa. There are bottom-up and top-down variants of each mechanism, which have equal rights. No external control strategy exists. The lower-level history and the higher-level history each proceed autonomously at their own chosen "speed", accepting information from the other level when suitable, but not depending on it. However, the general intuition behind a multi-level architecture suggests that microchanges are applied to higher levels less frequently than to lower levels.

I have tacitly ignored a technical difficulty. The renormalization to phase generators (step 7) can have the effect that characters in C^G, or transitions in C^E, are merged or duplicated. This poses the question of how χ is maintained accross the renormalization step. Again, "quick and dirty" methods to deal with this complication are not hard to devise (e.g., do not duplicate χ-links together with characters). Another question concerns the original source of the χ-links, and mechanisms to introduce new such links during a history. A simple method would be to start entirely without such links, and test every association selected for a microchange for its ability to give rise to a χ-link.

Figure 3.20 conveys a slightly misleading impression, insofar as there are only a few isolated χ-links in it. In systems at work, it should be expected that the coupling between levels is tighter than fig. 3.20 suggests.

A variant of great interest is to construct an emergence relation not between two dynamic symbol spaces, but within a single one. This is quite natural, at least on a conceptual level. For instance, the dynamic symbol `Garden_of_Eden` might ground in a resonance characterized by, e.g., `Eve Adam garden apple`. Constructions of this kind are a natural model for the cyclic closure solution to the Münchhausen trilemma at the upper end of the periphery-centre dimension. They capture the closedness of conceptual systems that is revealed in the cyclic closedness of an encyclopædia (cf. section 2.4). I have mentioned on several occasions that coarser-grained dynamics should be "slower" than the dynamics on finer-grained levels. Now, χ-links *within* a self-organizing stream indicate that there are portions of the stream that are comparatively "more dynamic" than others. Cognitively, this corresponds to the observation that concepts, which are relatively persistent in short time memory, organize the processing of clusters of more ephemerical concepts. For instance, one can "think about", e.g., `Garden_of_Eden` for quite a long time, this dynamic symbol being continually present, while the characters it grounds in are in flux. E.g., `Eve Adam garden apple` may develop first into `apple serpent Eve`, then into `Eve apple temptation`, etc. This associative activity does not disperse and eventually gets lost, but is continually re-focussed by the persistent influence of `Garden_of_Eden`.

A self-organizing stream can be coupled, by emergence/grounding, primarily to other self-organizing streams. In contrast to band communication, χ-links exploit the detailed internal structure of self-organizing streams. Therefore, only modules that can be interpreted as self-organizing streams can, in some fashion, be coupled to DSS self-organizing streams by emergence/grounding.

Constructing non-DSS modules that essentially behave like a self-organizing stream is not a far-fetched idea. In particular, I believe that DSS streams can serve as a guide for constructing quite interesting neural networks. The basic idea is,

- first, to interpret attractor states in a recurrent network as dynamic symbols. Since more than one dynamic symbol exists in a self-organizing stream, the network must be capable,
- second, to maintain several attractor states simultaneously. They can be localized in different regions of the network, but they might also be globally superimposed on each other.
- Third, the attractor states AS must be able to form "associations" of the form $AS_1...AS_n$. This boils down to a mechanism that favors the stability of AS_i given the presence of AS_{i-1}.
- Fourth, there must be a "move" dynamics, analogous to microchanges, which warrants that existing associations are brought into mutual contact in a dynamic fashion. Such a contacting mechanism can be of any nature. For instance, it can be effected by spatial neighborhood, or by correlations between broadcasted spike train patterns.

When these four points are realized, the network by and large resembles a self-organizing stream. Connectionist networks of this kind are biologically plausible; in fact, much that is presently known about the neocortex fits into the picture. The DSS construct of a self-organizing stream, in this view, is (a) an abstract model of the information processing in such networks, and (b) possibly helpful in guiding the design of artificial networks, by providing clues to the designer concerning which phenomena are to be realized, and how they interrelate.

Be this as it may, it seems premature to invest greater efforts into complex system architectures. It is more urgent, at present, to examine the behavior of simple DSS modules in simulations, in order to gain experience with the various phenomena of self-organization, and their control by global states and other parameters.

Summary of section 3.4

- Several self-organizing streams can be coupled together via band communication and by emergence/grounding relations, forming a multi-level architecture called an associety.
- Different streams in an associety will typically build on different dynamic symbol spaces. When two such streams are coupled by a band, the band must be translated from one dynamic symbol space's vocabulary to the other's. Thanks to the self-organizing nature of the information processing involved, this can be done in "quick and dirty" fashions.
- Band communication is a simple method, which is suited to integrate self-organizing streams into hybrid architectures.
- By emergence/grounding, a lower-level, fine-grained stream is coupled to a higher-level, coarse-grained stream. The basic idea is to interpret (characteristic parts of) lower-level resonances by higher-level dynamic symbols.
- A fully satisfying formal account of emergence/grounding is impaired by the present, imperfect definition of dynamic symbol spaces in terms of generators (instead of languages or phase generators).
- Emergence/grounding links between a lower-level and a higher-level stream should influence microchange activity in both, in a fashion such that bottom-up and top-down effects have equal rights. Each side supports self-organization in the other.

- Emergence/grounding links within a single stream provide a solution to the Münchhausen trilemma at a conceptual level, formalizing an aspect of cyclic closedness of conceptual systems.
- Self-organizing streams might serve as a guide in the development of non-DSS modules, in particular, connectionist networks that capture some aspects of biological information processing in the neocortex.

4 Application proposal: memory access

In this section, I describe an example of how a self-organizing stream could be used as an auxiliary device in an otherwise classical natural language processing system. The stream's self-organization capabilities are exploited for focussing relevant portions of a modular knowledge base. The basic idea is summed up in the following points:

- A classical knowledge base is partitioned into modules in a hierarchic fashion.
- Each module is labeled by a resonance from a dynamic symbol space.
- A sequence of dynamic symbols ("keywords") is derived from the natural language input and fed into a self-organizing stream.
- Characters of resonances are detected when they develop in the stream.
- Such a character is indicative for the knowledge base module which is labeled by the corresponding resonance.

More specifically, I use Wachsmuth's (1989) model of a partitioned knowledge base as a fundament for the following considerations. The model has grown out of empirical studies in mathematical problem solving, where it was examined to determine which portions of specific knowledge subjects use. I begin with a brief review of this model, *knowledge packet structures*.

A knowledge packet structure formalizes the global structuring of (parts of) long-term memory. The total memory content comes in separate modules, called *knowledge packets*, which are organized in an access hierarchy. General knowledge is contained in packets high in the hierarchy, more specific knowledge is found in lower packets. Figure 4.1 gives an example. The example is motivated by an assumed application of a natural language processing system as an information system in an urban tourist office, which proposes places where tourists can go. Packet P1 contains general knowledge about going out, P2 about finding a place to eat, P4 about going out to dine in a restaurant, P3 about cultural activities, P5 about the town's opera, and P6 about the town's famous textile fabrics and clothing museum.

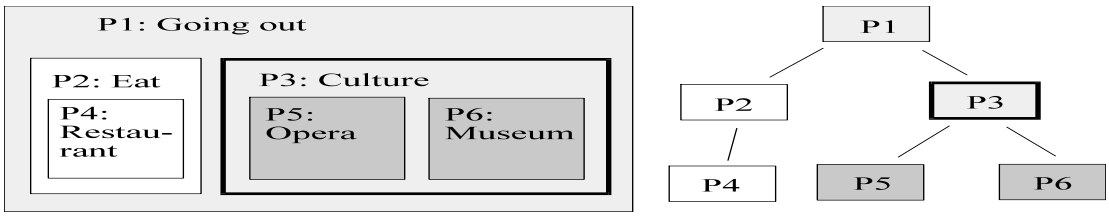


Fig. 4.1: A knowledge packet structure and the corresponding subordination tree.

Wachsmuth provides nine principles that explain how knowledge packet structures are to be built and used:
Principles concerning knowledge organization:

- *Principle of packing knowledge elements*: Knowledge elements, which belong to a specific knowledge domain, are packed together in a knowledge packet. Knowledge concerning an identifiable, more specific portion of the domain is packed in a subordinate packet (e.g., P2 in P1).
- *Principle of competitive knowledge*: Knowledge elements, which concern alternative assumptions or methods within a knowledge domain, are packed separately in "competing" packets within a superordinate packet (e.g., P5 and P6 compete in P3).
- *Principle of local consistency*: Knowledge elements belonging to one packet must be mutually consistent. Mutually inconsistent knowledge elements must be packed in competing packets.

Principles concerning static access conditions:

- *Principle of eligibility of knowledge elements*: A knowledge element is eligible by the knowledge processing system iff the packet it belongs to, or a superordinate packet thereof, is marked *accessible*. Eligible knowledge elements are called *visible*. In fig. 4.1, P3 is marked accessible (bold outline). Knowledge elements in P3 and P1 are, therefore, visible (light shading). Knowledge elements in P2, P4, P5, and P6 are not visible.
- *Principle of single access to packaged knowledge*: At a given time, only one packet may be marked accessible.
- *Principle of reachability of knowledge*: Knowledge elements belonging to (transitively) subordinate packets of the one marked accessible are *reachable* (P5 and P6 are reachable in fig. 4.1).

Principles concerning dynamic access conditions:

- *Principle of structure-dependent access*: If the processing of the current task fails, the access mark is shifted to a direct subordinate of the packet currently marked accessible. In the example, if the tourist's question cannot be answered using the knowledge contained in P1 and P3, more specific knowledge from either P5 or P6 is made accessible.
- *Principle of keyword-dependent access*: A knowledge packet can be marked accessible by means of keywords derived from the presented task, which are associated with knowledge elements in the packet.
- *Principle of persistence*: After a (sub)task is accomplished, the access mark persists as an initial access condition for the next (sub)task.

The knowledge packet structure approach is primarily intended for applications where relatively well-confined problems are posed, which can be accomplished with a given subset of knowledge elements from a stably structured knowledge base. Classification tasks are paradigmatic for this kind of task; the assumed tourist information task is of the same nature. By contrast, the approach is not intended for tasks where, over a prolonged period of time, the memory focus is likely to shift continually, or where memory elements should be regrouped dynamically (as in story understanding or in situated action).

I build my example on this basic form of knowledge packet structures. The approach has been generalized (Wachsmuth 1989) to account for overlap between packets, which gives rise to general acyclic directed graphs as subordination graphs (in contrast to tree graphs in the basic model). The principle of single access is preserved. In a more recent version (Antoniou &

Wachsmuth 1993), the subordination graph is generalized to an and/or-graph. Several packets can be marked accessible simultaneously here. The principles of competitive knowledge and local consistency are adapted to ensure that a set of simultaneously eligible knowledge elements still is consistent.

The importance of keyword dependent access is emphasized by Wachsmuth. Its power is intuitively evident. Charniak (1983) provides the example of a statement which is a syntax-free sequence of keywords ("fire match arson hotel"), but nonetheless allows the hearer's mind to focus sharply.

The most straightforward way to exploit keyword information for the access of knowledge packets would be to

- label packets by sets of keywords,
- generate a set of keywords from the input of the current task, and
- access the packet whose label set is closest to the generated set, in terms of some distance measure.

This approach, however, has its limitations. Some of them can be overcome by the technique to be described presently, which exploits the abilities of self-organizing streams to make use of contextual information. I will justify this claim at the end of the section.

For the remainder, I assume that a tourist information system is given, with a knowledge base as in fig. 4.1. Its knowledge packets contain knowledge in some classical symbolic format; its particular nature is of no further importance for the example. The user poses his or her request for counsel in spoken language. I assume that bottom-up speech processing routines can detect within the signal (not necessarily very reliably) a number of words from a pre-established list, without having to fist resort to top-down information. These words are mapped to dynamic symbols and fed into a self-organizing stream, whose sole purpose is to establish an access mark for the knowledge base. Once an access mark is set, the then visible knowledge is used for the further analysis of the preprocessed signal, and for an answer generation. When this fails, special coping strategies are invoked.

The general frame being set, I describe now the envisioned DSS mechanism in detail. The first step is to map a dynamic symbol space on the knowledge packet structure (fig. 4.2).

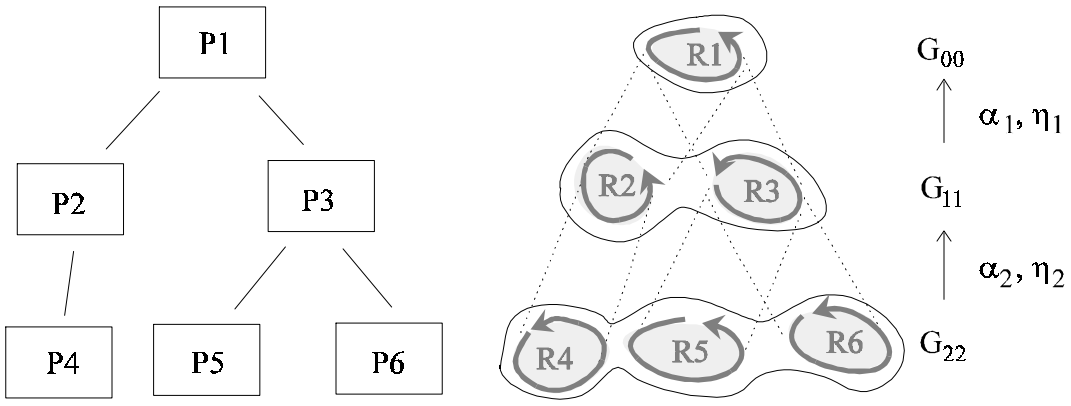


Fig. 4.2: Mapping a dynamic symbol space on a knowledge packet structure: the basic idea.

To this end, a generator G_{22} of a to-be-constructed dynamic symbol space $(G_{ij})_{i=0,...,2\ j=0,...,2}$ is invented, which has three major cyclic subgenerators for resonances R4, R5, R6, which in turn will later yield the addresses for the lowest-level packets P4, P5, P6.

G_{22} is mapped by an abstraction mapping α_2 and an epimorphism η_2 on G_{11}, such that the resonances R5 and R6 become a single resonance R3. This resonance and the α_2/η_2-image R2 of R4 correlate with P2 and P3. Finally, G_{11} becomes the single-resonance, abstract G_{00} by an application of α_1 and η_1. Figure 4.4 shows these generators in detail, and figure 4.3 provides the underlying abstraction tree.

Before I proceed, some remarks on figs. 4.3 and 4.4 may be helpful:

- The abstraction tree is conservative (cf. section 3.2), i.e., dynamic symbols from one abstraction level re-appear on all lower levels.
- The generic `going_out` plays the role of `top`.
- In classical concept subsumption hierarchies, it would be (to say the least) uncommon to have an activity concept like `going_out` subsume a building or institution concept like `restaurant`. In a similar vein, in a classical approach, one must decide whether `drink` or `dress` are meant as activity or as physical object classes. This is different in DSS, since here subsumption is not interpreted by an inclusion of sets of instances. One has the liberty to subsume a dynamic symbol *s* by a more abstract *r* whenever one finds it motivated to assume that *s* can replace *r* in an "associationistic" style of reasoning. It would be apriori permissible, e.g., to specialize `drink` by any of `wine`, `drunken`, and `swallow` - this being (in the classical view) classes of substances, properties, and activities. Whether subjects can replace, in specializing associations, `drink` by any of the other three, is an empirical question concerning subject behavior, not a model-theoretic one.

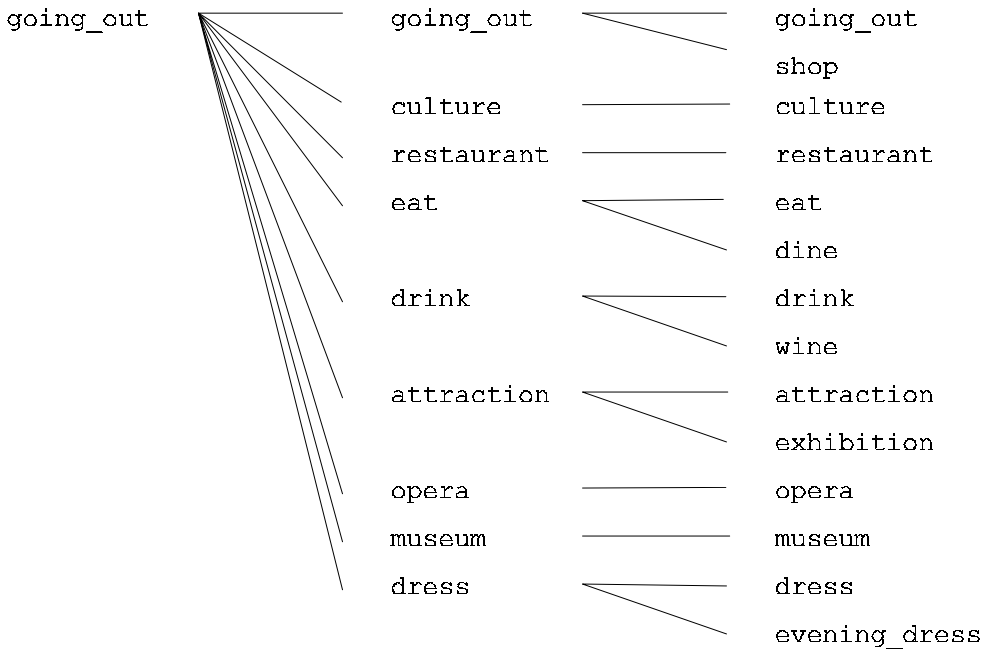


Fig. 4.3: The abstraction tree used in the example.

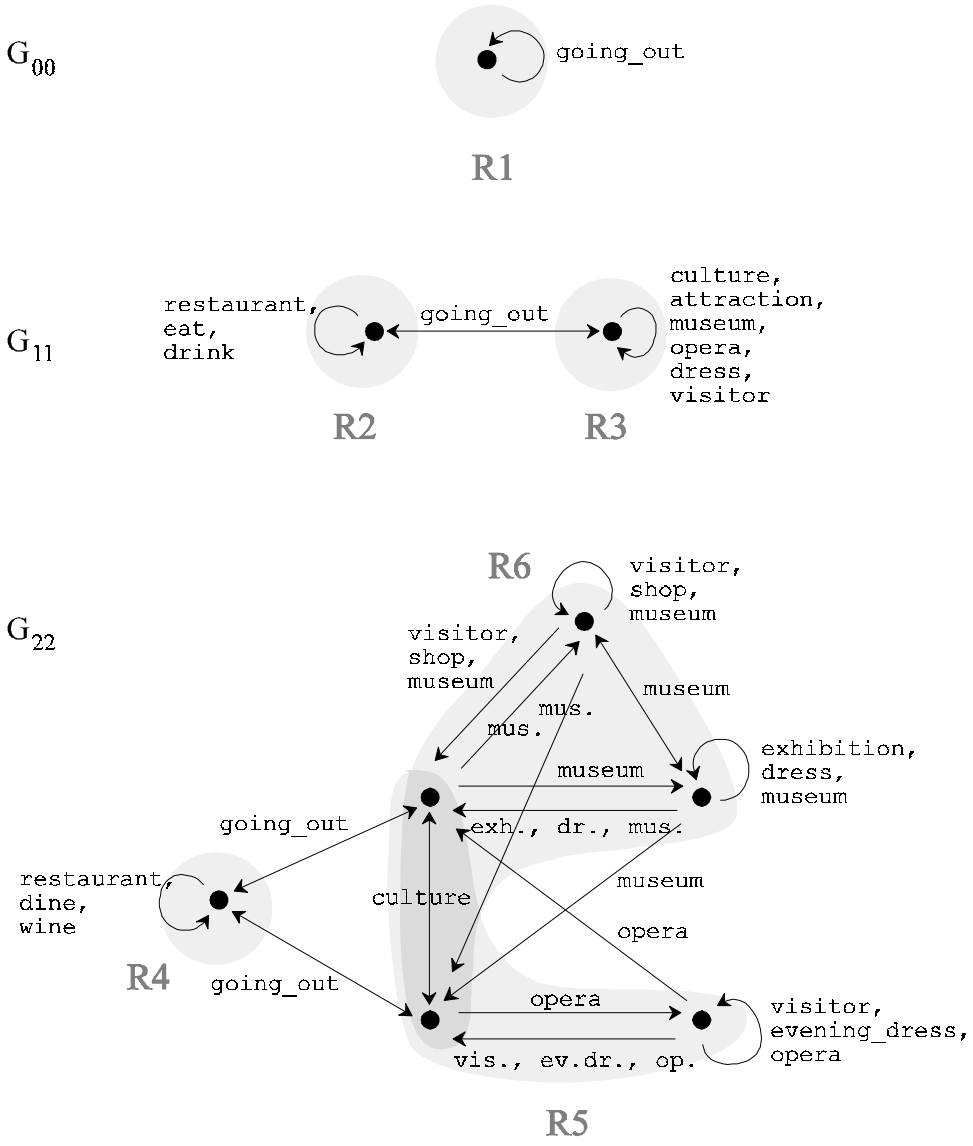


Fig. 4.4: The diagonal elements of (G_{ij}). Resonance subgenerators corresponding to knowledge packets appear in light shading.

- Dynamic symbols do not necessarily correspond one-to-one with words extracted from the external input signal. Extracted words are *mapped* to dynamic symbols, not identified with them. For instance, all of the words "food", "meal", "eat" could be mapped to `eat`.
- Fig. 4.4 is drawn economically, suppressing the conservative duplication of abstract dynamic symbols in more specialized global states. For instance, every `evening_dress` transition in G_{22} is implicitly also a `dress` and a `going_out` transition.
- The epimorphisms η_2 and η_1, which map G_{22} to G_{12} and G_{12} to G_{02} (which induces, in turn, the mappings $\alpha_2(\eta_2(G_{22})) = G_{11}$ and $\alpha_1(\eta_1(G_{11})) = G_{00}$), are not indicated in fig. 4.4. They should be obvious. The local states that belong to the R6-area in G_{22} are mapped on the local state that belongs to the R3-area in G_{11}, etc.
- The entire dynamic symbol space (G_{ij}) can be reconstructed from its diagonal, which is given in fig. 4.4, and the abstraction tree. There is, thus, no need to present the entire (G_{ij}) explicitly.

- (G_{ij}) features both monotonic and nonmonotonic specialization effects (cf. 3.2). For instance, `museum` acquires the all-new continuation `shop` in the step from G_{11} to G_{22}, and the continuation `eat` of `restaurant` in G_{11} is specialized to `dine`. These are monotonic specialization phenomena. By contrast, the continuation `attraction` of `opera` in G_{11} is nonmonotonically deleted in G_{22}.
- The shaded "resonance areas" R1, R2 etc. in fig. 4.4 are substructures of the G_{ii}, which generate coherent sublanguages of the languages $C_{G_{ii}}$. I.e., they are generators of resonances in the sense of definition 55 (section 3.4). These resonances have many characters. For instance, the resonance generated by R6 has, amongst others, the characters `shop` and `dress exhibition` (the town's museum is a clothing museum!), and the resonance generated by R5 has, among others, the character `dress visitor`. In the example, characters of length 1 abound; this is due to its unrealistic simplicity, where many dynamic symbols occur in only one resonance.

Stepping back from the particulars of the example, the following points characterize, more generally, how a dynamic symbol space is constructed in correspondence with a knowledge packet structure:

- The n levels of a knowledge packet structure correspond to the diagonal elements in an n×n dynamic symbol space.
- Each G_{ii} contains cyclic substructures which correspond to the knowledge packets in level i. These substructures need not be disjoint. In Fig. 4.4., there is an overlap between R5 and R6 (slightly darker shading).
- G_{00} is a trivial single-transition generator. The trivial resonance generated by G_{00} corresponds to the topmost knowledge packet.
- When a knowledge packet P in level i has direct subordinates P_1, ..., P_k in the more specialized level i+1, the cyclic substructure R of G_{ii}, which corresponds to P, is symmetry-broken (and its labels are specialized) into different cyclic substructures R_1, ..., R_k in G_{i+1i+1}.
- A cyclic substructure R, which corresponds to a knowledge packet P, is a generator of associations that are indicative for the domain covered by P. What "indicative" means, and how (G_{ij}) is in practice constructed (which requires an elicitation strategy for associative knowledge), must be left open here. Realistic (G_{ij}) will certainly be much larger than the example depicted in fig. 4.4.
- The dynamic symbol space can be constructed with great liberty. Neither the exact set of dynamic symbols used, nor its detailed structure are crucial. This liberty is typical for computational approaches that rely on self-organization. It contrasts with the exactitude demanded by logic-style knowledge representation.

Before I turn to dynamics, the definition of characters of resonances must be slightly specified:

Definition 66 (specifies definition 56): Let (G_{ij}) be a dynamic symbol space, ***r*** a G_{kl}-association, and $R \subseteq C_{G_{kl}}$ a resonance in $C_{G_{kl}}$. Then, ***r*** is a G_{kl}-**character** of R :iff the G_{kl}-interpretation (cf. def. 37) ***r***' of ***r*** is a character of R in $C_{G_{kl}}$.

The reason for considering characters is that the resonances they characterize in turn indicate knowledge packets. The next definition introduces a convenient terminology:

Convention 67: When ***r*** is a G_{kk}-character of a resonance R, which in turn corresponds to a knowledge packet P in the knowledge packet structure, ***r*** is said to **indicate** P.

Now I turn to dynamics. Assume that a sequence of dynamic symbols has been derived from the user's input signal. This sequence is fed into a self-organizing stream. Then, the general strategy is as follows. Throughout its history, the stream develops associations that indicate knowledge packets. Let the stream run through a history, and determine, for each configuration, the "best" knowledge packet that is indicated. Return, for each configuration, an access mark for this packet. During the stream's history, this mark is likely to become successively more special as the stream self-organizes. In parallel to the stream's activity, the classical calling system can start working with the preliminary access marks.

I shall now describe this procedure in some more detail.

As a pre-start preparation, the original dynamic symbol sequence is fed into a single `top`-transition configuration (in the example, `top` is replaced by a `going_out`) (fig. 4.5a). A series of micro-inputs leads to a starting configuration C_0 that consists of a single, large loop (fig. 4.5b).

Every word derivable in C_0 (and in every configuration to come) is a G_{00}-association, and it is also a character of the trivial resonance generated by G_{00}. I.e., from the outset, the topmost knowledge packet is indicated. It may happen that more specific knowledge packets are indicated, too. In the example, P3 is indicated by, e.g., `opera`, `museum`, or `culture`. Also, P5 and P6 are indicated (amongst others, by `opera` and `museum`). Determine, among all indicated packets, the unique packet P that satisfies

(i) no competing packet of P is also indicated, and
(ii) P does not have a subordinate P' satisfying (i).

Call P the packet that is **fixed** by C_0. Return an access mark for P to the calling system. This is the "best" mark that C_0 can yield. In the example, P3 is fixed by C_0.

Let the stream run a history. Each time some configuration fixes a packet which is subordinate to the one fixed before, a new, more specific access mark is returned. In the example, the configuration shown in fig. 4.5c fixes P6, and the one of fig. 4.5d fixes P5. Both are possible configurations in a history starting with C_0, which reflects that the initial sequence of dynamic symbols is highly ambiguous concerning the distinction between P5 and P6.

Stop when the calling system (which works in parallel to the stream) has successfully finished its task, or when the history reaches a state where a continuation cannot yield a more specific access mark. The latter can have at least three reasons:

- A packet is fixed that belongs to the most specific level in the knowledge packet hierarchy (fig. 4.5c, d).
- The stream loses so many dynamic symbols by micro-outputs, that the residual material cannot give rise to an improvement on the last returned access mark. In the extreme, the stream can completely degenerate (fig. 4.5e).
- Resonances form in the stream which indicate competing packets, but which do not mutually compete in the sense described in section 3.3. When a version of micro-dynamics without micro-outputs is chosen, these resonances are perfectly stable. As a consequence, none of the two competing packets can ever become fixed.

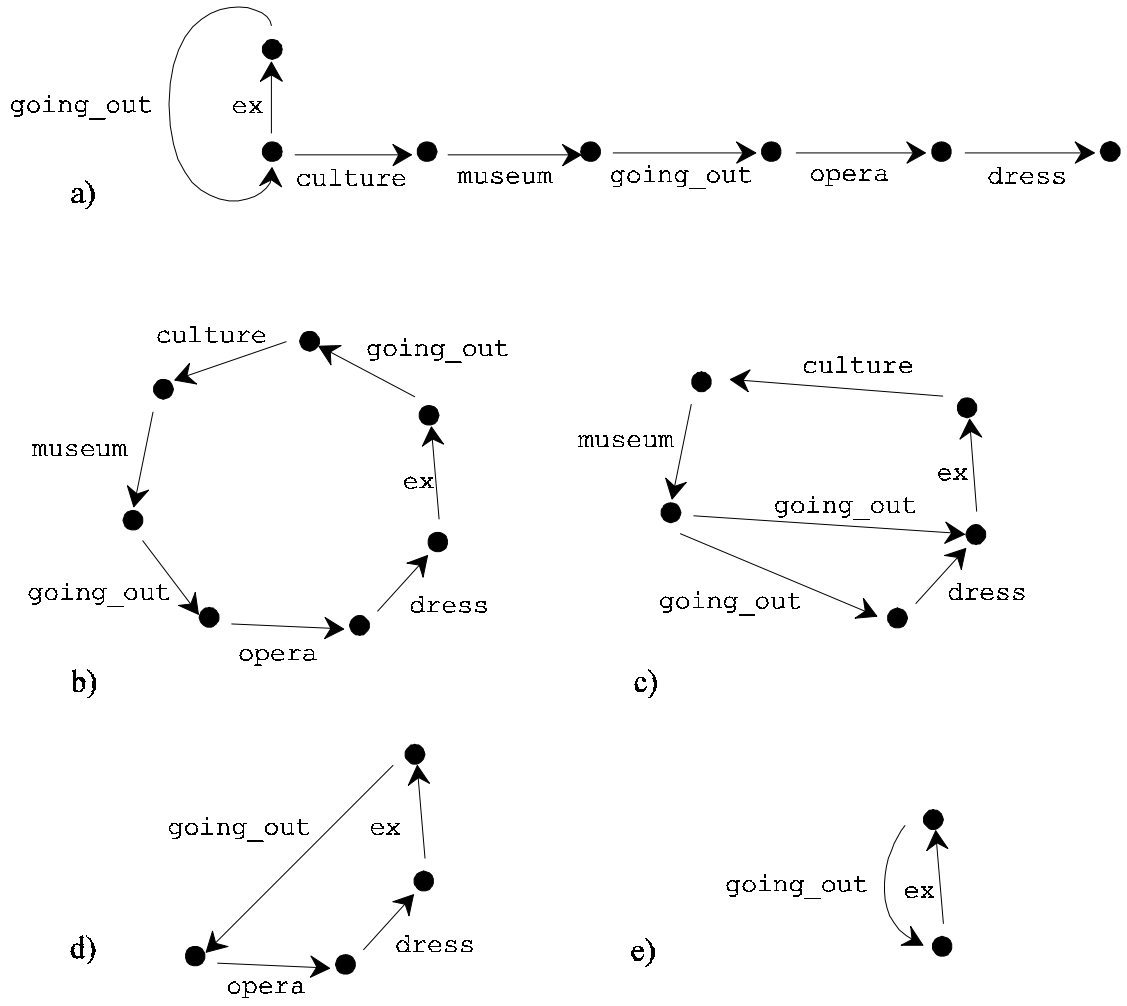


Fig. 4.5: The self-organizing stream at work.

During the entire process, the stream is in the most specific global state (G_{22} in the example). This setting is fixed because any more abstract state (i.e., a global state not in the rightmost column of the dynamic symbol space) would typically destroy information contained in the input band by abstracting dynamic symbols, and because more symmetric global states (i.e., higher up in the the dynamic symbol space) would impair microchange activity. In the extreme, when a global state from the top row of (G_{ij}) would be chosen, no non-empty microchange would be possible at all. Besides such particular reasons, taking the most specific global state is generally motivated by the ultimate purpose of the stream, i.e., developing packet-fixing characters of the most specific kind possible.

Besides fixing the global state, there are many other "knobs to turn":

- the dynamics can be made more or less "dissipative" (cf. 3.4) by adding further `top`-input (here: `going_out`-input),
- the tpye of microchange can be varied (cf. 3.3), and
- the microchange/micro-input/micro-output ratios can be varied.

Although abstract considerations can help with understanding the effects of these parameters, their influence will depend on the particulars of (G_{ij}), and on the kind of user input that typically occurs. Optimizing them requires experimentation.

Iclaimed at the beginning that an access control with a self-organizing stream is superior to straightforwardly labeling knowledge packets by sets of keywords. This claim can now be justified by the following observations:

- When packets are labeled by simple sets of keywords, these sets will often have nonempty intersections. Then, when a set of keywords is derived from the user's input, which is a subset of several of the labeling sets, a set-comparison based algorithm cannot decide between the corresponding packets. Contextual information can upgrade the indicative value of keywords in such cases. For instance, `visitor` and `dress` each occur both in R5 and R6. Thus, the *set* {`visitor`, `dress`} does not help to distinguish between the two resonances (i.e., the packets P5 and P6). However, the *sequence* `visitor dress` is indicative for P5 (and it will be specialized to `visitor evening_dress` in the stream).
- The user's input may lead to a set of dynamic symbols consisting of two subsets S and S', such that associations generated in S and S' are indicative for two specialized, but competing knowledge packets P and P'. Then, the access mark yielded by the stream mechanism can fix P, P', or a superordinate packet. All outcomes are principally possible. So far, this is not better than what could be achieved with some mechanism based on keyword sets. However, when the words in S are more apt to form associations than the words from S', i.e., the closer they are related, the more likely they will out-compete the words from S' in the stream. The dynamic symbol space used in the example is too simple to give rise to such an effect, since a more elaborate internal structure of the resonance subgenerators is required for it to become manifest. For a positive example, assume that a more differentiated dynamic symbol space is used by the tourist information system, and that there is an exhibition of theatre costumes in the museum. Consider the sequence `attraction admission_fee actor costume evening`. The first four of these dynamic symbols may occur in the resonance that indicates the museum packet; all of them may occur in the resonance indicative for the opera packet. But, only in the first case can they give rise to a long and indicative association (which might look something like `admission_fee attraction actor dress`); in the case of the opera resonance, the three subsets {`admission_fee`}, {`actor`, `costume`}, and {`evening`}, are too far away from each other to make a connection. Therefore, the first reading is due to come out, although it is made from a proper subset of the words that occur in the opera resonance.
- In set-based packet access, more general packets will typically be labeled by keywords which are more abstract than those that label more specific packets. Then, when a set of abstract keywords is entered, an unspecific packet is necessarily accessed. But, mutual contextual constraints can sometimes lead to a specialization of dynamic symbols in a self-organizing stream. A more specific packet can be accessed. The `visitor dress` example can be interpreted in this sense: the abstract `dress`, which is indicative only for P3, becomes specialized to `evening_dress` in the context of visitor. An occurence of `evening_dress` allows to fix P5.

Another technique for exploiting keywords for the access of knowledge packets suggests itself besides simple set labeling and DSS streams, namely, using a feed-forward neural network. Gängler et al. (1992) report on first experiments carried out with several types of such networks. The network's input nodes correspond to keywords, its output nodes to knowledge

packets. Training is done in a supervised mode, utilizing a training set of correct keyword-set/knowledge-packet pairings. Such an approach is in principle superior to a statistical evaluation of keyword sets, since nonlinear interdependencies between keywords within such sets can be exploited (the preliminary experiments by Gängler et al. do not reveal such effects due to a relatively small number of keywords and training examples). These interdependencies are a special kind of contextual influence. However, the input to the network still is a set; therefore, information contained in sequential order is lost. The discrimination principles described for DSS streams in the first two of the above points cannot be captured by neural networks of the examined types.

The stream-based technique, so far, describes the case when all goes well. But, what happens when the system cannot complete its task of user question analysis and answer generation? There are two cases.

First, it can happen that although a partial analysis of the user's request is achieved, the analysis must be refined. This will typically require the access of a more special knowledge packet than the one fixed so far. To this end, the following strategies can be invoked:

- The self-organizing stream, which has run dead, can be "reanimated", hoping that it can be made to yield a more specific access mark yet. Measures taken towards this end include a repetition of the original input; "energizing" the stream by `top`-input; or techniques of the simulated annealing kind, i.e., shifting the global state away from the most specific state for some time. The last two techniques seem to be particularly recommendable when the impasse is of the many-competing-resonance type.
- The system can request a further specification from the user. This leads to additional input into the stream, which might lead to a more specific access mark.
- The stream mechanism can be bypassed with a fall-back strategy, where the system accesses a more specific knowledge packet either at random, or using candidates that are already indicated by the stream. This corresponds to Wachsmuth's original principles of structure-dependent access and of persistence.

The second kind of failure occurs when not even a partial success has been achieved in the first run. Then, probably the most reasonable thing to do is to start again from scratch, discarding the access mark from the first attempt completely. Trying to retain it (according to the principle of persistence) might be harmful, since the complete failure indicates that a severe misunderstanding has occured in the first run; therefore, the access mark might be outrightly wrong, not merely too unspecific.

In concluding, I point out that the task sketched in this section essentially is a single-run classification. This kind of task does not fully exploit the powers of self-organizing streams, which lie in the continual processing of an essentially infinite input stream. The presented technique is not restricted to the simple case of a finite one-time input. When a non-terminating sequence of keywords is fed into the stream, it will yield a sequence of access marks for a dynamic memory access, such that the focus continually shifts. This would be required, e.g., for story understanding or extended dialogues with a user. But even in its basic version, the example demonstrates that DSS techniques appear to be suited for the non-trivial task of memory access, by combining symbol processing with self-organization.

Summary of section 4

- An application of a self-organizing stream in an otherwise classical AI information system is proposed.
- The system generates proposals of places where a tourist can go in a town. For the analysis of the user's request and the subsequent answer generation, the system makes use of a classical knowledge base. The contribution provided by the self-organizing stream consists in computing a focus for accessing the knowledge base.
- More specifically, the knowledge base is assumed to be organized in a knowledge packet structure. This is an access hierarchy of modules, which are tree-ordered according to their specificity.
- A dynamic symbol space is constructed, whose symmetry breaking and abstraction structure mirrors the packet hierarchy. Each packet is labeled by a resonance, such that packet subordination corresponds to symmetry breaking and specialization of the according resonances.
- A sequence of dynamic symbols is generated from the user's input. The sequence is fed into a self-organizing stream. Certain associations that can appear in the stream are indicative for resonances, which in turn indicate knowledge packets.
- During the stream's history, a sequence of access marks for the knowledge packet structure is produced, of increasing specificity, until a "best" focus is settled. The calling system can work in parallel, focussing increasingly more specific knowledge.
- Since contextual and sequential interdependencies between keywords can be exploited, the presented technique is in some aspects superior to a straightforward labeling of knowledge packets by sets of keywords, and to certain connectionist techniques.

5 Representing conceptual knowledge

The DSS formalism is intended to cover all levels on the periphery-centre axis. However, the conceptual level deserves a special treatment. It is the level which classical AI and cognitive science focus upon; it is the level which is (deceptively) easily accessible to introspection; and it is of singular importance for most AI applications. In spite of the central importance of conceptual knowledge, some basic issues concerning the nature and the representation of concepts are not settled (surveys: of classical aspects, in Smith & Medin 1981; of current symbolic representation techniques, in Lehmann 1992; of psychological and physiological findings, in Treisman 1986; of concepts in linguistics, in Lakoff 1987, Mehl 1992; general, in Wrobel 1991). In this section, I consider some of the relevant questions, comparing the answers provided by classical AI with those proposed by DSS.

More specifically, I want to expand on the issues of concepts vs. symbols; concepts vs. attributes; variability and contextual influences; and conceptual cycles. Some other issues, which I have treated previously in some detail, will not be reconsidered. This concerns the nature of abstraction (cf. 2.6, 3.2); nonmonotonic inheritance (cf. 3.2), and the nature of "instances" of a concept (cf. 3.3).

Concepts vs. symbols

It is not a settled question how concepts and symbols relate to each other. The standard answer given by logic-oriented AI is to define a concept by its extension, i.e., as a *class* of individuals, and to view symbols as conventional and essentially arbitrary signifiers in some *language*, in which a part of reality can be described. Concepts and symbols are, thus, entities of a fundamentally different nature. This perspective is not unchallenged. At least two objections are raised from within AI:

- Humans can reason about words (i.e., symbols) in the same way as about other objects. I.e., there seem to exist *word concepts*. Consequentially, Langacker's (1987) "cognitive grammar" treats words and object concepts as the same kind of mental entity, formalizing them in a unified fashion.
- The symbol grounding critique (Harnad 1990, Chalmers 1990) claims that symbols are not mere conventional tokens. Rather, when a human uses a symbol, it is intrinsically "meaningful" in an internalistic, experiential sense of "meaning". For instance, when a human thinks "red", there must be an experience of redness involved.

I would like to hint at yet another, quite fundamental objection. In the standard proof of the completeness of first-order logic (compare Ebbinghaus, Flum & Thomas 1978), a model is constructed for a given formula (the Henkin model), which is essentially composed of fragments of the formula itself. Thus, a *language*, and the *objects* about which the language tells, are effectively identified. Now, the great importance of the completeness theorem lies in the fact that it closely links syntactics (i.e., symbols) with semantics (i.e., models). It seemsthat this cannot be done other than by identifying the two sides. To put this in intuitive terms,

symbols and concepts must be treated as the same thing, in order to arrive at an interesting account of what can be expressed by symbols.

In the perspective of dynamic symbol structures, neither are symbols *arbitrary* signifiers, nor do they, in fact, *signify* externally at all in the first place. They are observables that play a dynamic role in an agent's information processing; and that is all that constitutes them.

First, they are not *arbitrary*, since in a complex dynamic system, an observable entity cannot be replaced by another one without crucially changing the system's behavior. For instance, in a recurrent neural network that responds to stimuli by attractor formation (as in Yao & Freeman's model of the olfactory bulb, cf. section 2.2), one of the attractors cannot be "exchanged" for a new one, since each attractor's dynamics is interwoven with the dynamics of all other such attractors.

Of course, in a formal account of a dynamic symbol structure (e.g., in a DSS associety), the formal symbol that stands for a dynamic symbol is arbitrary and can be changed. For instance, `apple` could be exchanged for `elppa`. But, this does not concern the dynamic symbols, as they exist, and can be observed, in an agent; it merely concerns the formal symbols that represent the dynamic symbol in the theory. This contrasts with the physical symbol systems paradigm, where it is assumed that symbols are principally arbitrary even within the agent.

Second, dynamic symbols do not intrinsically *signify* at all. Two correlates of signification are reconstructed in dynamic symbol structures, namely, external and internal reference. Both are considered as empirical phenomena. Detecting the first boils down to verifying a reliable causal connection between the presence of an external object and the occurence of a dynamic symbol, which is a major undertaking. Internal signification is explained in terms of emergence/grounding links between adjacent processing levels. Importantly, neither case is a prerequisite for a dynamic symbol to occur and to have causal influences within the system: in an associety, a dynamic symbol can appear on any level without grounding in the next lower one, and without being connected to any external circumstance. Thus, the classical interpretation of symbols as referential entities plays no substantial role in dynamic symbol structures.

So much for a recapitulation of the differences between classical and dynamic symbols. I turn now to the question of how *concepts* should be reconstructed in DSS. I propose to define as a *dynamic concept* a dynamic symbol on the conceptual level, *plus* what it grounds in, transitively through the lower levels, *plus* what it can associate with within the conceptual level. More specifically:

Definition 68 (informal): Let an emergence/grounding column be given in an associety, as in fig. 3.19, which spans the entire periphery-centre axis form the sensomotoric interface level to a topmost, conceptual level. Let C_1, C_2, ..., C_n be the sequence of the corresponding coherent languages, where C_1 belongs to the peripheric level and C_n to the conceptual level. Let upper indices of dynamic symbols and associations indicate the level from which the symbol comes, e.g., c^3 belongs to C_3.

(i) A **dynamic concept** c^* is

- a dynamic symbol c^n from the dynamic symbol space underlying the conceptual-level stream,
- plus the set of characters $\{c^{n-1}{}_1...c^{n-1}{}_k \mid k \geq 1, c^n$ grounds in $c^{n-1}{}_1...c^{n-1}{}_k\}$ it grounds in, plus the set of sequences of characters $\boldsymbol{c}^{n-2}{}_1...\boldsymbol{c}^{n-2}{}_k$ over which these characters $c^{n-1}{}_1...c^{n-1}{}_k$ emerge (cf. definition 61), etc., down through all levels,
- plus the set of all conceptual-level associations in which c occurs.

(ii) The dynamic symbol c^k is the **dynamic concept symbol** of the dynamic concept.
(iii) An **occurence** of a dynamic concept c^* in a history of the associety is an occurence of c^n in a configuration of the conceptual-level stream (cf. definition 44), plus the derivations of characters $c^{n-1}{}_1 \ldots c^{n-1}{}_k$ to which c^n is actually χ-linked (cf. definition 65), plus the derivations of the characters $\boldsymbol{c}^{n-2}{}_i$ to which the $c^{n-1}{}_i$ a are actually χ-linked, etc., plus the set of derivations of associations within the conceptual-level configuration of which the occurence of *c* is a part.

The implications of this definition will become clear in the sequel, when several of its aspects are examined in more detail. It accounts for the symbol grounding critique, since an irreducible aspect of lower-level "experience" is ascertained by tracing a dynamic concept from the conceptual level transitively through all grounding levels.

In sum, both symbols and concepts are considered in the dynamic symbol perspective as very much the same kind of entity. The difference is that a dynamic symbol is observed within a single level, whereas a dynamic concept is a rich, multi-level phenomenon, which includes not only a conceptual-level dynamic symbol, but also the information that is connected to such a dynamic symbol via associations and grounding. The aforementioned objections raised against the classical symbol-concept distinction do not affect this conception of symbols and concepts.

Concepts vs. attributes

Concepts are almost always defined in terms of "attributes". However, the empirical and logical status of attributes is problematic, which is reflected in the existence of many other, closely related notions ("features", "relations", "properties", "slots") and a wealth of concept representation formalisms. I shall examine two problematic aspects.

The grow-and-shrink problem. A question that is simple to state and hard to answer: Are attributes concepts? More often than not, attributes are treated as something more "basic" than concepts. The paradigmatic case is perceptive (in particular visual) features. For instance, "roundness" can be a feature of a ball concept, or "red" a feature of blood. This conception fits well with a level architecture for pattern classification devices, where features are processed at more peripheric levels than concepts. However, the issue is not so clear as it seems: isn't "red" also a concept? Smith and Medin (1981) acknowledge this problem. Their example is the concept of the letter "E". The small vertical bar in this letter is, on the one hand, a typical perceptual feature - but then, on the other hand, one can deliberately focus on it, reason about it, etc., just as if it were a concept. Features seem to be able to "grow" into concepts. Conversely, I would like to argue that concepts can shrink to features. For instance, the Statue of Liberty comes close to being a perceptual feature of my concept of New York.

The class-relation problem. In many logic-oriented semantic network formalisms (KL-ONE-like languages in particular), attributes are treated as being of the same logical type as concepts. Both are first-order predicates. The only difference lies in their arity: concepts are unary predicates (or "classes"), attributes n-ary (typically, binary) relations. Now, although concepts and attributes here are "ontologically" of the same basic type, the stringency of logical syntax and inference mechanisms requires that classes and relations be cleanly kept

apart. Once `Mother` is introduced as a binary relation in a KL-ONE network, it can no longer serve as a class node. This compels the network designer to settle for one of the alternatives. Kobsa (1991) describes how this cause a handicap for constructing intuitively appealing semantic networks. A related problem lies in the technical difficulties to work with relations of higher arity than two (Brachman et al. 1991). One would like to have such arities in the first place, a commodity not provided for in customary KL-ONE-like formalisms. Second, one might also desire relations with an unspecified arity, for instance, in order to formalize case frames with optional parameters. An elegant way out of all these problems would be to use a logic featuring what Scott (1978) calls "multi-relations", i.e., relations that accept any number of arguments. Something like this is not done, however. The reason is, one may suppose, that such added flexibility would intolerably increase the computational complexity of classification algorithms, or even render them inapplicable.

To summarize, in treating concepts as something different from attributes, one faces the problem that there are smooth transitions between the two; and in treating them as essentially being of the same type, one runs into trouble as these entities sometimes appear in a unary fashion (behaving more like concepts), sometimes in multi-ary fashion (behaving like relations), which raises technical difficulties.

Before I can examine these issues from a dynamic symbol perspective, I must supply a notion of attributes in DSS. I find it intuitive to consider the following phenomena as attributes:

- When *c** is a dynamic concept, and *cd* is an association, then the dynamic concept *d* is an **associative attribute** of *c**. This corresponds roughly to binary relations in classical representation formalisms like KL-ONE. In both cases, the attribute relates two conceptual entities. For instance, the classical relation statement `Mother(human_2, woman_3)` would reappear in a DSS configuration as the occurance of an association `human mother woman`, where `mother` is an associative attribute of `human*` (remember that the DSS counterpart of instances are occurences, cf. section 3.3).
- When *c** is a dynamic concept, and it grounds in a character $c_1...c_k$, then each of the c_i is a **grounding attribute** of *c**. Grounding attributes come in two subspecies, according to whether $c_1...c_k$ comes from the same (i.e., conceptual) level as *c*, or from a properly lower level. When $c_1...c_k$ comes from a sub-conceptual level, then a grounding attribute corresponds to what is usually called a perceptual feature (although one should not forget similar lower-level features connected with motor action). I shall call these attributes **experiential attributes**, since they are connected with the situated experience of a dynamic concept. When $c_1...c_k$ comes from the same (i.e., conceptual) level as *c*, a grounding attribute c_i plays essentially a similar role as a words that occurs in the explanation of an encyclopædia entry. I shall call them, therefore, **explanatory attributes**.

Explanatory attributes and associative attributes of a concept coincide to some extent. For instance, the concept `fall-of-Man*` might ground, within the conceptual level, in characters like `Eve apple` or `evil serpent`. The dynamic symbols that appear in these characters are also associatively accessible from `fall-of-Man`: `fall-of-Man Eve` or `fall-of-Man serpent` are plausible associations. However, some other explanatory attributes would be unlikely associative attributes, and vice versa. For instance, `Eve hand apple` might be a character in which `fall-of-Man` grounds, making `hand` an explanatory attribute (albeit one that needs a more comprehensive sequence around it to become characteristic); but `fall-of-Man hand` is hardly a sensible association. Conversely, `paint fall-of-Man Botticelli` is a good association, but `Botticelli` should not occur in an "explanation" of `fall-of-Man`. The intuitive difference between explanatory and

associative attributes is that the former concern a concept's "substance", whereas the latter are connected with the dynamic "behavior" of a concept in contexts.

Grounding attributes provide a satisfying answer to the grow-and-shrink problem, when a level topology like the one in fig. 2.1c is assumed, i.e., when the conceptual level is not disjoint from the level of perceptual (or motor action) features, but blends into it (Fig. 3.19 would have to be redrawn, by replacing streams 4 and 5 - which correspond to the perceptual feature level and the conceptual level - by a single new one, of the kind shown in fig. 2.1c). Then, a grounding attribute is apt to play both the role of an explanatory and an experiential attribute. Since explanatory attributes partially correspond to associative attributes, the latter also aquire a certain degree of level-indefiniteness in such topologies.

Grounding attributes also shed light on the issue of image-like, analogical vs. language-like, propositional representations (cf. Sloman 1975). Experiential attributes (or lower-level characters of a dynamic concept symbol in general) can be considered as analogical components of a concept, due to their nearness to perception and action. Associative and explanatory attributes (or associations and conceptual-level characters in general), by contrast, are of the same non-analogical quality as in any other logic-oriented representation format that captures a "language of mind". The possibility of blending one type of attributes smoothly into the other bridges the gap between analogical and language-like representations, which now appear as two extremes on a continuum.

Every conceptual-level dynamic symbol *c* plays a dual role as a concept *c** and as an associative attribute *c* (of all *b** for which *bc* is an association). Therefore, the class-relation problem does not occur in DSS. In fact, one of my primary motives, which guided the development of DSS, was to find a formal account of informational entities with a dual concept-relation nature (Jaeger 1991, 1992). The underlying intuition is that *using* a concept in reasoning can always be interpreted in terms of a meaningful connection, which the concept establishes between other concepts. A concept simply cannot be used in isolation. This is reflected in the connectivity structure of coherencies in dynamic symbol spaces, and in the dominant role of associations in microdynamics.

A reasonable notion of "arity" for associative attributes can be introduced in many ways. I have compared these attributes to binary relations above, likening an association *abc* to a relational statement $B(a, c)$. Other conceptions are equally well justified, e.g., *ab* as a unary predication $B(a)$, or *abcd* as a multi-relational statement, which comprises $B(a)$, $B(a, c)$, $B(a, c, d)$, etc. Generally, I would prefer not to interpret DSS constructs in terms of arity at all.

As a side remark, I mention that sequential associations, as used in this thesis, can be generalized to tree-like associations. Instead of associations *abcdef*, one would have nested terms like $a(b, c, d(e, f))$. Associations of this kind yield a notion of arity that comes close to the classical one: in $a(b, c, d(e, f))$, *a* occurs in a ternary, *b* in a 0-ary, *d* in a binary fashion. A DSS version that uses tree-like associations behaves essentially like the simpler version used here. This is not surprising, considering that (a) coherent languages are regular, (b) that regular languages can be straightforwardly generalized to regular *tree languages*, and (c) that regular tree languages exhibit virtually all the nice properties of linear regular languages (standard reference: Gécsec & Steinby 1984). Earlier versions of DSS were formulated in this more general fashion. I have abandoned it since, mainly because of the notational inconveniences of tree languages.

In sum, the grow-and-shrink problem does not arise in DSS due to the topological liberties afforded by associety level structures, and the class-relation problem vanishes since dynamic symbols are basically dualistic entities.

Vagueness, variability, and context sensitivity

Concepts used by humans vary in several dimensions, on several time scales, and for many reasons. I can only pick out a few examples of relevant research:

- Bartlett's (1932) famous monograph on remembering describes, among other things, how the recall of a story changes on a long-term time scale (days to years). The original story has many "exotic" aspects, which do not blend easily with the subject's cultural background and world knowledge. A basic finding is that the subject's conviction of what the story actually tells shifts towards "making more sense" over time. This indicates that human concepts are not separately acquired, stored, and recalled, but that they interact in memory, maximising mental coherency. The concepts of a human are part of an evolving system; they change together with the system.
- Barsalou (1987) reports empirical findings concerning the "graded" structure of concepts. In contrast to what is required by the classical philosophical and logical understanding of concepts, they turn out not to have the nature of a clear-cut set of instances in humans. Subjects declare, e.g., that a robin is a more "typical" bird than an ostrich. Typicality ratings vary accross subjects, with the context of concept presentation, and across time within a subject. In a theoretical reconstruction of these phenomena, Barsalou (1989) assumes that a concept is constituted by three different kinds of features: stable, mandatory, context-insensitive ones; those dependent on the current context; and others dependent on the subject's prior reasoning history (time scale: one to several days).
- Seidenberg et al. (1982) carry out subtle experiments in order to reveal the dynamics of concept activation in humans on a very short time scale (tenths of seconds). They find that a concept is not "loaded" as a unit, but rather rapidly "assembled" from different informational entities, and by different mechanisms, in a complex process. Similar to Barsalou, they distinguish context-sensitive and mandatory, context-insensitive subprocesses.

The general picture that emerges from such observations is that empirical concepts

- are *vague* rather than well-defined,
- that they are *variable* in time rather than stable, and
- that they intrinsically interact with their *context* rather than being stand-alone units.

Obviously, these points are interrelated. Taken together, they amount to a hard problem for any formal AI reconstruction of concepts: how can a concept be *fixed* at all, when it is a so dynamic, flexible, and elusive kind of entity?

In symbolic AI, concepts are usually defined via attributes. The problem lies in the question of how a concept can be *determined* by attributes when attributes are apt to *vary* with circumstances. Many answers have been proposed. I mention a few, without going into detail:

- An "ideal" account of concepts is given, where attributes provide necessary and sufficient conditions for concept definitions. This is, e.g., the strategy of KL-ONE-like concept representation schemes. Such schemes are fully appropriate for many technical applications, but they fall short of the empirically observed flexibility of concepts in humans.
- Two types of attributes are assumed, namely, definitory and contingent ones. The first are connected to a concept in a mandatory fashion, and they are sufficient for a characterization. Contingent attributes appear and disappear according to circumstances. This is apparently a better approximation to natural concepts than the ideal account mentioned before. However, for every natural concept one can find cases where any particular of its attributes is disabled. The assumption of truly definitory "core" attributes is empirically false. This is known as the "empty intersection" problem.
- A concept is defined as a probability distribution over attributes (e.g., Smith & Medin 1981). "Core" attributes get a high, "marginal" attributes a low probability value. Then, for a given set of attributes, a likeliness can be computed of which concept is actually "meant". This approach resolves the conflict between vagueness and definition by attributes, and the empty intersection problem. However, it falls short of context sensitivity. In a given context, a relatively unlikely combination of attributes can, in fact, be highly indicative for a concept *a*, although it is more indicative for some other concept *b*, when context information is not taken into account. A classification mechanism based on a probability distribution is apt to misclassify concepts in untypical contexts, because the probability distribution integrates over *all* occurences of a concept, i.e., over all contexts.
- Finally, I mention connectionist representations of concepts in localist networks (e.g., Waltz & Pollack 1985, Smolensky 1986, Bookman 1988). In this often-used type of recurrent networks, concepts and attributes are represented by labeled nodes. When a collection of attribute and concept nodes is activated in a clamped fashion, the network returns activation values for nodes of goal concepts after a phase of equilibration. This type of approach accounts for the vagueness of concepts. Furthermore, contextual information (represented by the initially activated concept nodes) is also fully effective. The only apparent shortcoming lies in the single-run type of dynamics. A classification is a "single-shot" event, leading from an initial activation to a result. A continuous conceptual reasoning dynamics, where a concept runs through several states over a period of time, is not captured. However, it should be possible to overcome this restriction by elaborations of the basic technique.

Dynamic concepts are vague, variable, and context sensitive, justifying the DSS approach in light of these generally hard-to-match constraints for concept representation:

- *Vagueness* results from both associative and grounding attributes, since in different occurences of a dynamic concept, different such attributes are typically present. No attributes must occur by necessity; therefore, the empty intersection problem does not arise. Furthermore, no apriori probability constraints are imposed on attribute occurences; therefore, contextual influence is not disabled. In spite of this great liberty, concepts are well-defined. The reason this is possible essentially rests in the fact that concepts are not defined locally in a one-by-one fashion, but within a complete, complex system: grounding attributes rely on characters, which characterize resonances on the background of an entire dynamic symbol space. Only associative attributes are, to a certain degree, "local" attributes.
- *Variability* also concerns both kinds of attributes. In a history, both types of attributes are apt to appear, stay for a while, and disappear again. To my knowledge, a similar dynamics has not yet been realized by neural networks that process conceptual knowledge, but I believe that it is in principle possible (cf. the remarks at the end of section 3.4).

- *Context sensitivity* is one of the most basic features of DSS information processing. Within the conceptual level, an associative attribute *d* of a concept *c** can be filtered out when *c* arrives in a context $b_1...b_k$: $b_1...b_k cd$ may be not an association, although $b_1...b_k c$ and *cd* are. Microdynamic interaction between levels couples contextual effects in one level to the processing in other levels.

DSS does not offer an account of long-term variability of the kind observed by Bartlett, because learning is not yet integrated into the approach.

Conceptual cycles

As a last point, I briefly consider the issue of circularity. Cyclic interrelationships between concepts abound in human reasoning. However, on a basis of classical first-order logics, it is hard to come to terms with this phenomenon.

Nebel (1990, 1991) discusses different types of circularity. He argues that knowledge representation systems should tolerate a certain kind of circularity, which he calls "terminological cycles". Terminological cycles are essentially the same phenomenon as cyclic cross-references in an encyclopædia. For instance, a `car` concept is naturally explained with the help of an `engine` concept, and vice versa. Terminologic cycles could be formally dealt with by using non-well-founded set theory (standard reference: Aczel 1988, introduction: Barwise & Moss 1991) for a model-theoretic semantics. However, unfounded semantics are still unusual in symbolic AI. Nebel (1991) explicitly backs off from unfounded semantics. He proposes several ways of arriving at founded models for terminological cycles on a background of KL-ONE-like representation languages, acknowledging that each has its drawbacks. In (Nebel 1990), a KL-ONE-like formalism is augmented to support a kind of terminological cycles that is restricted, among others, by the requirement that such cycles must not lead (transitively) from a concept to a subconcept.

Cyclicity is a basic phenomenon in DSS. From an abstract point of view, this is a consequence of the emphasis on self-organization, which is intimately connected with feedback and hence, with a kind of cyclicity. When the issue is examined more closely, one finds two distinct types of cyclicity in DSS.

First, the global states of a dynamic symbol space are coherencies, which are cyclic. This leads to the phenomenon of associations looping back into themselves, i.e., to resonances. The forming of associations and resonances in a self-organizing stream is a *dynamic* type of cyclicity; I shall call it **associative** cyclicity.

Second, the grounding relation can give rise to cycles within the conceptual level. This corresponds to terminologic cycles of the encyclopædia type, which can be considered as *structural*. Let it be called, for obvious reasons, **explanatory** cyclicity.

In section 2.4, I have given an abstract characterization of the conceptual level in terms of, as it now turns out, these two kinds of cyclicity. I have required there that each kind of cyclicity leads to a global cyclic interconnectedness of the conceptual level. This requirement is satisfied

for associative cyclicity, by virtue of the construction of dynamic symbol spaces in terms of coherencies. Chains of grounding relations, on the other hand, have not as yet been required to be globally cyclic. It is motivated, then, to add this as an additional condition for associeties, in order to make the conceptual-level "encyclopædia" cyclic. This does not lead to any technical difficulties.

Associative and explanatory cyclicity are related with each other to the extent that associative attributes coincide with explanatory ones (see above). This coupling seems to add an important aspect to the structure/process problem (cf. 3.3), and should have an impact on self-organization. However, the issue remains to be worked out.

In sum, cyclicity is a built-in feature of DSS; thus, it does not pose the technical problems as in logic-oriented knowledge representation. Furthermore, DSS provides a formal frame to distinguish different kinds of cyclicity, and to investigate their interaction.

Summary of section 5

- A dynamic concepts is a dynamic symbol from the uppermost "conceptual" level in an associety, together with what it grounds in in lower levels, and with what it associates within the conceptual level.
- This account of a concept sheds a new light on several fundamental questions concerning the representation of concepts. In some cases, it offers an integrative perspective on seemingly disparate alternatives.
- Two basic kinds of attributes exist for dynamic concepts. Associative attributes concern the information that is dynamically accessible from a concept within the conceptual level; grounding attributes concern the information that is coupled to a dynamic concept symbol by the grounding relation.
- Grounding attributes can "grow" from a perceptual feature to a fully conceptual state. This variability of status allows the issue of analogical vs. language-like representations to be interpreted as a matter of degree.
- Associative attributes have a dual concept-relation nature. Technical problems concerning the arity of relations, as they are known from classical representation formats, do not occur.
- Dynamic concepts are vague in that many different subsets of their attributes can be present in different occurences of the concept. The empty intersection problem does not arise.
- With respect to their attributes, dynamic concepts vary dynamically in a history. Among other factors, this variation is influenced by context.
- "Structural" terminologic cycles and "dynamic" feedback cycles are two forms of cyclicity, which can be observed, and conveniantly examined, in associeties.

6 Related work

The work presented in this thesis draws from many sources, and tries to integrate insights that have been achieved at disparate places. The most important of these sources are (in alphabetical order):

(i) complex systems theories,
(ii) concept representation research in AI and cognitive science,
(iii) finite automata and regular languages, and
(iv) situated action and behavior-oriented agent design.

Its proximity to many different areas of research connects DSS with rather many threads of other work. But, the relationship between DSS and another approach is typically confined to some selected aspects. For instance, DSS is partially related to classical semantic networks through some aspects of dynamic symbol spaces; or to Carpenter and Grossberg's ART (1990) through the processing of sequential input, self-organization, and resonance formation; or to behavior-oriented agent design through the epistemological outlook on the agent/environment feedback loop. Such correspondences have been explored at various places in the preceding sections, and I do not reconsider them here.

When the DSS is characterized not in terms of disciplines concerned, but in terms of system properties, the following characteristics stand out:

(i) a fast self-organizing dynamics,
(ii) a symbolic format of information,
(iii) stream processing, and
(iv) support of hierarchic, multi-granularity architectures.

No other approach featuring this combination of properties appears to exist. However, there exists a class of neural network techniques which typically realize (i), (ii), and (iv): localist recurrent networks, where nodes are labeled by symbols. Being *localist*, these networks do not belong to the "mainstream" of connectionism, which is more inclined towards parallel *distributed* information processing. Localist networks are frequently used in cognitive science and linguistics, where their capabilities to handle contextual influences are exploited for various tasks (e.g., Waltz & Pollack 1985, Bookman 1988, Mangold-Allwinn 1991). These approaches are related to classical spreading activation techniques in semantic networks (e.g., Quillian 1968, Wilks 1975), and sometimes both traditions are integrated in a single system (e.g., Mehl 1992). I shall not review the entire family of approaches. Instead, I select two of its members for a closer inspection, namely, Smolensky's "harmony theory", and the "Copycat" architecture of Hofstadter and Mitchell. There are two reasons for picking out these two: first, they are, in some aspects, more closely related to DSS than others; second, they are motivated by an exploration of fundamental computational mechanisms (like DSS), rather than being "mere" tools for a specific application (like the majority of approaches in the field).

From a bird's eye view, Smolensky's (1986) harmony theory is a localist neural network model for pattern recognition, which spans the entire periphery-centre axis. Its particular appeal is formal simplicity and generality. This allows Smolensky to develop a principled mathematical theory, which describes the context-sensitive assembly of *schemas* in terms of minimizing an energy measure. In my presentation, I focus on the network architecture and the basic ideas, omitting mathematical detail. Smolensky uses a four-letter-word recognition task as an introductory example, which I shall use as well.

A *harmonium*, as Smolensky calls his architecture, is a two-layer network. The bottom layer is made from *representational feature nodes*. These nodes indicate by their activation whether some feature is present or not. No direct connections between them exist. The top layer consists of *knowledge atom nodes*. Each knowledge atom node corresponds to an on-off pattern of feature nodes. This pattern is determined by excitatory (+) and inhibitory (-) links to some of the feature nodes. For instance, a knowledge atom node for the detection of the letter "A" might have (+)-links to three feature nodes representing a horizontal bar "-", a diagonal bar "\", and a diagonal bar "/", and it would have (-)-links to feature nodes that represent, e.g., curved line segments (fig. 6.1). Obviously, there are many ways to detect an "A" under varying circumstances, and from different features. This is accounted for by the existence of many knowledge atom nodes, each of which corresponds to one possible way of detecting an "A".

Features, as understood by Smolensky, are not restricted to low perceptual levels. Besides features like bars and curve segments, there could be "features" corresponding to complete letters, complete words, or any other entity that one wishes to treat, all in one harmonium. Every knowledge atom responsible for detecting an "A" is (+)-connected to the "A" feature node, besides its connections to line segment nodes.

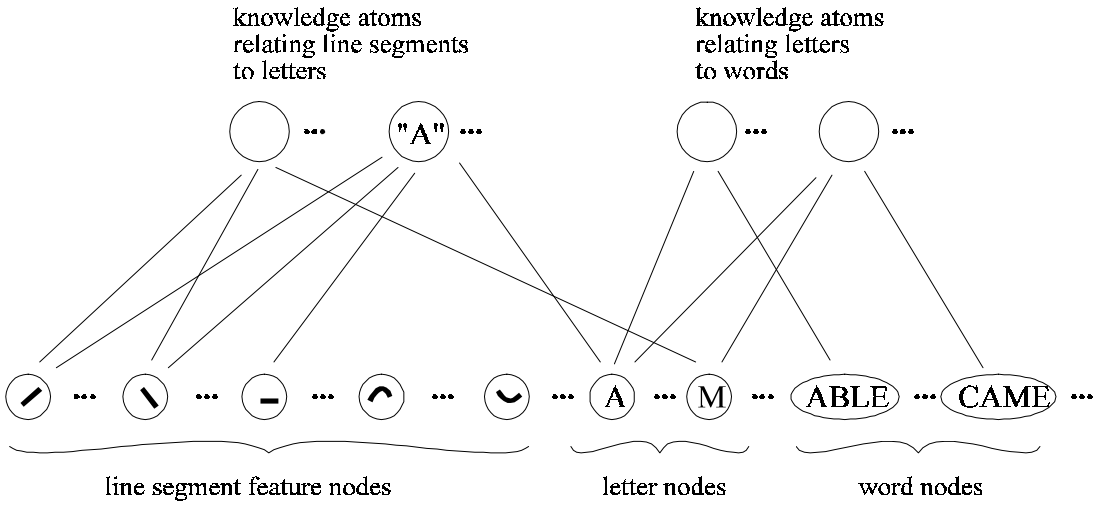


Fig. 6.1: A portion of a harmonium (adapted, and slightly simplified, from Smolensky 1986). Only (+)-links are shown.

The introduction of feature nodes that span several levels of integration allows one to "fold" a multi-level architecture into a two-layer network. When the harmonium from fig. 6.1 would be rearranged vertically, it becomes clear that a harmonium can span the entire periphery-centre axis (fig. 6.2).

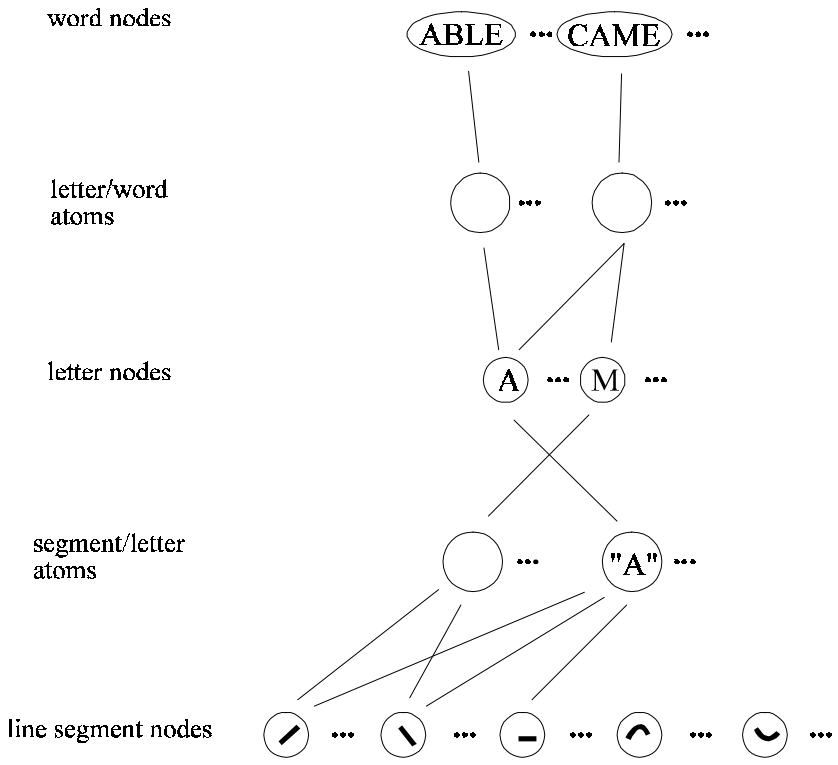


Fig. 6.2: The harmonium rearranged (adapted from Smolensky 1986).

A harmonium works in a *pattern completion task* as follows. Initially, some of the feature nodes are externally activated, and the activation is clamped. Then, a stochastic activation propagation and equilibration process is run. This process is of the simulated annealing kind, involving the successive lowering of a *computational temperature*. At temperature zero, the system settles down in a global state that maximizes a numerical value called *harmony*. Intuitively, the harmony value of an activation distribution measures the *coherency* of the network state. The coherency increases, roughly, when (+)-linked nodes (both feature and knowledge atom nodes) are simultaneously activated; it decreases when (-)-linked nodes are simultaneously activated (the harmony value also depends on the computational temperature, an aspect that I skip here). In the process, feature nodes that have not been initially activated may become so, which is the pattern completion aspect of the harmonium's dynamics. Furthermore, knowledge atom nodes can become activated, which is the "interpretative" aspect of the dynamics. The collective of activated knowledge atoms forms the *schema* that is recognized.

The major part of Smolensky's paper consists in a rigorous treatment of the pattern completion process in terms of statistical thermodynamics. This also explains why the author insists on a two-level architecture: the tools of statistical thermodynamics are tailored to deal with one

microlevel and one macrolevel, which correspond to the feature and the knowledge atom level. I can only highlight the central aspects of Smolensky's formal treatment:

- The harmony function and the computational temperature correspond formally to the notions of energy (or Hamiltonian function) and temperature, as they are used in thermodynamics. They allow explaining the harmonium in terms of *entropy*, which in turn is formally equivalent to *information*. Thus, the formal treatment is dually an information theoretic one, and one in terms of statistical mechanics, with probability theory as the shared mathematical basis (this dualism is inherent in the two disciplines; it is not a particular property of Smolensky's approach). It allows investigation of self-organization phenomena, as described in thermodynamics, in an information processing system.
- During the cooling process, *phase transitions*, or *symmetry breakings*, occur. Intuitively, they correspond to points in the process when the system arrives at an alternative of how currently stabilized, smaller collectives of mutually supportive nodes can be further coupled into larger such collectives. Formally, they reveal themselves in discontinuities of the harmony function. Smolensky reports simulation runs where this phenomenon can be clearly observed.
- The final activation values of knowledge atom nodes can be interpreted as likelihood *estimates* for the presence of the corresponding feature pattern. A harmonium performs by definition optimally, when these estimates yield exactly the probability of the completed pattern, given the incomplete information of initially activated features. Smolensky proves that the harmonium can behave optimally, and provides a *learning* strategy to arrive at such a behavior.

There are several clear similarities between a harmonium and an associety:

- A periphery-centre hierarchy is described in terms of symbolic units of increasing compositional integration (knowledge atoms made from feature patterns vs. dynamic symbols grounding in dynamic composites).
- A self-organizing dynamics leads to the formation of coherent assemblies of informational entities (schemas vs. resonances and characters).
- Contextual influences are crucial for the interaction of informational entities ((+) and (-) links vs. filtering of continuations).
- Many circumstance-dependent "versions" of "one" informational entity are possible (e.g., many "A" knowledge atoms vs. many occurences of the `apple` dynamic symbol in a global state like the one from fig. 3.5).
- Computational temperature and symmetry breaking are basic phenomena in both approaches.

Another similarity is not so easily detected, though it is, I believe, fundamental. Both in harmony theory and DSS, concepts and attributes essentially coincide. Now, in order to make this point, I must first determine which objects should count as "concepts" in harmony theory, since the notion is not used by Smolensky. The most natural candidate for, e.g., an "A" concept is, as I see it, the set of "A"-recognizing knowledge atom nodes, plus the feature nodes they are (+)-linked to. At first sight, this defines a concept in terms of a collection of sets of features, each of which is sufficient to establish the presence of the concept. Features (i.e., attributes) and concepts would be clearly distinct kinds of entities, then. But, among these features, there exists a very special one, which leads to a revision of this first impression: the "A" feature node. This "feature" is not a visual feature or a partial aspect of the "A" concept. In a way, it can be regarded as the essence of the concept, since in this node, all the different circumstantial versions of the "A" concept are tied together. Considering the role of this node as a link for coupling adjacent levels of granularity (cf. fig. 6.2), a close similarity is revealed

with a dynamic symbol emerging from a lower-level dynamic composite. A technical difference lies in the fact that the "A" feature node "grounds" in lower-level composites through the relays of knowledge atom nodes, whereas in DSS the grounding is unmediated. A feature node, thus, can be seen dually as the "representative" of a concept, and as a feature of other concepts. In sum, symbols, concepts, and attributes are not clearly separated from each other in (my interpretation of) harmony theory.

On the other hand, there are many differences between harmony theory and DSS. I list the most conspicuous ones:

- Associeties are intended for continual stream processing, whereas a harmonium performs a single run on a single input.
- The information processed in an associety comes in a relatively expressive format (directed edge-labeled graphs, with an abstraction defined for labels, and with the possibility to define analogues of "instances", i.e., occurences of a concept). By contrast, a harmonium is put to work on a simple set of features, and returns a set of knowledge atoms.
- DSS distinguishes between "long-term memory" (dynamic symbol spaces) and "working memory" (streams and associeties), whereas in a harmonium both sites coincide.
- Microdynamics works by modifying the very connectivity structures between informational entities, whereas a harmonium works by spreading activation through a fixed connectivity network.

The two approaches stem from different basic motives. DSS is intended to help in the modeling of situated agents. This leads to stream processing, rich formats - and some unfortunate blanks in the formal analysis of associeties. Harmony theory aims at a rigorous formal treatment of one, albeit fundamental, information processing task, i.e., pattern completion. This leads to a simple basic architecture and working cycle - and a satisfyingly complete formal penetration.

Interpreting DSS in terms of statistical thermodynamics (or information theory) will certainly be much more difficult than in harmony theory. One of the major obstacles is the multi-level structure of an associety, which would require a thermodynamic treatment of more levels than one micro- and one macrolevel. This is uncommon but not impossible for statistical thermodynamics. Von Weizsäcker (1985, p. 180ff) carries out such an analysis for a simple three-level system. On the other hand, some starting points for a thermodynamic/information theoretic interpretation of DSS can be already made out, namely, the interpretation of abstraction in terms of resolution and differentiation (which should be amenable to information theory), and the definition of the information contained in an association. Furthermore, there already exists a well-established strand of research concerned with the statistical thermodynamics of sequence generating systems, in particular finite automata (e.g., Crutchfield & Young 1990). This work is directly relevant for the analysis of coherencies.

Copycat

The Copycat project (Hofstadter & Mitchell 1992) constitutes a major endeavor to make an AI program behave creatively in a nontrivial sense of the word. Copycat's task is to complete letter sequences, by way of exploiting analogy. A typical question posed to the system might run like this:

Suppose the letter-string **abc** *were changed to* **abd***; how would* **ijk** *have to be changed "in the same way"?*

A plausible answer would be **ijl**, and indeed Copycat will very often come up with it (and sometimes with others). Finding this answer boils down to *discovering* a rule that yields **abd** from **abc**, and then *transferring* this rule by analogy to **ijk**. This example may not seem very demanding, but Copycat can cope with much more subtle tasks. For instance, when (again) **abc** changes to **abd**, to what changes **mrrjjj**? Of course, there are several answers. The one most frequently given by humans is **mrrkkk** (rule: replace last letter by its alphabetic successor), and this is also the solution most frequently found by Copycat. However, this response is not entirely satisfying, since **abc** displays alphabetical order, whereas **mrrjjj** does not. There exists a different, surprisingly elegant solution, which is less frequently found by humans, and which is also rarer among Copycat's answers - but sometimes, indeed, the required "stroke of genius" effectively happens in the machine (solution at the end of the Copycat review). The authors argue that the "microdomain" of letter sequence completions contains arbitrarily demanding tasks; therefore, it is, in a sense, a universal testbed for creative computation.

Copycat's architecture has two principal modules. The *Slipnet* more or less functions as a long-term memory, and the *Workspace* as a working memory or blackboard. Additionally, there is a *Coderack*, which is roughly analogous to an agenda of pending tasks. Before I describe these modules, I emphasize that Copycat is a highly intricate affair, with dozens of mechanisms interacting in a finely tuned fashion. I can outline here but some major principles.

The Slipnet is a network of labeled nodes, which are connected by links of variable *length*. The node labels concern all sorts of concepts that are useful for the description of letter sequences. For instance, there are nodes for each letter of the alphabet, nodes for relations between letters (e.g., "alphabetic successor"), and nodes for *deeper* concepts like "symmetry" or "opposite". They are *not* ordered in an abstraction hierarchy as one is used to from semantic networks. The length of a link between two nodes corresponds to the "semantic distance" between the nodes. For instance, a node for the letter **b** will typically have a shorter link to the **c** node than it has to the **r** node. The semantic distance is important, among other effects, for "mental slips" (hence, Slipnet): when a node is read into the "working memory", it can happen that a semantically close one is also read.

The nodes in the slipnet are dynamically *activated* during Copycat's run, with activation reflecting the importance of this node for the interpretation of the currently considered letter sequence. This activity spreads over the links to neighboring nodes; the spreading rate accross a link is inversely related to the link's length. So far, this is all rather conventional. But now comes Slipnet's special contribution to the art of long-term knowledge representation. *The links are labeled by nodes*. For each link, there exists some node in the Slipnet that serves as a label for the link. For instance, a link between a "left" and a "right" node might be labeled by the "opposite" node. Furthermore, a link's length is inversely related to the corresponding node's activation value. I.e., when the "opposite" node is highly activated, the link between the

"left" and the "right" node becomes short, favoring the spreading of activation between the two, and favoring mental slippage.

A *concept* is a node in the Slipnet, *plus* the nodes which are linked to it by short links. Hofstadter and Mitchell apply the metaphor of a "probabilistic cloud around a node" to this conception of concepts. The variability of link length, which is induced by labeling node activation, lets these "semantic clouds" fluctuate in a context-sensitive fashion. For instance, when the "opposite" context is highly active, then "left" belongs closely to the concept centered around "right".

The Workspace is a complex machinery, which I can sketch only superficially. At the beginning of a Copycat run, the workspace contains the bare task description, e.g. "**abc** $\rightarrow$ **abd**; **ijk** $\rightarrow$?". Successively, relations between **abc** and **abd**, and between **ijk** and **abc** are discovered, and relations between these relations, etc. This interpretative information is added to the workspace in several formats. During a run, a currently established interpretation (called *viewpoint*) can be superseded by more *consistent* ones. The consistency parameter depends on many influences, among others, on the interconnectedness of the viewpoint and its hierarchic depth (in terms of relations between relations). This is, however, not yet a fully accurate picture; consistency need not increase monotonically, since there are ways for less consistent viewpoints to push their way stochastically into the Workspace.

The actual computation is done by calling *codelets*. A codelet is a small piece of executable code. There are various types of codelets, some that can only detect and memorize some regularity (called *scouts*), and others that can actually alter the state of the Workspace and the Slipnet. New codelets can be generated by several mechanisms. At a given time, a (large) number of codelets waits in the Coderack to be called in a stochastic fashion. Each of them is marked with a priority value, which governs its chances to be called.

Although only a single viewpoint is explicitly present in the Workspace, competing viewpoints can be implicit in codelets that lie "dormant" in the Coderack. When they happen to be called, it may occur that a relatively coherent viewpoint is exchanged for a less coherent one. This is an important mechanism to avoid getting stuck in impasses, and for finding "surprising" solutions.

The coherency of a given viewpoint regulates a *computational temperature*. The greater the coherency, the lower the temperature. High temperature has the effect that codelet execution, roughly, is "more random". At the beginning, coherency is minimal, temperature maximal, and the overall behavior is nearly random. At the end, the opposite is true. Thus, by and large, the dynamics behaves like in simulated annealing. However, in simulated annealing the temperature is an externally supplied control parameter, whereas here it reflects the current degree of coherency; also, in Copycat the temperature is apt to fluctuate, wich it usually does not in simulated annealing regimes.

The authors provide a metaphor for the basic principles of Copycat that is quite helpful to get an intuitive grasp of the matter. They compare the Workspace with a biological cell, where complex biochemical aggregates are built through the activity of enzymes, which in turn are produced from DNA strands, with many inhibitory and positive feedback cycles being involved, and with mechanisms to suppress or express particular portions of the DNA. The enzymes correspond to the codelets, the DNA to the Slipnet (with suppression/expression representing the activation dynamics), and the biochemical aggregates to the viewpoint in the Workspace. One of the crucial things to be gathered from this metaphor is the stochastic and

fine-grained collective nature of the ongoing processing, where a single codelet execution has only minimal effects.

There are several similarities between Copycat and DSS. Some of them are of a more general nature:

- Both systems combine a self-organizing dynamics with symbolic knowledge processing (my interpretation; the authors locate Copycat somewhere between the symbolic and the subsymbolic paradigm).
- There is a long-term memory and a working memory (Slipnet and Workspace vs. dynamic symbol space and self-organizing stream).
- In working memory, coherent assemblies of informational entities are formed (viewpoints vs. resonances).

However, such similarities could be found to relate DSS with many other systems in the field besides Copycat. The central aspect of likeness, which motivated my reviewing of Copycat, lies in the Slipnet's principle of labeling links by nodes. By virtue of this elegant technique, relations are effectively identified with concepts. This exactly matches the interpretion of the conceptual level in DSS associeties (cf. section 5). The technical apparatus of DSS is not very similar to the one realized in the Slipnet, but the basic idea, and the implications for context sensitivity of attribute accessibility (where the attributes of a Slipnet concept node are taken to be its proximal nodes), are very much alike.

It is, on the other hand, not difficult to spot many differences between Copycat and DSS. I list the most important ones:

- In contrast to multi-level associetes, Copycat works only on one, namely, the conceptual level in the periphery-centre hierarchy.
- Copycat is a single-task single-run device, whereas associeties are for stream processing.
- Copycat's dynamics is, one might say, "more collective", or "finer grained", than DSS microdynamics. In its "evolutionary" character, it can be compared (which Hofstadter and Mitchell do) to the dynamics of classifier systems. By contrast, the equilibration of a self-organizing scene can be expected to be a rapid phenomenon.

One of the things that DSS might learn from Copycat concerns the control of global working parameters. A self-organizing stream history unfolds under the given of a global state. I have not addressed in this thesis the question of general strategies of how a "suitable" global state can be determined (in the application described in section 4, the task is simple enough to fix a particular global state beforehand). In Copycat, the computational temperature depends directly on the currently achieved degree of coherence. By analogy to Copycat, it might be motivated to explore whether the global state might be set dynamically in agreement with some measure of coherence of configurations.

(P.S.: When **abc** changes to **abd**, then **mrrjjj** changes, quite elegantly, to **mrrjjjj**. That's how: **a** is the 1st, **b** the 2nd, **c** the 3rd letter of the alphabet; **m** appears 1 time, **r** 2 times, and **j** 3 times. The transition "3 $\rightarrow$ 4" governs, then, both the original and the solution.)

Discussion

Harmony theory, Copycat, and DSS are three approaches to realizing a kind of information processing that is both symbolic and self-organizing. To be sure, Smolensky would not consider his work "symbolic", and neither would Hofstadter and Mitchell be happy with this characteriziation. However, the rejection of the term by these researchers stems from their opposition to a logic-oriented, inferential *dynamics*, and the underlying model-theoretic *semantics*; but I use the term here merely to refer to the *representation* of information in terms of symbolic labels.

The three approaches aim at quite different tasks: pattern completion, creative analogy-making, and stream processing for agents. A mutual comparison cannot, therefore, rest on the *tasks*. Rather, it should focus on the *mechanisms* that are exploited, and the underlying *principles* that are assumed to govern information processing.

I find several basic aspects that are shared by the three architectures, which might turn out to be universal for *any* approach that combines a self-organizing dynamics with a symbolic format of information:

- *Local interaction in a collective of informational entities*. This certainly characterizes the three approaches in question here. Why might it be universal? First, a symbolic format implies that there are discrete entities. Since there obviously must be more than one entity present for anything interesting to happen, one has a collective of informational entities with which to start. Second, it is hard to conceive how a globally controlled dynamics might give rise to self-organization. *Control* and *self*-organization is all but a contradiction in terms. Discarding global control, one is left with local interaction.

- *Formation of "coherent" composites*. In a harmonium run, a complex schema is successively assembled from mutually (+)-linked clusters and meta-clusters of nodes. In Copycat's Workspace, interpretations of letter sequences, and their mutual analogies, are successively built by intermeshing letters and letter-groups with relations, relations between relations, etc. Resonances develop in a self-organizing stream. In all these cases, the authors apply the term "coherent" to the respective phenomenon. Furthermore, there is a numerical measure for coherency in each case, which is directly related to the relative stability of the composite: the harmony function in harmony theory; the coherence value of a viewpoint in Copycat; and the information of associations in DSS. This cannot be a coincidence. Harmony theory provides the relevant insight: the harmony function can be rigorously interpreted as a measure of energy. Principally, from a thermodynamic point of view, self-organization is explained in terms of minimizing an energy measure. For DSS, such an interpretation is a goal for further formal research.

- *Coincidence of concepts, attributes, relations; and context sensitivity of these unified entities.* This is an ill-specified claim, since the terms "concept", "attribute", and "relation" are not used either by Smolensky or by me, in the first place. However, in harmony theory and DSS, these notions can be plausibly reconstructed. Then, the underlying similarity that concepts relate between concepts is revealed. In harmony theory, this is manifest in the fact that a node is a relay station for activation; in DSS, a dynamic concept is characterized by when and with which others it can associate; in Copycat's Slipnet, a concept is outrightly equivalent to a modulatory relation between other concepts. Again, this appears to be more than a coincidence. In a dynamic, self-organizing system, an informational entity cannot be understood in isolation from other such entities, and their mutual interactions. An

informational entity, in such a system, is intrinsically a dynamic object, and its dynamics intrinsically relates the object to others. An object, in a nutshell, *is* its own behavior towards neighboring objects; it cannot be abstracted away from that. Context sensitivity can be considered a necessary consequence of this situation. When a concept is a dynamic relation, then its relating to other concepts forms a context for the latter; since this relating activity is in some way or other causally influential for the other concepts, they are context-sensitive. Seen this way, context sensitivity is a mere epiphenomenon, and there is nothing in it that is remarkable in any way.

- *Feedback and cyclicity.* I need not expand on this issue, since it is obvious both in its role for the three approaches in question here, and in its universal relevance for self-organization.

- *Thermodynamic states as global control parameters.* The harmonium, Copycat, and self-organizing streams are uniformly ruled in their microbehavior by global states that allow for a thermodynamic interpretation, though only in harmony theory this interpretation is rigorously carried out. The question is, again, how universal is this? It might seem that it is not, considering cellular automata and neural networks that work without reference to global states. However, this may be due to the fact that such a global state is simply not *varied* there, although a fixed one could be defined. In collective computation techniques, it should always be possible to introduce and to vary randomness, thereby equipping the system with some kind of computational temperature. Non-random computations, in this view, are simply computations at zero temperature (note that "heating" is generally not possible for classical symbolic algorithms, which would typically break down completely when randomness was allowed). Therefore, I will take it as granted that collective computation is generally subject to global states that bear out a more or less rigorous thermodynamic interpretation. A question of general relevance concerns the control of such global parameters. In the harmonium, a one-way cooling strategy is employed; in Copycat, the computational temperature is self-adjusting to the currently achieved coherency; in DSS, the problem is left open. It seems to me that decisive insights into the nature of this question are still missing, in particular when a continuous processing of a stream of information is at stake.

A general observation can be made concerning the formal reconstruction of systems like the harmonium or DSS streams, in terms of set-theoretic mathematical structures. A standard formalization seems inadequate or even impossible. This follows (a) from the coincidence of concepts with relations, which disagrees with the modeling of standard fixed-arity relations by sets of tuples (cf. Scott 1978 for "multi-relations"), (b) from feedback, which would require, at least, an unfounded version of model theory (for unfounded set theory in accounts of stream processing cf. Barwise & Moss 1991), and (c) from context sensitivity of concepts, which would require "sets with context-sensitive extension". Although set-theoretic structures with such properties would be somewhat non-standard, I believe that exploring them is fruitful for a deeper understanding of self-organiziation, and I have started to work in this direction.

Harmony theory, DSS, and Copycat differ widely in their complexity, with harmony theory being the simplest and Copycat the most intricate. This is mirrored in the degree of formal analysis that has been achieved in each case, which varies from the almost complete and rigorous treatment in harmony theory, via the formal language underpinnings of DSS and its prospects for a further exploration in terms of statistical thermodynamics, to the non-existence (impossibility?) of a formal model for Copycat. This raises the important question whether a "truly" intelligent (which would include creativity, as performed by Copycat) system can be

formally penetrable at all, or whether this might in the end turn out a contradiction in terms. Be this as it may, DSS bears a formal analysis.

The harmonium, Copycat, and self-organizing streams differ greatly with respect to the actual computational dynamics. Spreading activation, codelet execution, and microchanges are quite dissimilar "micro-mechanisms". Furthermore, other approaches in the field of collective symbolic computation, such as classifier systems and cellular automata, add yet other principles of micro-mechanisms. This variance is repeated at the macrolevel. Slow "evolutionary" systems (Copycat, classifier systems) contrast with fast "equilibrative" ones (harmonium and most neural networks, cellular automata, DSS). A unified theoretical perspective is not in sight; in the long run, statistical thermodynamics and information theory might yield an appropriate general frame of reference.

I have selected harmony theory and Copycat for review not only because they are as close to DSS as can be found, but also because they highlight the similarities as well as the dissimilarities of approaches to self-organizing, symbolic information processing. The dissimilarities lie in the concrete computational techniques, in the nature of tasks, and, importantly, in the suitedness for a formal analysis; the similarities lie in the generic properties of collective, yet symbolic, information processing, which I have listed above. DSS is a new member in a family of approaches, which are hardly yet systematically interrelated beyond those generic properties. The field is in a stadium of initial growth, which is characterized by the co-existence of many paradigms and techniques. Given this situation, the entry of DSS into the game is justified by two points:

- Among the systems that perform self-organization of symbolic information, DSS is unique in its combination of stream processing with a coverage of the full periphery-centre hierarchy. This reflects the central purpose of my work, i.e., to contribute to the modeling of intelligent, situated agents.
- DSS unfolds stringently from a simple formal basis, namely, coherent languages, which are a subclass of regular languages. It can reasonably be expected that a thorough analysis of DSS in terms of statistical thermodynamics is possible. In a field that is far from being understood systematically, formal stringency is particularly desirable.

Summary of section 6

- There are many approaches that combine a self-organizing dynamics with a basically symbolic mode of information processing. However, they are quite diverse. I examine two of them more closely, namely, Smolensky's "harmony theory" and Hofstadter and Mitchell's "Copycat". Although these two are selected for maximal proximity to DSS, it turns out that there are still considerable dissimilarities.
- Harmony theory is a localist connectionist technique for pattern completion tasks. An outstanding feature is a rigorous formal analysis in terms of statistical thermodynamics, which is made possible by the architecture's basic simplicity.
- Copycat performs on letter sequence completion tasks, which require a creative exploitation of analogy.
- Several general, shared traits are abstracted from harmony theory, Copycat, and DSS. I argue that these similar characteristics are, in fact, fundamental for any approach that combines self-organization with a symbolic format of information. In particular, the following points are identified:

- local interaction in a collective of informational entities,
- formation of "coherent" composites,
- coincidence of concepts, attributes, relations; and context sensitivity of these unified entities,
- feedback and cyclicity, and
- thermodynamic states as global control parameters.

- The unique contribution of DSS to this family of computational approaches lies in its combination of stream processing with a multi-level architecture. A second important justification is its accessibility for a formal treatment.

7 Conclusion

Looking back on the preceding sections, the following points appear as the fundamental characteristics of my work:

- DSS unfolds, in a formally rigorous way, from the theory of coherent languages. Coherent languages are a subclass of regular languages. They are motivated by fundamental aspects of local observations of dynamic systems, which in turn directly imply a notion of context sensitivity.
- The basic informational entities in DSS are dynamic symbols. They are interpreted as empirical observables. This leads directly to an abstraction relation between dynamic symbols, which is formally almost, but not quite, similar to the classical abstraction defined in terms of inclusion of extensions.
- The DSS model of long-term memory, dynamic symbol spaces, is in many aspects related to classical semantic networks. However, the proper perspective on dynamic symbol spaces is to view them as thermodynamic global state spaces.
- The DSS model of a processing module, the self-organizing stream, is an open, dissipative, rapidly self-organizing system. In computational terms, it is an anytime-algorithm for stream processing.
- Coherent spatiotemporal subsystems, resonances, can develop in a self-organizing stream. They can be interpreted as a phenomenon of gestalt formation.
- Complex multi-level architectures can be constructed by coupling several self-organizing streams. There are two coupling mechanisms. Coupling by bands is suited for connecting streams "laterally" by distinct communication pathways. Coupling by emergence/grounding relations yields multi-granular, "vertical" columns of processing levels, where top-down and bottom-up influences have equal rights.

Contributions

The work reported in this thesis contributes to AI in general through the following points:

- The structuralistic epistemological framework of dynamic symbol structures, of which DSS is a concrete instance, provides a unified theoretical perspective on two paradigms that are often considered to be mutually incompatible, namely, the paradigms of situated action and physical symbol systems.
- On the conceptual level, DSS proposes solutions for some problems that are hard for traditional, logic-oriented concept representations formalisms:
  - nonmonotonic inheritance,
  - the nature of symbols vs. concepts,
  - the nature of concepts vs. attributes,
  - analogical vs. language-like representations,
  - vagueness, variability, and context sensitivity,
  - conceptual cycles.

- DSS is unique in its combination of the following features, which are of particular interest for modeling agents:
  - a self-organizing dynamics,
  - a symbolic format of information,
  - stream processing, and
  - support of hierarchic, multi-granularity architectures.
- DSS is unique within AI in that it exploits topology-changing graph modifications for the micromechanism of collective computation. Since these graphs (configurations) are of a dual temporal/structural nature, DSS is a suitable formal frame for further explorations into fundamental questions concerning time vs. structure.

The central application aimed at by the approach lies in agent design. Associeties provide a programming scheme for the entire periphery-centre axis of a situated agent. The following points justify DSS in this perspective:

- Due to the symbolic nature of DSS, the design can be explicit (cf. the "explicit design principle" in 2.10). Together with the unified format for all levels of the periphery-centre dimension, this fosters transparency and intelligibility, which in turn are beneficial for modification and systematic experimentation.
- Due to compositionality, an ad-hoc setup of behaviors is in principle possible, which in turn is a precondition for discover-and-modify development schemes (cf. the "open design principle" in 2.10). However, before this can be realized, learning must be incorporated into the approach.
- Fast self-organization, the anytime character of self-organizing streams, and the tight interaction of bottom-up with top-down processing warrant the agent to be coupled into its environment by a tight agent/environment feedback loop, as emphasized by situated action.
- Conversely, the relative detachment and autonomy of higher processing levels and the representational capabilities comparable to classical semanitc networks, provide a precondition for the "intellective" type of intelligence that is emphasized in classical AI.

However, this is speculation, since DSS is unimplemented as yet. It remains to be seen to what extent these points can be realized.

Outlook

The work presented in this thesis provides only the formal groundwork for DSS. Before DSS can be utilized in practical applications, further work must be carried out. I list some obvious tasks, in the order of their urgency, which I am going to tackle with the least possible delay:

- Simulations of self-organizing scenes and streams must be run, in order to gain experience with the diverse phenomena of self-organization. Such computer simulations are indispensable, since a theoretical prediction of ergodic properties of the self-organization phenomena concerned is inherently difficult.
- Control regimes for the dynamic selection of global states for passages of a history must be developed in the course of these simulation experiments.
- Acquisition techniques for the construction of dynamic symbol spaces must be developed. In the long run, it would be very desirable, e.g., to acquire conceptual-level dynamic

symbol spaces automatically from texts, by a direct transformation of observed keyword sequences into a growing dynamic symbol space. The problem of *learning*, in the sense of augmenting a given such space incrementally, is directly related.

- Dynamic symbol spaces are, as yet, defined in terms of more or less arbitrarily selected generators. It is desirable to find a representation format that is independent from particular generators. This requires a further mathematical elaboration of the interdependencies between, on the one hand, abstraction and symmetry breaking, and on the other, phase generators and simulation mappings.
- A task that is intriguing, but not quite so urgent, lies in a theoretical analysis of self-organizing streams in terms of statistical thermodynamics and information theory. This constitutes an ultimate goal for the approach.

A final remark

Self-organization, at least as it is - imperfectly - understood today, is possibly necessary, but certainly not sufficient to explain intelligent behavior. In particular, a purely DSS-based robot could not properly react to instructions that require a nontrivial syntax analysis. Charniak (1983) argues that language understanding builds on *two* kinds of mechanisms, one being self-organizing (though that term is not used by Charniak himself), the other exploiting functional relations between concepts that are expressed by purely syntactical means. I find his arguments convincing.

Furthermore, self-organization is not at all wanted for AI applications where a full control is required. Self-organization intrinsically implies unpredictability. An AI system for the surveillance of a chemical plant, or the support of air traffic control, or almost any other conceivable application in management and industry must not behave unpredictably. Such systems constitute the vast majority of economically relevant AI applications. The symbols manipulated there have a clear (often externally extensional) meaning *for the user* of such a system, just as the speedometer readings and the steering wheel have a clear meaning for the driver of a car. A trained logician, I believe that a classical extensional logic is the appropriate mathematical background. This insight notwithstanding, self-organizing mechanisms can serve auxiliary tasks in such systems, for instance in facilitating memory access.

Thus, in sum, self-organization is far from being a panacea for AI. It may be a valuable auxiliary mechanism in classical applications. In the particular AI subdiscipline of agent design, however, I believe it to be, though not sufficient, essential. These closing statements put my work in the frame where it belongs.

References

Aczel, P. (1988): Non-well-founded Sets. CSLI lecture notes 14, Stanford 1988

Antoniou, G., Wachsmuth, I. (1993): Structuring and Modules for Knowledge Bases: Motivation for a New Model. Knowledge Based Systems 7, No 1, 1994, 49-51

Arbib, M.A. (1987): Modularity and Interaction of Brain Regions Underlying Visuomotor Coordination. In: Garfield, J.L. (ed.) (1987): Modularity in Knowledge Representation and Natural Language Understanding. Bradford/MIT Press, Cambridge, Mass., 1987, 333-364

Barsalou, L.W. (1987): The Instability of Graded Structure: Implications for the Nature of Concepts. In: Neisser, U. (ed.): Concepts and Conceptual Development: Ecological and Intellectual Factors in Categorization. Cambridge University Press, Cambridge 1987

Barsalou, L.W. (1989): Intraconcept similarity and its implications for interconcept similarity. In: Vosniadou, S. und Ortony, A. (eds.): Similarity and Analogical Reasoning. Cambridge University Press, Cambridge 1989, 76-121

Bartlett, F.C. (1932): Remembering: A Study in Experimental and Social Psychology. Cambridge University Press, Cambridge 1932 (last reprint 1977)

Barwise, J., Moss, L. (1991): Hypersets. The Mathematical Intelligencer 13 (4), 1991, 31-41

Bookman, L.A. (1988): A Connectionist Scheme for Modelling Context. In: Touretzky, D., Hinton, G., Sejnowski, T. (eds.): Proceedings of the 1988 Summer School on Connectionist Models, Morgan Kaufmann 1988, 281-290

Brachman, R.J., McGuinness, D.L., Patel-Schneider, P.F., Resnick, L.A. (1991): Living with CLASSIC: When and How to Use a KL-ONE-like Language. In: Sowa, John (ed.) (1991). Principles of Semantic Networks. Morgan Kaufmann, San Mateo, 401-456

Brachman, R.J., Schmolze, J.G. (1985): An Overview of the KL-ONE Knowledge Representation Scheme. Cognitive Science 9 (2), 1985, 171-216

Brooks, R.A. (1989): The Whole Iguana. In: Brady, M. (ed.): Robotics Science. MIT Press, Cambridge, Mass., 1989, 432-456

Brooks, R.A. (1991): New Approaches to Robotics. Science 253, 1227-1232

Burge, T. (1987): Marr's Theory of Vision. In: Garfield, J.L. (ed.) (1987): Modularity in Knowledge Representation and Natural Language Understanding. Bradford/MIT Press, Cambridge, Mass., 1987, 365-382

Burks, A.W. (ed.) (1966): John von Neumann: Theory of Self-reproducing Automata. University of Illinois Press, Urbana 1966

Cabanac, M. (1992): Pleasure: The Common Currency. Journal of theoretical Biology 155 (1992), 173-200

Carpenter, G.A., Grossberg, S. (1990): Adaptive Resonance Theory: Neural Network Architectures for Self-Organizing Pattern Recognition. In: Eckmilller, R., Hartmann, G., Hauske, G. (eds.) (1990): Parallel Processing in Neural Systems and Computers. Proceedings of the International Conference on Parallel Processing in Neural Systems and Computers, Düsseldorf, May 1990. Elsevier/North Holland, Amsterdam 1990, 383-389

McCarthy, J., Hayes, P.J. (1969): Some Philosophical Problems from the Standpoint of Artificial Intelligence. Machine Intelligence 4, 463-502. Reprinted in: Webber, B.L., Nilsson, N.J. (eds.): Readings in Artificial Intelligence, Tioga, Palo Alto 1981, 438 - 443. Also reprinted in: Ginsberg, M.L.(ed.): Readings in Nonmonotonic Reasoning. Morgan Kaufmann, Los Altos 1987.

Chalmers, D.J. (1990): Subsymbolic Computation and the Chinese Room. In: Dinsmore, J. (ed.): The Symbolic and Connectionist Paradigms: Closing the Gap. Lawrence Erlbaum, Hillsdale N.J., 1992, 25-48

Chalmers, D.J. (1992): Connectionism and Compositionality: Why Fodor and Pylyshyn Were Wrong. Philosophical Psychology 6, 1993, 305-319

Chalmers, D.J. (1993): Toward a Theory of Consciousness. Dissertation thesis, Indiana University, Bloomington, Indiana 1993

Charniak, E. (1983): Passing Markers: A Theory of Contextual Influence in Language Comprehension. Cognitive Science 7, 1983, 171-190

Clancey, W.J. (1993): Situated Action: A Neuropsychological Interpretation Response to Vera and Simon. Cognitive Science 17, No 1, 1993, 87-116

Cohen, P.R., Levesque, H.J. (1990): Intention is Choice with Commitment. Artificial Intelligence 42 (1990), 213-261

Congdon, C.B., Huber, M., Kortenkamp, D., Bidlack, C., Cohen, C., Huffman, S., Koss, F., Raschke, U., Wemouth, T., Konolige, K., Myers, K., Saffiotti, A., Ruspini, E., Musto, D. (1994): CARMEL vs. Flakey: A Comparison of Two Robots. Preprint from the AI Lab of the University of Michigan, Ann Arbor, and the SRI, Stanford.

Crutchfield, J.P., Young, K. (1990): Computation at the Onset of Chaos. In: Zurek, W.H. (ed.): Complexity, Entropy, and the Physics of Information. SFI Studies in the Sciences of Complexity, vol. VIII, Addison-Wesley, 1990, 223-269

Devlin, K. (1991): Logic and Information. Cambridge University Press, Cambridge, 1991

Doyle, J. (1979): A Truth Maintenance System. Artificial Intelligence 12 (1979), 231-272

Drescher, G.L. (1991): Made-up Minds: A Constructivist Approach to Artificial Intelligence. MIT Press, Cambridge, Mass., 1991

Dress, A., Hendrichs, H., Küppers, G. (eds.) (1986): Selbstorganisation: Die Entstehung von Ordnung in Natur und Gesellschaft. Piper, München 1986

Ebbinghaus, H.-D., Flum, J., Thomas, W. (1978): Einführung in die mathematische Logik. Wissenschaftliche Buchgesellschaft, Darmstadt 1978

Flavell, J.H. (1968): The Developmental Psychology of Jean Piaget. Van Nostrand, Princeton 1968

Fodor, J.A. (1987): Modules, Frames, Fridgeons, Sleeping Dogs and the Music of the Spheres. In: Garfield, J.L. (ed.): Modularity in Knowledge Representation and Natural Language Understanding. MIT Press, Cambridge, Mass. 1987, 25 - 36.

Fodor, J.A., Pylyshyn, Z.W. (1988): Connectionism and Cognitive Architecture: A Critical Analysis. Cognition 28, 1988, 3-71

Forrest, S. (1989): Emergent Computation: Self-organizing, Collective, and Cooperative Phenomena in Natural and Artificial Computing Networks. Introduction to the proceedings of the 9th annual CNLS conference, Los Alamos 1989. North Holland 1990

Forrest, S., Miller, J.H. (1990): Emergent Behavior in Classifier Systems. Physica D42 (1990), 213-227

Gängler, B., Greten, M., Linke, T., Wachsmuth, I. (1992): Assoziative Zuordnung und Suche von Wissen in einer thematisch strukturierten Wissensbasis. MOSYS Report 11, Faculty of Technology, University of Bielefeld 1992

Gécsec, F., Steinby, M. (1984): Tree Automata. Akadémiai Kiadó, Budapest 1984

van Gelder, T., Port, R. (1993): Beyond Symbolic: Prolegomena to a Kama-Sutra of Compositionality. To appear in: Honavar, V., Uhr, L. (eds.): Symbol Processing and Connectionist Models in Artificial Intelligence and Cognition: Steps towards Integration. Academic Press, 1993

Goldberg, D.E. (1989): Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, Mass. 1989

Greeno, J.G., Moore, J.L. (1993): Situativity and Symbols: Response to Vera and Simon. Cognitive Science 17, No 1, 1993, 49-60

Haken, H. (1983): Advanced Synergetics. Springer Series in Synergetics Vol. 20, Berlin/ Heidelberg/New York 1983, chapter 1

Hanks, S., Pollack, M.E., Cohen, P.R. (1993): Benchmarks, Test Beds, Controlled Experimentation, and the Design of Agent Architectures. AI Magazine, Winter 1993, 17-42

Harnad, S. (1987): Category Induction and Representation. In: Harnad, S. (ed.) (1987): Categorical Perception. Cambridge University Press, Cambridge, Mass., 535-565

Harnad, S. (1990): The Symbol Grounding Problem. Physica D 42, 335-346.

Holland, J.H. (1975): Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, 1975

Holland, J.H. (1986): Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems. In: Anderson, J.R., Michalski, R.S., Carbonell, J.G., Mitchell, T.M., Amarel, S. (eds.): Machine Learning, Vol. II, Morgan Kaufman, San Mateo 1986

Hofstadter, D.R., Mitchell, M. (1992): The Copycat Project: A Model of Mental Fluidity and Analogy-Making. To appear in: Holyoak, K., Barnden, J. (eds.): Advances in Connectionist and Neural Computation Theory, Vol. II: Analogical Connections. Ablex, Norwood, N.J.

Hopcroft, J.E., Ullman, J.D. (1979): Introduction to Automata Theory, Languages, and Computation. Addison Wesley, Reading, Mass., 1979

Jaeger, H. (1991): An Introduction to Dynamic Concept Systems. In: Boley, H., Richter, M.M. (eds.), Processing Declarative Knowledge. Proceedings of the PDK-91 at Kaiserslautern, Springer Verlag, Berlin (Lecture Notes in Artificial Intelligence 567), 87-106

Jaeger, H. (1992): A Type-free Semantics for Concepts in Contexts. In: Brézillon, P. (ed.): Proceedings of the IJCAI-93 Workshop on Using Knowledge in its Context. Technical Report LAFORIA 93/13, Institut Blaise Pascal, Université Paris VI et VII, 1993, 51-61

Kamlah, W., Lorenzen, P. (1972): Logische Propädeutik. Bibliographisches Institut, Mannheim/Wien/Zürich 1972 (B.I. HTB 227)

Kobsa, A. (1991): Utilizing Knowledge: The Components of the SB-ONE Knowledge Representation Workbench. In: Sowa, John (ed.) (1991): Principles of Semantic Networks. Morgan Kaufmann, San Mateo, 457-486

Krause, W. (1989): Über menschliches Denken - Denken als Ordungsbildung. Zeitschrift für Psychologie 197, 1989, 1-30

Krohn, W., Küppers, G., Paslack, R. (1987): Selbstorganisation - Zur Genese und Entwicklung einer wissenschaftlichen Revolution. In: Schmidt, S.J. (ed.) (1987): Der Diskurs des Radikalen Konstruktivismus. stw 636, Suhrkamp, Frankfurt/Main, 441-465

Lakoff, G. (1987): Women, Fire, and Dangerous Things. The University of Chicago Press, Chicago 1987

Langacker, R.W. (1987): Foundations of Cognitive Grammar, Vol. 1. Stanford University Press, Stanford 1987

Lashley, K.S. (1951): The Problem of Serial Order in Behavior. In: Jefress, L.A. (ed.): Cerebral Mechanisms in Behavior. Wiley, New York 1951, 112-136

van Leeuven, C. (1990): Perceptual Learning Structures as Conservative Structures: Is Economy an Attractor? Technical Report 45/1990 of the Research Group "Mind and Brain" at the Center for Interdisciplinary Research (ZIF), University of Bielefeld 1990

Lehmann, F. (1992): Semantic Networks. In: Lehmann, F. (ed.) (1992): Semantic Networks in Artificial Intelligence. Pergamon, Oxford 1992, 1-50

Lightfoot, N., Shiffrin, R.M. (1992): On the Unitization of Novel, Complex Visual Stimuli. Proceedings of the 14th Annual Conference of the Cognitive Science Society, Erlbaum, Hillsdale N.J., 277-282.

Mangold-Allwinn, R. (1991): Flexible Konzepte: Modelle, Experimente, Simulationen. Habilitation thesis, University of Mannheim, 1991

el Manira, A., Cattaert, D., Wallén, P., diCaprio, R. A., Clarac, F. (1993): Electrical Coupling of Mechanoreceptor Afferents in the Crayfish: A Possible Mechanism for Sensory Signal Transmission. Journal of Neurophysiology, Vol. 69, No 6, 1993, 2248-2251

Maturana, H.R. (1978): Kognition. In: Schmidt, S.J. (ed.) (1987): Der Diskurs des Radikalen Konstruktivismus. stw 636, Suhrkamp, Frankfurt/Main, 89-118 (= translation from: Cognition. In: Hejl, P.M., Köck, K., Roth, G. (eds.), Wahrnehmung und Kommunikation. Frankfurt/Main 1978)

Maturana, H.R., Varela, F.J. (1984): El árbol del concocimiento. English: The Tree of Knowledge: the Biological Roots of Human Understanding. Shamhala Press, Boston 1987. German: Der Baum der Erkenntnis: die biologischen Wurzeln des menschlichen Erkennens. Scherz, Bern/München 1987; reprinted: Goldmann Verlag, 1992[4].

Mehl, S. (1992): Dynamische Semantische Netze: Zur Kontextabhängigkeit von Wortbedeutungen. Dissertation thesis, University of Koblenz-Landau at Koblenz, 1992

Minsky, M. (1985): The Society of Mind. Heinemann/Picador, London 1988

Nebel, B. (1990). Reasoning and Revision in Hybrid Representation Systems. Lecture Notes in Artificial Intelligence 422, Springer Verlag, Berlin/Heidelberg/New York 1990.

Nebel, B. (1991): Terminological Cycles: Semantics and Computational Properties. In: Sowa, J. (ed.) (1991). Principles of Semantic Networks. Morgan Kaufman, San Mateo, 331-361

Newell, A. (1980a): Physical Symbol Systems. Cognitive Science 4, 1980, 135-183

Newell, A. (1980b): The Knowledge Level. Artificial Intelligence 18, 1982, 87-127

Nilson, N.J. (1994): Teleo-Reactive Programs for Agent Control. Journal of Artificial Intelligence Research 1 (1994), 139-158

Nourbakhsh, I., Mose, S., Becker, C., Balabanovic, M., Gat, E., Simmons, R., Goodridge, S., Potlapalli, H., Hinkle, D., Jung, K., van Vactor, D. (1993): The Winning Robots from the 1993 Robot Competition. AI Magazine, Winter 1993, 51-62

Obermeyer, K. Ritter, H., Schulten, K. (1990): A Principle for the Formation of the Spatial Structure of Cortical Feature Maps. Proceedings of the National Academy of Sciences of the USA, Vol. 87 (1990), 8345-8349

Patel, M.J., und Schnepf, U. (1991): Concept Formation as Emergent Phenomena. Arbeitspapiere der GMD 602, Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin 1991

Piaget, J. (1968): Le structuralisme. Presses Universitaires de France, Paris 1968. (German translation: Der Strukturalismus. Walter-Verlag, Olden/Freiburg 1982[2])

Port, R.F. (1990): Representation and Recognition of Temporal Patterns. Connection Science, Vol. 2, Nos 1/2, 1990, 151-176

Prigogine, I., Stengers, I. (1980): Dialog mit der Natur. Piper Verlag, München 1981

Quillian, M.R. (1968): Semantic Memory. In: Minsky, M. (ed.): Semantic Information Processing. MIT Press, Cambridge, Mass., 227-270

Reiter, R. (1980): A Logic for Default Reasoning. Artificial Intelligence 13 (1980), 81-132

Sandewall, E. (1993): Nonmonotonic Temporal Logics and Autonomous Agents: Each Contributes to the Rigorous Basis for the Other. In: Herzog, O., Christaller, Th., Schütt, D. (eds.): Grundlagen und Anwendungen der KI (= Proceedings of the "17. Fachtagung für KI", Berlin 1993), Springer Verlag, Berlin/Heidelberg 1993

Schank, R.C. (1982): Dynamic Memory. Cambridge University Press, Cambridge 1982

Schwefel, H.-P. (1977): Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie. Birkhäuser, Basel/Stuttgart 1977

Scott, D.S. (1978): Lambda Calculus: Some Models, Some Philosophy. In: Barwise, J., Keisler, H.J., Kunen, K. (eds.): The Kleene Symposium. Studies in Logic 101, North Holland, Amsterdam 1980, 232-265

Seidenberg, M.S., Tanenhaus, M.K., Leiman, J.M., Bienkowski, M. (1982): Automatic Access of the Meaning of Ambiguous Words in Contexts: Some Limitations of Knowledge-Based Processing. Cognitive Psychology 14 (1982), 489-537

Shastri, L., Ajjanagadde, V. (1992): From Simple Associations to Systematic Reasoning: A Connectionist Representation of Rules, Variables, and Dynamic Bindings. To appear in Behavioral and Brain Sciences. Earlier version: Technical report MS-CIS-90-05, University of Pennsylvania, 1990

Shiffrin, R.M. (1992): Short-Term Memory: A Brief Commentary. Research Report 91, Cognitive Science Program, Indiana University, Bloomington, Indiana 1992

Shoham, Y. (1993): Agent-oriented Programming. Artificial Intelligence 60 (1993), 51-92

Sloman, A. (1975): Afterthoughts on Analogical Representations. Proceedings of the First Conference on Theoretical Issues in Natural Language Understanding, Cambridge, Mass., 1975, 178-182. Reprinted in: Brachman, R.J., Levesque, H.J. (eds.) (1985): Readings in Knowledge Representation. Morgan Kaufman, Los Altos 1985, 431-439

Smith, E.E., Medin, D.L. (1981): Categories and Concepts. Harvard University Press, Cambridge, Mass., 1981

Smolensky, P.(1986): Information Processing in Dynamical Systems: Foundations of Harmony Theory. In: Rumelhart, D.E., McClelland, J.L. (eds.): Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1, MIT Press, Cambridge, Mass. 1986, 194-281

Sowa, J.F. (1992): Conceptual Graphs as a Universal Knowledge Representation. In: Lehmann, F. (ed.) (1992): Semantic Networks in Artificial Intelligence. Pergamon, Oxford 1992, 75-94

Steels, L. (1993a): Building Agents out of Autonomous Behavior Systems. In: Steels, L., Brooks, R. (eds.) (1993): The "Artificial Life" Route to "Artificial Intelligence": Building Situated Embodied Agents. Lawrence Erlbaum, New Haven 1993

Steels, L. (1993b): A Mathematical Framework for Autonomous Robots. Proceedings of the EWAIC '93, Moscow 1993, 333-334

Steels, L. (1994): The Artificial Life Roots of Artificial Intelligence. To appear in the Artificial Life Journal 1, No 1

Strohner, H., Rickheit, G. (1990): Kognitive, kommunikative und sprachliche Zusammenhänge: Eine systemtheoretische Konzeption linguistischer Kohärenz. Linguistische Berichte 125/1990, 3-23

Suchman, L.A. (1987): Plans and Situated Actions: The Problem of Human-Machine Communication. Cambridge University Press, Cambridge, Mass., 1987

Treisman, A. (1986): Properties, Parts, and Objects. In: Boff, K. R., Kaufman, L., und Thomas, J.P. (eds.): Handbook of Perception and Human Performance, vol. 2. John Wiley and Sons, New York, 1986, chapter 35

Vera A.H., Simon, H.A. (1993a): Situated Action: A Symbolic Interpretation. Cognitive Science 17 No. 1, 1993, 7-48

Vera A.H., Simon, H.A. (1993b): Situated Action: Reply to Reviewers. Cognitive Science 17 No. 1, 1993, 77-86

Vera A.H., Simon, H.A. (1993c): Situated Action: Reply to William Clancey. Cognitive Science 17 No. 1, 1993, 117-133

Wachsmuth, I. (1989): Zur intelligenten Organisation von Wissensbeständen in künstlichen Systemen. Habilitation thesis, University of Osnabrück. Published as IWBS Report 91, IBM Institute for Knowledge-based Systems, Stuttgart 1989

Wachsmuth, I. (1994): The Concept of Intelligence in AI. Proceedings of the conference "Prerational Intelligence - Phenomenology of Complexity Emerging in Systems of Agents Interacting Using Simple Rules" at the Center for Interdisciplinary Research (ZIF), Bielefeld 1994, 121-132

Waltz, D. L., Pollack, J.B. (1985): Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation. Cognitive Sience 9, 51-74.

von Weizsäcker, C. F. (1985): Aufbau der Physik. Hanser Verlag, München/Wien 1985

Wilks, Y. (1975): A Preferential, Pattern-Seeking Semantics for Natural Language Inference. Artificial Intelligence 6, 1975, 53-74

Wolfram, S. (1984): Universality and Complexity in Cellular Automata. Physica 10D, 1984, 1-35

Wolfram, S. (ed.) (1986): Theory and Applications of Cellular Automata. World Scientific, Singapore 1986

Wrobel, S. (1991): Concept Formation in Man and Machine: Fundamental Issues. Arbeitspapiere der GMD 560, Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin 1991

Yao, Y., Freeman, W.J. (1990): A Model of Biological Pattern Recognition with Spatially Chaotic Dynamics. Neural Networks 3, No. 2, 153-170

Zuse, K. (1975): Ansätze einer Theorie des Netzautomaten. Nova Acta Leopoldina, Neue Folge, Nr. 220 (Band 43), Deutsche Akademie der Naturforscher Leopoldina, Halle/Saale 1975

Symbol index

Subject Index

page numbers in italics refer to formal definitions