# ADVANCED PATTERN THEORY FOR PATTERN ENGINEERS

## Chapter 5 of the Anti-thesis

***Kent D. Palmer, Ph.D.***

P.O. Box 1632
Orange CA 92856 USA
714-633-9508
palmer@exo.com


Version 0.05; 02/10/26; sd04v05.doc

**Keywords:**

Pattern, Schema, Meta-system, System, Process, Engineering

**Summary:**

None yet.

## Worlds and Patterns

The worlds schema and the Pattern schema together have an internal resonance. This is because they define the limits of experience on both ends of the spectrum at least with regard to the schemas. We could define everything within the world as patterns if we so wished. For instance, Ben Goertzel does this in his book Chaotic Logic. For him everything is a pattern and all higher schemas within experience are reducible to this schema. However, we do not subscribe to this reductionist view. Instead we think that each schema is an emergent threshold of organization with its own unique properties that cannot be reduce to the other schemas. But certainly we must recognize that this is one possible way of looking at things which has a fair appeal to many theorists. For instance, Lawson's Closure could be interpreted in this way. We talk about fabric as material, and we talk about the texture of such a material and the openness of its weave. In other words his terminology suggests he has something like the pattern level in mind when he conceptualizes his theory of Closure in relation to openness. Focusing on pattern is a standard way to produce an universal theory. It avoids the pitfalls of focusing too much on Form, and one does not have to consider System or any of the higher level schemas which complicate things seemingly unnecessarily. In the Nineteen Seventies this ruse was pursued under the name Structuralism. Structuralism seeks to produce a theory at the level of the content of forms without considering the forms themselves. Levi Strauss was the major theorist in this movement which was summarized in The Savage Mind. There is now a History of Structuralism to which one might refer to reprise this intellectual movement. Structuralism moved from the hard sciences into the social sciences and out to effect most disciplines among the humanities, including Systems Theory. Goerge Klir's work Architecture of Systems Problem Solving is an excellent example of a General Systems Theory that takes into account the Pattern level. It has already been explained how his work might fruitfully be compared with Baudrillard's Critique of the Economy of the Sign. By combining the two theorists work we produce a more comprehensive theory of pattern that can be seen as having at least four different kinds, value, sign, structure and flux. Each of these are composed of discontinuities. Structure is a discontinuity of space of contents. Flux is a discontinuity of the time of contents. Value is a discontinuity of the Utility of contents. Sign is a discontinuity of the meaning of contents. These four different kinds of patterning may have chiasmic relations with each other of the type that Klir discussed in his epistemological hierarchy and Baudrillard hinted at. We use this idea as a basis for a deeper theory pattern theory which takes into account all four kinds of discontinuity at the level of content. Baudrillard only considered the difference between economic value and social significance of commodities. Klir only considered the actual deformations of the space of data contents in terms of space and time that could be construed as a Formal Structural System to produce an architectonic of the system. But note that the commodity may be seen as the thing within the meta-system of the general political economy and that it can also be seen as a signifier within the social meta-system as well. But the commodity can also be one of the things within the system, as a figure on its ground, that are related to other things within the system which can be seen architecturally in terms of space and time discontinuities of content. So the two theories fit together very conveniently. It is interesting to note that Jung defines the psyche as the conjunction of signification and value. In his later theorizing Jung also introduces the psychoid which are objects outside the subject that mirror its own processes and express archetypes via synchroniety. We might see that psychoid realm in terms of patterning of discontinuities in space and time. Thus it is interesting that Jungian theory relates to these theories of commodities on the one hand and formal structural systems on the other. The Jungian theories gives us a way to understand how the other two theories might be seen from a psychological perspective rather than an external objective perspective. When we recognize that the Unconscious might be treated as the Meta-system, the general economy of Bataille, in relation to consciousness or the system, the restricted economy of Bataille, then it is possible to see how a theory of patterning of the psyche/psychoid in terms of associations of words, feelings, images, concepts etc. can mirror our relation between the contents of spacetime and the contents within our consciousness.

This is a point where it might be wise to begin to introduce the ideas of Deleuze and Guattari from Anti-Oedipus and Thousand Plateaus. In that book there is a devaluation of the individual organism by the focus on a different ontological level called desiring machines. Desiring machines are contrast to the Socius which is a social field. Desiring Mahcines are defined as partial objects in the sense of M. Kline. They are said to exist in patterns of flows across bodies and to make up a unstructured mass called the rhizome which is not hierarchical and has no beginning or end. The rhizome is something we are always in the midst of. What is unique about Deleuze and Guattari's treatment of Desiring machines is that they are seen as orthogonal to each other exemplifying directly the unconscious, which is named the "body-without-organs". The desiring machines hang off of the body-without-organs like the medals of a general at a military parade. Like the theories of Klir and Bataille we want to modify this theory somewhat. We want to introduce the opposite of the desiring machine which we might call the ***avoiding*** machine. These together are opposite another pair that might be called the ***disseminating*** machine and the ***absorbing*** machine. When Deleuze and Guattari say "machine" they mean something that is prior to the split between organism and machine, something on the order of dasein. But it is very disconcerting to continue to use the term machine in this context, so I would prefer to use the term practice following Foucault. Thus we can see that there are desiring practices, avoiding practices, disseminating practices, and absorbing practices, and our point here is that these practices operate on the pattern level which may produce structures, fluxes, values and signs. When we combine the concept of these four practices with the concept of the four kids of pattern I think we get a very powerful theory of patterning which might serve as a basis for understanding patterning engineering as the practical result of patterning theory. Of course, the formal theory that we will appeal to in order to define pattern theory is Grenander's Theory of Patterns. Grenander has the only known mathematical theory of patterns known by the Author. It allows us to be very precise about the nature of Pattern Theory when combined with the ideas of Klir in his Architecture of Systems Problem Solving. But these theories don't allow us to develop the practical reason related to pattern theory as easily as we can if we combine Deleuze and Guattari's theory of the desiring machines and Foucault's theory of practices. In a way this is the bedrock of our schemas theory, because this is the level at which schemas first come into contact with experience at the smallest increment. Much of the critique that leads to this is that of William James who noted that there is a specious present, i.e. that content itself as isolated is really an abstraction and we never experience it as monads. So Husserl's *hyle* as the object of the *intentional morphe* is just a concept with no experiential determination. This is one of the things that drives Gurwitch to explore the fringes of consciousness and amorphousness in consciousness. Monads of content of consciousness are just never seen in isolation even though we project them, as Leibniz so famously did. So we see patterns in things and we relate to these patterns through our practices that can be desiring, avoiding, disseminating, and absorbing. We do not see monads, or their facets in consciousness except as elements of our representations.

We place the patterns we see and produce within the context of the world. If we only had these two schemas we could construct a good theory of how we relate the patterns we see to the world in which we live, our lifeworld that we relate to as dasein made up of patterns of practice which were not yet broken into subject and object. Of course we would like to extend this theory to the level of Hyper Being and Wild Being which we would do by projecting monads and facets or by projecting kosmos and pluriverse beyond experience of patterns contextualized within the lifeworld. However, we would also have to project forms and domains in order to represent Pure Being, as both the world and pattern are established at the level of Process Being. That is in Process Being we have these ready-to-hand proto-things called patterns in the world. We call the ready-to-hand a mode of being-in-the-world of dasein. Dasein is proto-subject/object. The understanding, talk and discoveredness of Dasein is in terms of patterns within the world. That is patterns of understanding, i.e. schemas, patterns of talk that describe what is seen via the schemas, and patterns of discovery which goes beyond the schemas, as in science that recognizes anomalies and reorganizes emergently to deal with those anomalies in paradigm changes, episteme changes, and reinterpretations of Being. Heidegger's phenomenology is a study in the pattern formation prior to the arising of the subject/object forms embedded in the world. Foucault's Order of Things is a study of epistemic patterns. Kuhn's Structure of Scientific Revolutions is a study of paradigmatic patterns, i.e. patterns of assumptions that lie at the roots of theories. All of these patterns exhibit emergent effects of the nature of the emergent events described first by G.H. Mead. One way to attempt to understand these emergent effects is to study as Wolfram does in his A New Kind of Science cellular automata which exhibit the emergent effects of rule changes and the way local effects generate global patterns. Cellular Automata Theory is a prime example of how Grenander's generalized mathematical theory of pattern might be implemented and operationalized. It gives us small worlds of pattern production and recognition that we can experiment with in order to try to understand global pattern formation based on local rules. Rules are sign structures that set values of the cell color. The cells exist in a lattice which is stepped through time. The pattern is seen because of the discontinuities that are produced as the rules fire in the entire field of the cellular automata network over the time steps producing color values that we apprehend globally in spite of the local influences that produced them. The production or dissemination of the color patterns that evolve in the cellular automata is balanced against our absorption of them. The desire for titillation is what drives our continued viewing of the scene. In order to see that scene we must not look too closely, or stop the clock, but must instead look across the whole field concentrating on those fleeting designs that catch our fancy. Our interaction with the Cellular Automata is a fusion of the various practices we have described as the Cellular Automata itself is a fusion of the various kinds of pattern we have discussed. It is a prime example because it brings together in a single mechanism all the various aspects of patterning devices and patterning related practices. Wolfram's point that complexity of patterning can be understood in terms of software programs that execute rules in A New Kind of Science is well taken. This is supported by Goertzel's idea in Chaotic Logic that we should seek the smallest possible algorithm to produce a given pattern and by Grenander's Mathematical Patterns Theory of which is a sort of abstraction of cellular automata made up of a lattice or network of generators. Many theorists are converging on a similar understanding of the fact that pattern is a basic schema for understanding the organization of things.

## Patterns and Things (and Stuff)

Something should be said about the relation between patterns and forms of things. Forms have shape which encompasses our surrounds or serves as a basis for content. Content here means the "sensory data" which fills up the form, or covers the form, or surrounds the form, depending on the kind of form we are talking about. This sensory data is qualitative for the most part. Usually in science the primary qualities are associated more closely with the form itself, such as position, mass, velocity, etc. But we can separate the quantitative definition of the form as something countable from most of the qualitative aspects of the content of the form. The content is what is patterned. There are cases like a bolt of cloth where the raw material lacks any specific form but acts as a media for the content. Media are broadly speaking a tabula rasa for certain types of content and forms. Sometimes as with cloth the content comes as part of the media, while in other cases the media is imprinted with the content at the same time it is imprinted with the forms, as in photography for example.

However, all this assumes we are dealing with countable things, but masses also have content. The body of the mass, its instances are generally thought of as the content of the mass. Masses overall have global forms, say a surface upon which we see waves. But generally in masses the content is emphasized over the form. We might also speak of the ipsities that are juxtaposed in conglomerates were form and content might be merged, say when we talk about molecules whose forms are determined directly by their contents. In any case it is important to keep separate in our minds the patterning of content verses the form, even though we can choose to see the form as just a higher level of content patterning. Normally the form and content are separable as in a glass and the decoration on the side of the glass. But there are many elements in which the form is made up of the arrangement of the content, as in many famous Dali paintings, or in faces that are made up by arrangements of fruit, and the like.

Apparition of Face and Fruit Dish on a Beach -- Salvador Dali -- 1938

Giuseppe Arcimboldo -- Summer -- 1573

These instances are of course anomalous, and they are the exceptions that prove the rule concerning the orthogonality of form and content.

### Structure and Organization

Maturana and Varela in their theory of Autopoiesis distinguish organization from structure. Organization is a persistent pattern which overrides the shifting of content at some lower level of abstraction called structure. This raises the issue of interembedding of pattern within pattern, and the fact that on level of pattern may be in flux while another higher level of pattern may be maintained as persistent, even though the actual content being organized is shifted out and been replaced. Each schema can be hierarchically embedded in itself, so there may be a nesting of forms like Russian Dolls. Likewise there may be a hierarchy of patterns within patterns within patterns which is fractal. The interesting thing is that both interembedded forms and patterns may have fractal dimension.

## Teleology and Teleonomy

Jacques Monod distinguishes between Teleology and Teleonomy. Teleology is knowing the goal before hand whileTeleonomy is a goal that is discovered by approximation along the way. One of the four causes of Aristotle is the Final cause. Another is the formal. Then there is the material and efficent causes. These four causes may be related to the kinds of Being:

Pure Being = Formal Cause

Process Being = Efficient Cause

Hyper Being = Material Cause

Wild Being = Final Cause

Here the material cause is the content which may be patterned. Form is a separate cause. Efficiency is what relates the Form to the Material Content. The Final Cause is the end which is assumed to be pre-given. Teleonomy is when the cause is discovered n the process which Monod says can happen if there is both chance and filtering working together. Monod's theory is an excellent example of a Formal Structural theory because the structures within the hierarchy of chance and necessity is seen as what produces the teleonomy of evolution. It is interesting how difficult it is to think of another kind of cause at this same level of abstraction other than those that Aristotle names.

## Pattern Language

Alexander's Pattern Language[1] has become a famous way to approach software design. It started out as an approach to architectural building and town planning. What we can say about the pattern language is that it is an inversion of the schema, in the sense that we normally think of patterns of content applied to or within forms as shapes. But the pattern language deals with patterns of forms at various hierarchical levels of the built environment. This is an excellent case to see how schemas can interact in surprising ways where a lower level schema like pattern can be used to understand the ordering of higher level schemas like form. Alexander evidently goes on in his new books[2] to talk about sequences for applying patterns to building projects. He is looking at unfolding generative sequences by which patterns are built up, this could be similar to the lattices of generators talked about by Grenander. Alexander makes the interesting point that the number of sequences are exponential but the number of sequences that work without backtracking in a particular context are very few comparatively. This idea of generative unfolding of sequences to produce patterns are exactly the theme that I would like to pursue. I call this the meta-essence which is at the level of Hyper Being up from the essences that exists at the Process Being level. Meta-essences are timing constraints or essential constraints of attributes. Meta-essences determine genetic unfolding of things like patterns, or forms etc. Any schema can fit over something that genetically unfolds. The schema includes the higher meta-levels that appertain to it. For instance in pattern at the meta-level two has categorization as its essence. At meta-level three there is a spectra. At meta-level four there are singularities. What is meant by this. As Schlegel says in his aphorisms categorizations are definitions of definitions. A categorization is a classification into taxons of the various contents that are found in a pattern. All patterns can be reduced to categorizations by the postulation of monads and then the classification of those monads

[1] http://patternlanguage.com

[2] The Nature of Order Series

and their various groupings synchronically and diachronically or in term of value and signification. The categorizations with respect to patterns are similar to the rules with respect to systems. The categorizations are the essence of the pattern. But at the Hyper Being level where the properties exist in systems we see in patterns spectra. Spectra means that the various categorized contents produce a pallet say of colors or shapes or tones out of which the pattern is produced. Categories of contents of a pattern are given dynamic meaning by their array in spectra that are constructed out of these categorized contents. What is important here is that at the level of pattern there does not have to be any continuity of the kind that produces forms, in fact there does not have to be any continuity of media either. In other words for patterns continuity is not essential, and in fact we define patterns as discontinuities in space, time, value and signification. It is the discontinuities that are abstracted to produce the categories, meta-definitions. Sometimes these discontinuities are such that they are actually singularities, i.e. anomalies that cannot be seen to fit in the patterns at all. Patterns it must be stressed are like gestalts in the eye of the beholder. They are projected schemas. Of course when we categorized the pattern contents we are imposing a set-like countability on the mass of the contents of the pattern. But we would prefer to think of the contents as ipsities that are juxtaposed into conglomerations which are seen as patterns. The patterns may be focused on discontinuities of space, time, value or signification, or chiasmic combinations there of. When we attempt to categorize the contents and realize their spectra we are really treating them like sets. On the other hand if we merely take in the contents in a global picture we are considering them as a mass. But if we consider each element of the pattern in its uniqueness both as content and in relation to its neighbors spreading out to further and further neighbors then we are considering the pattern in its ipsity.

## Patterning Patterns

The archetypal pattern media is cloth. Into its warp and weft by the intertwining of threads patterns have been created by women over millennia. So much so that in Greek myth the archetypal roles of women were weaving and water gathering at the well. Both of these activities are related to the primal scene of the Indo-Europeans called the Well and the Tree[3]. It is interesting that the first programmable device was the Jacquard loom which could be programmed to produce patterns in cloth being woven by machines. It was the Jacquard loom that was the basis of Babbage's programmable calculating machine which was never completed. Ultimately the programmable calculating machines were produced as electronic gadgets rather than mechanical ones which achieved universal computability. These were described by the Turing machines. If we look a Turing machines and their duals, Universal Turing machines, we see that these machines have tapes. One end is infinite and the other is finite. On the tape are readable and writeable signs. Each place on the tape has a value from among the possible signs perhaps represented by the pattern of bits that are turned on in a sequence of bits, for instance like the ASCII code. So the place that patterns show up in computing is as the pattern of bits on the tape of the Turing machine. The ASCII code relates these bit patterns to the forms of letters, numbers, punctuation, and various symbols. Note that this translation from the patterns to ASCII equivalents actually happens in our heads, the computers actually only work with the bit patterns and they are oblivious to the meanings we project by coding onto these bit patterns. So we see there that the transition between bit patterns and forms of letters in the ASCII code occurs not in the computer itself but in our interaction and interpretation of what the computer is doing and our

[3] See The Fragmentation of Being and the Path beyond the Void by the author.

interpretation of pixels on its screen. But that transition from pattern to form is well defined in the Turing machine representation and in the architecture of the general purpose computers that we interpret as Turing machine equivalent. In fact, we can see that the transition from form to system is also well defined, because the whole Turing machine is a system where as the symbols are forms. So there is another well defined transition between pattern to form to system. The same can be said for the transition to the meta-system that is equivalent to the universal Turing machine. That is a Turing machine that has other Turing machines represented on its tapes which it loads and executes thus being able to change from simulating one Turing machine to another as a meta-program. What is so strange about the meta-system is that it is exactly the same as the Turing machine except the program that it is executing is itself a Turing machine not merely an algorithm, but rather an application to which the universal Turing machine is acting as an operating (*meta-)*system. No wonder we have difficulty seeing the difference between systems and meta-systems, if their difference is so subtle. Note that the distance between pattern and form is less than the difference between form and system, note that the difference between meta-system and system is less than the difference between pattern and form. These measures of difference between schemas show us that the schemas are themselves very different in how they interact and nest with each other. Ultimately our model of pattern is bits in a computer. They have a sort of ultra efficiency because they can be set and reset so easily, and fit into such a small space that it allows great leverage of computing power such as we have been experiencing through the playing out of Moore's law over the last twenty years or so. Babbage's mechanical computer eventually led to IBM sorting machines that were controlled by electronics. But the real leap was when computing machines were developed that had generalized architectures that made general purpose computing possible. All this relies on the recognition of electronic bit patterns, and execution of operations based on that input to produce an output pattern of bits. This is of course the functional model of computing in which there is input and output through functions implemented in software. Machines deal with the patterns of bits and we humans interpret the inputs and outputs in relation to forms and symbols. This is what produces the whole problem of Artificial Intelligence which is that the machines do not really understand what they are processing unless we build in some sort of knowledge into them, and then still they do not really understand that knowledge but only mimic such an understanding. Turing's test for Artificial Intelligence is that we ourselves cannot tell the machines are faking it.

Now why is it necessary for machines to fake their understanding of forms out of patterns. The reason is that the machines cannot comprehend the transition between schemas. For a computer everything happens at the bit level of patterning, everything above that level is an artifice, the symbols of the computation are an artifice, the system of the program is an artifice, the meta-system of computing, i.e. the operating system is also an artifice. By artifice is mean a sterile construction that does not involve its own synthesis. You see discontinuity is built into the schema of patterning. This discontinuity is inherited as we move up the various constructed levels of schemas. So the computer does not really have the continuities that allow us to comprehend form. That would require something analog, and the computer is by definition digital. Its digitality is that inherent discontinuity that is inherited from the pattern level and persists in all higher levels in spite of their needing some other basis for synthesis in order to be genuine representations of schemas such as we as humans possess. So computer programs will always be elaborate illusions at the schema levels higher than pattern. The question is whether we can tell they are illusions with our in-built capability for understanding the schemas based on our organically based continuums that provide a staging area for our own understanding of the schemas. We could argue that our own continuities by which we recognize forms and systems and even meta-systems are also in some sense illusions created by our neuro-circuits. This is in fact what many of the findings of contemporary cognitive science point toward. But as we live within our own illusion and it seems real to us, then the question is whether other illusions conform to our illusions which must remain for us the touchstone of our ratification of experience.

An interesting example is Douglas Hofstadter's book on analogies[4]. In that book he describes Melanie Mitchell's[5] Copycat program, the architecture of which attempts to understand very simple analogies. It is interesting how much programming work is necessary to do even this very simple task. The architecture of copy cat has some things in common with Holland's Complex Adaptive Systems[6] in which there are multiple agents working in an environment together. Many such simulations are now being produced under the rubric of artificial life[7]. This is a whole realm of inquiry that we will leave aside because here we are trying to understand the nature of the schemas themselves and are not so interested in their expression in various fields. But the key point made above is that there is a fundamental level of discontinuity that are the basis of all these simulations, of life or intelligence or the social[8] which cannot be overcome because the illusory continuities on which the synthetic comprehension of the schemas are built in our human organism and society is missing in the simulations. All this comes from the fact that they are based on patterns that build up simulations of higher level schemas but do not leave the discontinuities associated with the pattern level behind. These discontinuities between bits in memory, or bytes, etc are fundamental and all continuities that are projected at higher levels are merely illusory because these gaps cannot be closed by analog synthesis such as those built into our physiology and our social matrix which makes us organisms not machines.

## Variables and Values in Time and Space

If we are going to understand patterns in any fundamental level then, we must understand the value in the variable as it is situated in time and space. This means going back to basics and looking carefully at the variable which is a piece of computer memory accessible to software. That variable as we know has a pattern of ones and zeros among its bits. That variable is external to other variables in memory. Here memory is space and cpu cycles are time. This externality and non-relation between variables is the deadly aspect of discontinuity in space which is mimicked by the discontinuity in time of the cpu cycles governed by the program variable that tells what line of the program is being executed at any one time. So we have discontinuity in both space and time which we attributed to the pattern schema in the architecture that guarantees the separableness and continuity in hardware of the variable as a memory location. But let us also look at the values that the variables assume. These values are conceptually distinct from each other, determinably so. As is their significations to us as programmers and users of computers programs. In other words there are two pieces to this puzzle. There is one in the hardware and software of

[4] Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought
[5] http://www.cse.ogi.edu/~mm/
[6] cf Emergence (Addison-Wesley, 1997), Hidden Order: How Adaptation Builds Complexity (Perseus, 1996)
[7] http://biota.org/
[8] http://jasss.soc.surrey.ac.uk/JASSS.html

the computing device, and there is another in us. Value and Signification are in us while Structure and Flux are in the computer. We associate the value with the pattern of bits in the variable by inventing a code like the ASCII code. We associate the signification of the input and output of the function by a projection of meaning onto the behavior of the computer program. But in both cases there are externalities. Inventing codes and computer science structures is separate from the projection of meaning on the behavior of the program. There is this fundamental split between design and implementation on the one hand and execution and validation on the other. So not only is there absolute space and time separated in the computer in a Kantian style as memory and cpu cycles, but there is also a radical separation of these from values and signs which are in turn separated each other. These separations are continually glossed over but they are crucial because they make the essential interaction between humans and computers a patterning, but they mean that what is going on at the most fundamental level in both the computers and the humans as they interact with computers. For artificial intelligence to occur the valuing and the signification would have to occur within the computer itself rather than as merely a human projection. Let us consider that Jung defined the psyche as our valuing and signification that occurred inside ourselves. Since we do not know how we do valuing and signification then it is going to be difficult to figure out how to project those fundamentals of our cognition outside of ourselves. Also let us just remind ourselves that Nietzsche's great innovation was to ask about the value of value. Similarly IA Richards asked about the meaning of meaning. We do not know very well what these meta-level cognitions are even in ourselves.

But let us pretend that we understand at least the structure of present day computers and our own discrete concepts by which we code them and the discrete states that they take as they execute. Looking at these externalities we see that all four kinds of pattern come together at the executing variable that is used to simulate a turing machine. We can quickly see that there are two types of relation between variables. One is that we can see many of them at a time changing with values running through them in parallel in which we see a global pattern. The other is that we can see values moving along from variable to variable perhaps transforming as they move in a way that we recognize as a local pattern.

There is a similar thing on the value and signification side. We can construct a hierarchy of values where one value decomposes into another value at a lower level. There can be state transitions between these values at either level. We can imagine that there are higher level states that are connected with lower level transitions OR there are lower level states that are connected with higher level transitions. By considering two levels at once we note that there are these dual ways of looking at the state space with its transitions that produces a tension between them which are dual views of the whole containing both levels. If we think of signification as something that comes from a reference from one thing to another then we can see these duals as two ways to produce such a reference. These are the four ways that variables can come into conjunction with hierarchical state networks. And these taken together are also a simplified version of the Turing machine. A Turing machine is a state machine and a stack. The state machine is executed on the stack. Here the two views of the hierarchical state space (higher states with lower transitions OR lower states with higher transitions) are ways that value and signification might be married. When lower level transitions occur what is the higher level state that is produced? When higher level transitions fire what are the lower level input and output states that are gathered together in that function? Similarly we can look at global values of variables across a set of variables, or we can trace values from variable to variable locally. These are the four possibilities for breaking up the Turing Machine in terms of its variables and its state networks. What is interesting is that if we take these in pairs then they underlie the minimal methods at the formal level and provide us with slices of a Turing machine. The minimal methods are based on the pairwise composition of these fundamental elements for modeling data and event and because of that the minimal methods become slices of a Turing machine. This is what makes them able to represent in simplified form the basic structures of realtime software systems[9] and systems in general.

If we go back to Klir's structuralism of General Systems we will see that it is made up using the building blocks of variables in which data are carried, the changes in these data are seen as events. The combinatorics of variables gives us the fundamental architectural possibilities of the system because the variables can represent the inputs and outputs of the system. What we miss in this is the fact that outside of the structural system reduced to variables there is the value and sign shadow that is seen in the human response to that system, which shows up as process.

## Energy, Matter, Information, Entropy

At this point it behooves us to discuss the relation to this nexus of the basic processes of the fundamental elements of physics and thermodynamics, of Being and Becoming. When we look at the hardware of a computer we immediately see the relation between energy and matter as the physical presence of the machine. We are of course talking about Electromagnetism which was the great discovery that made computers feasible in a way that completely mechanical computers were not as the experience of Babbage showed. Computers are very complex deterministic electromagnetic gadgets. They are information processors. They can take many data streams and compare and contrast them to wring information from the data and present that information to humans via various human-machine interfaces. The key point is how entropy is seen in the working of these gadgets. Entropy is not so easily spotted. It takes quite a bit of human work to get data into and information out of computers. The entropy is mostly in the interface between the human and the computer. Computers are the great wasters of time. They appear to make things better, until you realize how much down time they entail when they are broken, or working wrong, or giving out garbage because they were fed garbage. The gain in efficiency that comes with using computers is offset by a loss from the disorder that they create as the environment needs to be changed to accommodate them, or from technological incompatibilities, or from other sources of lost effort. There is also some entropy in the actual degrading of the computer system itself, and the waste of power that is needed to run them, but this is minor compared with the entropy of the human-machine interface.

So if we can locate the four elements of information, entropy, energy and matter, then it falls upon us to think about there opposites. Negative Entropy is seen when a computer program takes input data and introduces order to it so that the output is more ordered than the input. Negative Energy and Anti-matter do not appear in the computer explicitly, unless we think about them in terms of virtuality. The computer ultimately produces a virtual world which has its own seeming materiality and its own kind of energy which we might see as the inverse of the energy and matter in our world. We have noted that anti-matter does not appear normally in this universe. It has all been annihilated, and if any anti-matter appeared it would soon vanish. But the anti-matter world is a possibility that is definitely there within our physical world. When we

[9] See Wild Software Meta-systems

produce a virtual environment it is like a production of the shadow of that possibility. We can walk through walls in virtual reality and break all kinds of physical laws. We can try experiments that would be too dangerous or impossible in our own world. Virtuality is a kind of anti-matter and a kind of anti-energy in as much as the virtual world is composed of the shadow of matter and energy of our world. That shadow is different from the anti-matter and negative energy of the real world, but still it is some distant relative of them perhaps. Negative energy appears as attractive or repulsive forces in virtual reality. Negative Matter appears as grids of invisible lines that become solid through software produced constraints. The other element that I would like to talk about is negative information. You never hear about negative information, but we know that, dissimulation, lies and secrets have a value within human society. Negative information such as computer viruses, such as trojans, and mountains of spam exist in cyberspace as well.

Now let us go down to the variable in spacetime with its value that has signification for someone. In order to talk about these elements we need to take at least two variables at a time, so that the externality becomes significant. Energy appears as electromagnetic force as contained and channeled by the silicon computer chips that are glass matter. Information comes from the comparison of at least two sources of data in the variables of the computer. Entropy comes from wasted heat and from the sheer frustration of trying to interface with anything so small as a computer chip. Negative Entropy comes from the ability of the program to order the information in the variables increasing its complexity and coherence. Negative Information comes from its ability to be out of sync with the environment in some way so that it in fact becomes a lie, deception, or secret. Negative Energy and Negative Matter appears in the mirror world of virtuality that can be produced based on the functioning of the software system in a way completely cut off from the external reality. By this reckoning virtual energy and virtual things are negative energy and negative matter.

Computers have their own internal clocks and thus these clocks can be related to each other relativisticly. But for computers to participate in the realm of quantum mechanics they must become quantum mechanical themselves as David Deutsch suggested in <u>The Fabric of Reality</u>. Such computers operate within quantum superimposition until they are observed and thus have the kind of uncertainty that exists between the particle and the wave.

We have taken the four types of pattern and have connected them to the foundations of computer science by looking closely at variables and their values. But then we have woven around that the concepts of entropy, information, energy and matter that were breached in the last chapter. We have seen how that is a short step to the quantum and relativistic computer. But we must also think about this computer in relation to the logos and the physus. We can see it in terms of physics. But our own organism owes just as much to thermodynamics as to particle physics. The physus is the organisms that grow and develop unfolding according to the plan inscribed into the meta-essence. Opposite this is the logos which is the unfolding of language with the core of logic which is the physus within the logos. We can see the core of the physus as the logos within the physus which we understand as the schemas. Between them stand the nomos, order that is the non-dual that unites them yet separates them. We participate in the unfolding of physus and the unfolding of the logos. We use as our measures the schemas and the logic. Our theories relate to our experiments via the ordering of the nomos. When we put it in that light we see that computers have a long way to go if they are to spread their functions into the realms which we find so natural, which are that of the unfolding physus of the growing organism, the unfolding logos of thought. Just imitating the recognition of objects in the world via the schemas and ordering the processing of information based on logic is not enough. And we have not mentioned the deeper non-duals such as right, good, fate, source and root which show up so naturally in human life as we deal with the fundamental dualisms of our worldview. Patterning is a fundamental way of looking at things. Computer science has made patterning its own as a basis for the construction and running of software. But we can see how narrow the use of computers are in relation to what we ourselves encompass as human beings.

## Aspects of Being and Categories

We are focusing on the variables with values embedded in spacetime which are significant for some observer. We already noted that EnergyMatter represents Being while InfoEntropy represents Becoming in this picture. It is interesting that for a general computer we need to imagine negative EnergyMatter as virtuality. EnergyMatter is a differentiation of the physus. Contrary to that is the logos which is the root of idealism rather than materialism. The most notable idealism is that of Kant, but we could mention Hegel and others. Between materialism and idealism there is a rich spectrum of positions. The key question concerns the relation between the aspects of Being and the duality of physus and logos inscribed in Being that underlies the variable given a value of significance in spacetime. Let us first consider identity. There is difference though time, at each moment the pattern of bits in the variable might change. There is the question of repetition when the same variable appears again in the variable during a different cpu cycle. There is the identity of the pattern in multiple variables at the same time or at different times. There is even the possibility of re-coding in which the same pattern means something different in different contexts, for instance is EBCDIC and ASCII. There are many different contexts for the application of identity and difference within the context of a network of variables with values. Presence also means different things. It could mean the presence of the pattern in the variable. It could mean the presence of the pattern to the program that looks at the variable or changes it. It could mean the presence on the screen or printout of the contents of the variable. There are many different ways that a pattern could be present, and then there is the presence though the mediation of the codes, or other intermediate interpretations by programs. As to truth there are also many different ways to verify the pattern against different criteria in order to establish truth. We can verify against requirements, we can verify one variable against another, we can verify against an output or input. With respect to reality this is normally established by verification with the outside world in some way, i.e. what is beyond the system in which the variable exists in the context in which it operates. Testing is the normal way to establish reality in the context of the environment. We can test many different aspects of the system against many different aspect of the environment. So when we look at the aspects of Being we see that each of them can be applied myriad ways to the simple situation of a small network of variables with value given significance within spacetime by someone. The network of variables gives us a multi-aspectival situation when they operate together. And we need to take seriously this multi-aspectival situation because there is no way to reduce its inherent complexity.

A similar thing can be said for the Kantian part/whole categories. Plurality and Totality and Unity apply in many different ways to the variable with value given significance by someone within spacetime. The pattern can be seen as either totality or unity or plurality depending on the point one wants to make. However, the one thing we would withhold is wholeness. It is hard to see wholeness exemplified in the situation. In fact, the radical separation that is inherent in the computer at the level of patterning that carries on up through the simulation of higher levels of organization seems to bar wholeness from being achieved except as something that we project upon the computational situation.

What we are seeing here are more limitations to the computational milieu which will cause us to insist on its embeddedness in the human environment. This goes against the prevailing cognitive metaphor of the computer as the equivalent to human intelligent processes. We need to take that deficiency in the metaphor seriously as well. The computer appears at one level of the worldview to produce a determinate system (application) and meta-system (operating system) combination. It appears in the context of the deeper levels of duality and non-duality which it does not relate to and it is also multi-aspectival and multi-categorical at the same time still not achieving wholeness except in terms of projections by us.

## Pattern Theory

In this essay we will take Grenander's definition[10] of the pattern algebra as canonical. The definition is composed of different elements that are necessary to produce patterns. The first of these is the generator:

**4.1. Generators.** We shall build representations of patterns from simple building blocks that will be referred to as *generators* and denoted by $g$, $g_1, g_2, g_i, g_i^0 \ldots$ with subscripts/superscripts as needed. The set of generators needed for a particular knowledge representation, the *generator space* $G$, will vary from case to case but at the moment we shall let it be quite general.

A generator is something that produces something else. It does work and thus consumes exergy. Klir has generators in his Formal Structural Systems theory which he describes after the object, source, and data systems. In math generators are functions. However, we can take them to have the meaning of generators as they appear in UNICON as a routine that will produce every possible combination of a pattern. In other words we can take generator in the structural sense which would produce every possible permutation of a given pattern. Grenander is not being so specific but we can extend his definition easily in this way. The generators that Grenander posits are equivalent to the Peirce-Fuller category of Firsts.

To represent *symmetries* and *invariances* of patterns we shall use some group of transformation of $G$ onto $G$ (note 4.1), the *similarity group* $S$ with elements denoted by $s$, $s \in S$. Each $s$ shall mean a bijective mapping $s : G \longleftrightarrow G$, and we shall speak of $s$ as a similarity. We shall assume that the generator index $\alpha$, see below, is $S$-invariant so that $g$ and $sg$ are in the same set $G^\alpha$, $\forall s \in S$.

The key point concerning patterns is that they preserve certain invariances and symmetries that are repeated. Grenander moves in his definition to take care of these symmetries and invariances first. This is equivalent to the Peirce-Fuller category of Fourth which stands for synergy.

Sometimes it is natural to split up the whole generator space $G$ into parts $G^\alpha$, where $\alpha$

82 *Chapter 4: Analyzing Patterns*

is called the *generator index* varying over some space $A$. We shall then have a partition

$$G = \bigcup_{\alpha \in A} G^\alpha$$

into disjoint subsets $G^\alpha$.

This part of the definition allows patterns to have parts which act independently and differently. Generators can be either mass like instances if all the generators are copies of the same generator template, or interpenetrated masses if all the generator instances are of different types but mixed together probabilistically. For generators to approximate set particulars the tags are used to collect the generators of different types whose essence is determined by the sorts of inputs and outputs they will accept, i.e. by

[10] Grenander, Ulf. Elements of Pattern Theory (John Hopkins UP, 1996) pages 81-94 excerpts scanned

their particular attributes. We can think of the generators as an ipsity if we find the middle between the obsession with sameness on the mass side and the obsession with difference on the set side. Generators that are juxtaposed but not connected via their bonds are conglomerated ipsities that are juxtaposed which is the non-dual between the set and mass extremes.

To be able to combine such building blocks, the $g$'s, into larger structures each generator $g$ will possess bonds $b_1, b_2, \ldots b_\omega$ where $\omega = \omega(g)$, the *arity* of $g$, means the number of bonds and can vary from generator to generator. To each bond $b_j$ corresponds a *bond value* $\beta_j$ from some *bond value space* $B$.

Bonds may carry markers, for example labels "in" or "out," to indicate that the bonds are directed in, toward $g$, or out, away from $g$. For any $g \in G$ the notation $B_s(g)$ will mean the set $\{b_j;\ j = 1, 2, \ldots \omega(g)\}$ where $b_j$ means the *bond coordinate* $j$ together with the markers in the bond, if any, and $B_v(g)$ will mean the set $\{\beta_j;\ 1, 2, \ldots \omega(g)\}$. We shall denote by $B(g)$ the combination of bond structure $B_s(g)$ and bond values $B_v(g)$.

The set $B_s(g)$, the *bond structure* is assumed to be $S$-invariant, that is if $g_1$ and $g_2$ are *similar*, meaning that there exists a similarity $s$ such that $g_2 = sg_1$, then $g_1$ and $g_2$ should have the same bond structure. Usually they will have different *bond value sets* $B_v$ however.

It will sometimes be natural to construct the generator space $G$ from a smaller set $G_0$, the *initial generator space*, $G_0 \subset G$. There will then be a group BSG, usually distinct from the similarity group $S$, and this *bond structure group BSG* will extend $G_0$ to $G$ so that any $g \in G$ is obtained (perhaps in more than one way) from some $g_0 \in G_0, g = \mu g_0$, $\mu \in BSG$.

Here Grenander introduces relations between generators so that he can describe the bond structure, which he distinguishes from tags with values, Tags values signify something to someone beyond the bond and generator network. Notice that the idea of the generator assumes that it operates though time, so the generator itself represents flux, while the external relations between generators via bonding equals the discontinuities in space that the bonds traverse. Then the tag values represent signs to someone observing the configuration of the generators. Thus we can see in the definition the four kinds of pattern: structure, flux, value and sign. The bonds of the generators represent the Peirce-Fuller category of Secondness.

**4.2. Configurations.** We can now glue generators together and the bonds will tell us what combinations will hold together. This is a bit like chemistry: atoms (generators) are connected together into molecules (configurations) and the nature of the chemical bonds, ionic, covalent, and so on, decides what combinations of atoms will be stable enough to form molecules.

Say that we try to form a configuration from the generators $g_1, g_2, \ldots g_i \ldots g_n$ where the subscript $i$ will be referred to as a *generator coordinate*. For $g_i$ let the bonds be denoted $b_{i1}, b_{i2}, \ldots b_{ij} \ldots b_{i\omega}, \omega = \omega(g_i)$; $j$ is called the *bond coordinates* for $g_i$. Figure 3 illustrates this for $n = 2$, where the bond $(i_1, j_1)$, with the value $\beta_1$, tries to connect with $(i_2, j_2)$, with the value $\beta_2$.

Molecules of generators are built up through the bonding process. This is the extension of Secondness. But how generators themselves come into existence and go out of existence is not covered. Emergence is the appearances of firsts not just the chemistry-like combination of generator atoms into configuration molecules to get new combinations. However this idea of exploring adjacent possible molecular reaction spaces where new combinations become possible is what S. Kauffman suggests in his Investigations[11]. He says that there is a potential energy toward expansion of the actual reaction space into the adjacently possible. Yet the adjacently possible is predicated on reactions between existing molecules composed of existing atoms. However, what happens when new generator atoms come into existence, as happens in supernova with regard to real atoms. Turns out big atoms of physical substance are unstable. But here we are talking about generators of patterns. Those generators are idealization, not material objects. The production of new generators are not restricted in the way the production of atoms of mater are restricted physically. So emergence of new generators is always a possibility, which expands the adjacently possible, but makes the discontinuity between actuality and the adjacently possible more a stark discontinuity of the nature of an emergence.

84 *Chapter 4: Analyzing Patterns*

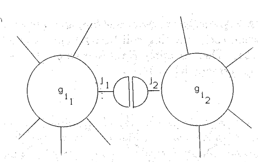
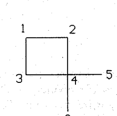



FIGURE 3. INTERACTING GENERATORS

Introduce a *configuration architecture* by selecting a graph, for example the one in Figure 4a where the sites are enumerated by $i = 1, 2, \ldots n$; $n = 6$. At site $i$ we position a generator $g_i \in G$ and then connect the bonds, for example as in the diagram Figure 4b. The resulting structure is called a *configuration*.



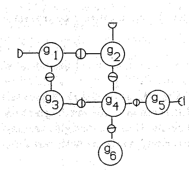




FIGURE 4A. CONNECTOR GRAPH FIGURE 4B. CONFIGURATION DIAGRAM

[11] pp 142-157 (Oxford 2000)

We use a graph $\sigma$, called a *connector*, that connects some bonds $k_1 = (i_1, j_1)$ with others $k_2 = (i_2, j_2)$, where $k = (i, j)$ labels bonds in general. We shall symbolically represent the configuration as

$$c = \sigma(g_1, g_2, \cdots g_n).$$

Some bonds in $c$ connect to other bonds; they will be called *internal bonds*. The remaining, closed bonds are the *external* ones, the set of them denoted ext$(c)$. In the figure $|ext(c)| = 3$.

We shall often restrict the configuration by local as well as global constraints. On the product space $\mathcal{B} \times B$ of the bond value space $B$ crossed with itself we assume, given a truth valued function

$$\rho : B \times B \to \{\text{TRUE,FALSE}\},$$

Here Grenander adds a Cartesian product between the bonds and assumes that each point in the table can be either true or false based on whether the bonds will connect. In this way he produces configurations of Bonds, i.e. a second order connection. We can imagine third and fourth, etc order connections as well between the bonds.

*Chapter 4: Analyzing Patterns* 85

so that for a pair $(\beta', \beta'')$ of bond values the pair is either regular, if $\rho(\beta', \beta'') = \text{TRUE}$ or irregular, if $\rho(\beta', \beta'') = \text{FALSE}$. This partitions $B \times B$ into two subsets and $\rho$ therefore is equivalent to a relation, to be called the *bond value relation*. It is not completely arbitrary, we shall assume that it is $S$-invariant in the sense that if for two internal bonds $(i', j')$ of $g_{i'}$ and $(i'', j'')$ of $g_{i''}$ we have $\rho[\beta_{j'}(g_{i'}), \beta_{j''}(g_{i''})] = \text{TRUE}$ then $\rho[\beta_{j'}(sg_{i'}), \beta_{j''}(sg_{i''})] = \text{TRUE}$ for all similarities $s \in S$.

At site $i$ the generator $g_i$ sends out bond values $\beta_1(g_i), \beta_2(g_i) \dots \beta_\omega(g_i)$. To some or all of these bonds $b_j$ there is a bond value, call it $\beta'_j$, that comes from the neighboring sites of $i$ in $\sigma$ and connects to $b_j$. We shall sometimes use the concept *environment* for the set

$$env(g_i) = (\beta'_1, \beta'_2, \dots \beta'_\omega).$$

Note that the bonds can sense each other and connect to each other if they can. This propagation of bond values can be seen as operating somewhat like a cellular automata. Adjacent bond values are sensed and that leads to the correct bond values connecting. This produces stable patterns among the generators.

We now introduce the crucial

**Definition.** A configuration $c = \sigma(g_1, g_2, \dots g_n)$ is said to be *locally regular* if for any internal bond couple $(i', j') - (i'', j'')$ we have

$$\rho[\beta_{j'}(g_{i'}), \beta_{j''}(g_{i''})] = \text{TRUE}.$$

Local regularity can be formalized in a way that will serve as a guide when we shall relax the regularity in section 4.5. Writing $k = (i, j)$, $k' = (i', j')$ to enumerate bonds in $c$ let us introduce the *first structure formula* as

$$\bigwedge_{<k,k'>} \rho[\beta_j(g_i), \beta_{j'}(g_{i'})] = \text{TRUE}.$$

In this logical conjunction (note 3.11) the notation $< k, k' >$ means that the conjunction is taken over all closed (internal) bond couples $(k, k')$. The structure formula is just another way of writing Definition 1, but one that will help our intuition later.

We shall also assume that $\sigma \in \Sigma$, some family of graphs called the *connection type*. Among connection types to be used later we mention $\Sigma$ = LINEAR (all linear chain graphs), $\Sigma$ = TREE (tree shaped graphs), $\Sigma$ = LATTICE (graphs with square lattice structure), but many others will also appear.

Non-regular rhizomatic graphs are also possible.

**Definition.** A configuration $c = \sigma(g_1, g_2, \dots g_n)$ is called *globally regular* if $\sigma \in \Sigma$.

If $c$ is both locally and globally regular it is said to be *regular* and the set of all such configurations, the *configuration space* $\mathcal{C}(\mathcal{R})$, where $\mathcal{R} =< G, S, \rho, \Sigma >$ is referred to as a *regularity*.

In Figure 5 consider the subconfiguration $c'$ inside the dotted line; is it regular if the whole configuration $c \in \mathcal{C}(\mathcal{R})$? Since all its internal bond couples remain regular (satisfy

86 *Chapter 4: Analyzing Patterns*

the bond relation $\rho$) it is certainly locally regular. Whether it is globally regular depends upon whether its connector $\sigma'$ belongs to the connection type $\Sigma$ as $\sigma$ does. An important case when this occurs is when the connection type is *monotonic*, which means that if some graph $\sigma \in \Sigma$ then any subgraph $\sigma'$ of $\sigma$ also belongs to $\Sigma$.

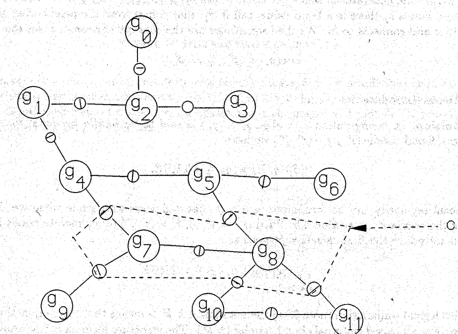



FIGURE 5. CONFIGURATION OF SUBCONFIGURATION

In Figure 5 we can write, with notation to be used in section 4.2,

$$\begin{cases} c & = \sigma(g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}) \\ c' & = \sigma'(g_8, g_9) \\ c'' & = \sigma''(g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_{10}, g_{11}) = \text{the rest of } c \text{ when } c' \text{ has been removed} \\ c''' & = \sigma'''(g_4, g_5, g_7, g_{10}, g_{11}) = \text{“the external boundary” of } c' \text{ in } c. \end{cases}$$

In other words we can combine regular configuration, for example $c'$ and $c''$, into larger configurations, say $c$, not always regular. It is to emphasize this *combinatory* nature

*Chapter 4: Analyzing Patterns* 87

of pattern theory that we have selected the notation $c = \sigma(g_1, \dots g_n)$ which should be interpreted as meaning that $c$ is a function of $g_1, \dots g_n$, the variables, and just as in function composition a $g_i$ can be replaced by a configuration $c_i$ if ext$(c_i) = B(g_i)$.

In Figure 6 we illustrate such a combination of three configurations $c', c'', c'''$ into a fourth one $c$. Think of the *coupling connector* $\sigma_0(c', c'', c''')$ as a function of three variables $c', c'', c'''$ and taking values in $\mathcal{C}(\mathcal{R})$

$$\sigma_0 : \mathcal{C}(\mathcal{R}) \times \mathcal{C}(\mathcal{R}) \times \mathcal{C}(\mathcal{R}) \to \mathcal{C}(\mathcal{R}).$$

Note, however, that the value in $\mathcal{C}(\mathcal{R})$ of this function is not defined for all $c', c'', c'''$: it may be that the three configurations do not fit, they cannot be glued together. Therefore $\sigma_0$ is only a *partial function*, not defined everywhere in $\mathcal{C}(\mathcal{R}) \times \mathcal{C}(\mathcal{R}) \times \mathcal{C}(\mathcal{R})$.

Another type of function, mapping $\mathcal{C}(\mathcal{R})$ onto itself, is given by the similarities applied to $c : c \to sc$; one such function for each $s \in S$. These functions are entire, they are defined everywhere. Compare this to the classical algebraic structures, for example the real line $\mathbb{R}$ with the operations $+$ and $\times$. For $x, y \in \mathbb{R}$ the operations $x + y$ and $x \times y$ are always defined. We can describe these functions by two tables: one for addition and another for multiplication. The function $\div$ on the other hand is partial, since $x \div y$ is not defined for $y = 0$.

A coupling connector, say with two variables $\sigma(\cdot, \cdot)$, can be described by a *composition table* with two margins, whose values may include $u$ = undefined. Such a table tells us what regular configuration $c = \sigma(c_1, c_2)$ is obtained by coupling $c_1$ with $c_2$ via the connector $\sigma$, if it is defined (regular) in $\mathcal{C}(\mathcal{R})$. Similarly for coupling connectors with three variables $\sigma(\cdot, \cdot, \cdot)$, and so on, and one table for each similarity in $S$. The latter table tells us what configuration $c'$ we get by applying a similarity to $c = \sigma(g_1, g_2, \dots g_n)$, $c' = \sigma(sg_1, sg_2, \dots sg_n)$; it is automatically defined (regular) since the bond relation $\rho$ is $S$-invariant and the connector $\sigma \in \Sigma$ is unchanged when we apply $s$.

88 *Chapter 4: Analyzing Patterns*

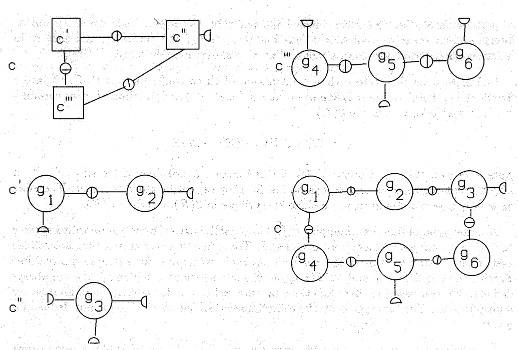


FIGURE 6. COMBINATIONS OF CONFIGURATIONS

Let us illustrate the concepts introduced in this section by our previous

**Example.** Let $\rho$ be defined by the $6 \times 6$ truth valued matrix of bond values $\beta, \beta'$ where 1 stands for TRUE and 0 for FALSE. For example, $\beta = 2, \beta' = 3$ fit but $\beta = 0, \beta' = 1$ do not. Introduce the connection type SQUARE LATTICE with all graphs as subgraphs of $\mathbb{Z}^2$, the set of points $(x, y)$ in the plane with $x$ and $y$ integers, and where a site $(x, y)$ is connected to four neighbors $(x+1, y), (x, y+1), (x-1, y), (x, y-1)$. Let $S$ be the translation group $\mathbb{Z}_2$ of translations $(x, y) \to (x+h, y+k)$ with $h$ and $k$ arbitrary integers (note 4.2).

*Chapter 4: Analyzing Patterns* 89

| | | $\beta$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 |
| | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| $\beta'$ | 2 | 0 | 0 | 0 | 1 | 0 | 0 |
| | 3 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 4 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 0 | 1 | 0 | 0 | 0 | 0 |

A regular configuration can then be shown by a diagram consisting of certain sites of $\sigma \in$ SQUARE LATTICE carrying $g$-values, here from $G = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$, and bond couples from $B = \{0, 1, 2, 3, 4, 5\}$, each couple satisfying $\rho$. The diagram could define a picture like the one in Figure 7 where the arrows indicate generators of the type $\mu g_i$, obtained from 2,3, or 4 in Figure 2 of section 4.1, and $\mu$ is an element $\mu \in BSG$ (the bond structure group); here BSG means $\mathbb{Z}_4$ consisting of rotations by a multiple of $90^o$.

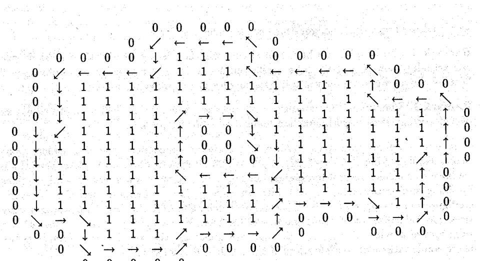


FIGURE 7. CONFIGURATION WITH BOUNDARY GENERATORS

If this configuration is denoted $c$ then $sc$ means the same configuration *shifted* a certain

90 *Chapter 4: Analyzing Patterns*

amount up or down and right or left. In addition to this shift-operation in $\mathcal{C}(\mathcal{R})$ we shall sometimes also use a *jittering* operation meaning the transformation

$$\sigma(g_1, g_2, \dots g_n) \to \sigma(s_1 g_1, s_2 g_2, \dots s_n g_n)$$

for some vector $(s_1, s_2, \dots s_n) \in S^n$. This important concept will also appear in section 6.3 as deformable templates. It must be pointed out, however, that the result on the right hand side is not always in $\mathcal{C}(\mathcal{R})$ since regularity may be lost when jittering a regular $c$. Hence we have only a partial operation: it is not defined for all $s$-values.

But we shall also have reason to consider mappings of configurations $c \in \mathcal{C}(\mathcal{R})$ to $c' \in \mathcal{C}(\mathcal{R}')$: a different configuration space from the first, but where the mapping preserves structure in some essential way. In analogy with algebra we shall introduce *configuration homomorphisms.*

Let us note that Bateson relates physical movement to the kinds of Being. Stasis is Pure Being, movement is Process Being, acceleration/deceleration is Hyper Being and jitter is Wild Being. We note here that patterns may be moved and transposed and display all these kinds of movement. Bateson talks about levels of learning as opposite the physus of movement. He discusses learning[1] as Pure Being, learning to learn (learning[2]) as Process Being, learning[3] as Hyper Being and learning[4] as Wild Being. Thus the types of learning that exemplify logos are opposite the types of movement that exemplify physus. Learning patterns, i.e. pattern recognition, is the dual of pattern generation.

**Definition.** Consider two regularities $\mathcal{R} =< G, S, \Sigma, \rho >$ and $\mathcal{R}' =< G', S, \Sigma, \rho' >$ over the same similarity group $S$ and connection type $\Sigma$. A mapping $h : \mathcal{C}(\mathcal{R}) \to \mathcal{C}(\mathcal{R}')$ is said to be a homomorphism if $h$ taking $c = \sigma(g_1, g_2, \dots g_n)$ into $\sigma(g'_1, g'_2, \dots)$ has the properties

$$\begin{cases} h[\sigma_0(c_1, c_2)] = \sigma_0(hc_1, hc_2) \\ h(sc) = sh(c) \end{cases}$$

where all the appearing configurations are assumed regular.

**Remark 1.** To simplify we have stated the definition less generally than needed: one can actually allow $\mathcal{R}$ and $\mathcal{R}'$ to have different $S$ and $\Sigma$. The role of homomorphisms will become clearer when we illustrate it by examples in the following chapters.

**Remark 2.** Sometimes it is necessary to restrict the $s$-values in the definition to a subgroup of $S$, for example for deformable templates used below (note 4.3).

In some configuration space $\mathcal{C}(\mathcal{R})$ with monotonic connection type $\Sigma$ consider a particular configuration $c^0 = \sigma(g_1^0, g_2^0, \dots g_n^0)$; it will be spoken of as a *template* (sometimes we have to consider several templates, but not here). Form a new space of configurations by fixing similarities $s_1, s_2, \dots s_n$ arbitrarily except that we demand that the "jittered" configuration $\sigma(s_1 g_1^0, s_2 g_2^0, \dots s_n g_n^0)$ be regular, belong to $\mathcal{C}(\mathcal{R})$, which is not automatically true. Consider the set of configurations of the form $c' = \sigma'(g_{i_1}^0, g_{i_2}^0, \dots g_{i_n}^0)$ where $\sigma'$ is a subgraph of $\sigma$ with sites $i_1, i_2, \dots i_n$. On this space $\mathcal{C}_0$, the set of *partial templates*, we allow combinations of configurations if the result still belongs to $\mathcal{C}_0$.

The partial template spaces play an important role in knowledge representations of variability. For any $c' \in \mathcal{C}_0$ we define $c'' = h(c') = \sigma'(s_{i_1} g_{i_1}^0, s_{i_2} g_{i_2}, \dots s_{i_m} g_{i_m}^0)$ which is automatically regular, $c'' \in \mathcal{C}_0$. This mapping $h$ satisfies the first condition in the definition of homomorphisms since if $c'_1, c'_2, c' = \sigma''(c'_1, c'_2)$ are all regular then $h[\sigma''(c'_1, c'_2)] =$

Grenander here talks about pattern templates. This is where we see patterns becoming templates of understanding or schemas.

*Chapter 4: Analyzing Patterns* 91

$\sigma''(hc'_1, hc'_2)$. The second condition demands that, for any $s \in S$, we have with the same notation

$$\begin{aligned} h(sc') = h[\sigma'(sg_{i_1}^0, sg_{i_2}^0, \dots sg_{i_m})] = \\ = \sigma'(s_{i_1} sg_{i_1}^0, s_{i_2} sg_{i_2}^0, \dots s_{i_m} sg_{i_m}^0) = \\ = sh(c') = s\sigma'(s_{i_1} g_{i_1}^0, s_{i_2} g_{i_2}^0, \dots s_{i_m} g_{i_m}^0) = \\ = \sigma'(ss_{i_1} g_{i_1}^0, ss_{i_2}, \dots ss_{i_m} g_{i_m}^0) \end{aligned}$$

but only if all $ss_i = s_i s$. This is the case if $S$ is commutative but can be interpreted under more general conditions (see note 4.3). We shall speak of such homomorphisms as producing *deformed templates.* Sometimes we shall let the $s_i$ be randomly chosen and we then refer to *probabilistically deformed templates*; many examples of this will appear later and make the concept more intuitive.

Now we see the images were Grenander produces the representations based on the configurations of generators. The image space is an exemplification of a Peirce-Fuller third, because it is based on continuity. Notice that in the image the discontinuities between the generators vanishes and we get continuous representations of patterns all be it perhaps deformed. It is the images that are observed by the one who understands via the pattern template or schema and also the one who based on their pattern recognition projects significance on the seen image of a pattern.

**4.3. Images.** The configuration space is the first of the *regular structures* that we are building. The next one consists of *images*, a concept that formalizes the idea of *observables*. In other words, a configuration is a mathematical abstraction which typically cannot be observed directly, but the image can. An ideal observer, with perfect instrumentation so that the sensor used has no observational errors, will be able to see some object, called image, that may carry less information than the configuration that is being observed. The loss of information is not caused by noise in the sensors (that will be treated in section 4.6) but is more fundamental and it takes some care to formalize the concept of image in order to get a suitable algebraic structure.

To do this we introduce an equivalence relation (note 4.4) $R$ between elements in $\mathcal{C}(\mathcal{R})$, so that for $c_1, c_2 \in \mathcal{C}(\mathcal{R})$ they are either equivalent, $c_1 \equiv c_2(modR)$ or they belong to different equivalence classes in the partition $\mathcal{I} = \mathcal{C}(\mathcal{R})/R$. We ask that $R$ has the following properties that will later on be seen to be needed. $R$ will be called an identification rule. We shall write $I = [c]_R$ for the image that contains $c$.

**Definition.** An equivalence relation $R$ in $\mathcal{C}(\mathcal{R})$ is said to be an identification rule

$$\begin{cases} (i) & \text{if } c \equiv c'(modR) \text{ then } ext(c) = ext(c') \\ (ii) & \text{if } c \equiv c'(modR) \text{ then } sc \equiv sc'(modR),\ \forall s \in S \\ (iii) & \text{if } c_1, c_2, c_1', c_2' \text{ as well as } c = \sigma(c_1, c_2), c' = \sigma(c_1', c_2') \end{cases}$$

are all regular and if $c_i \equiv c_i'(modR)$; $i = 1, 2$; then $c \equiv c'(modR)$.

The equivalence classes in $\mathcal{C}(\mathcal{R})$ are then called *images* and the set of images in the quotient space

$$\mathcal{I} = \mathcal{C}(\mathcal{R})/R$$

is called an *image algebra*. The images can be treated in a combinatory way as shown by

92 *Chapter 4: Analyzing Patterns*

**Theorem.** *The similarity transformations can be uniquely extended to $\mathcal{I}$ and connectors $\sigma$ to part of $\mathcal{I} \times \mathcal{I}$ in such a way that*

$$\begin{cases} (i) & s_1(s_2 I) = (s_1 s_2) I;\ I \in \mathcal{I} \\ (ii) & s\sigma(I_1, I_2) = \sigma(sI_1, sI_2) \ if\ \sigma(I_1, I_2) \in \mathcal{I}. \end{cases}$$

**Proof:** The natural extension of $s : \mathcal{C}(\mathcal{R}) \to \mathcal{C}(\mathcal{R})$ to $\mathcal{I}$ is to define $sI = [sc]_R$ where $c \in I$. To see that this definition is unique let $c'$ be another configuration in $I \subset \mathcal{C}(\mathcal{R})$. But condition (ii) in the definition tells us that $[sc']_R = [sc]_R$ and this proves uniqueness.

Similarly for given $I_1$ and $I_2$ in the image algebra $\mathcal{I}$ consider two regular configurations $c_1 \in I$ and $c_2 \in I$. If $\sigma(c_1, c_2)$ is meaningful as a regular configuration define $\sigma(I_1, I_2) = [\sigma(c_1, c_2)]_R$. Uniqueness of this definition follows from condition (iii) in the definition.

To see that (i) holds in the statement of the theorem we observe that, if $c \in I$,

$$s_1(s_2 I) = s_1(s_2[c]_R) = s_1[s_2 c]_R = [s_1 s_2 c] = (s_1 s_2) I$$

using condition (ii) in the definition. Also, to prove statement (ii), if $c_1 \in I_1$ and $c_2 \in I_2$,

$$\begin{aligned} s\sigma(I_1, I_2) = s[\sigma(c_1, c_2)]_R = [s\sigma(c_1, c_2)]_R = [\sigma(sc_1, sc_2)]_R \\ = s\sigma(I_1, I_2). \end{aligned}$$

Q.E.D.

Hence we can deal with the $I$'s as an element of $\mathcal{I}$ and form composition tables in $\mathcal{I}$ as earlier in $\mathcal{C}(\mathcal{R})$, so that $\mathcal{I}$ forms an algebra. At the moment we shall not study its properties any further but go back to our example to illustrate the idea of images.

Now we see where Grenander introduces the idea of the pattern. Pattern means a repetition based on similarity, which is a play of identity and difference. We note that identity and difference is an aspect of Being. We see presence in the concept of images which are different from the configuration space which is hidden from the observer so that presence and absence also plays a profound role in the definition of the pattern schema. In terms of Truth and Falsehood we saw that appear in the bonding arrangements. Truth and Falsehood are used as markers for the ability of bonds to produce connection types. Reality and Illusion appear in the relation of the pattern generator complex to the actually observed patterns. There is pattern synthesis and analysis that are meant to be the two ways in which we connect pattern simulators to the world and derive the pattern generators from the world. Thus reality comes from a testing of the pattern simulators against the world of actual patterns. This is very similar to Klir's idea of producing generators that can create patterns of data exactly like those that appear in the real world. Complete understanding of a pattern is based on success at being able to simulate that patter completely.

**4.4. Patterns.** The word "pattern" in everyday parlance often means something typical that can be repeated, something characteristic capable of repetition into similar copies. The textile patterns in section 3.2, for example, clearly have this flavor. The operative term here is "similar," which we have given a precise meaning in terms of a similarity group $S$.

This leads us naturally to the following formalization of "pattern."

**Definition.** Given an image algebra $\mathcal{I}$ we shall understand by a pattern any $S$-invariant subset $\mathcal{P}$ of $\mathcal{I}$, and by a pattern family $\{\mathcal{P}_\gamma\}$ a partition of $\mathcal{I}$ into patterns.

In other words a subset $\mathcal{P} \subset \mathcal{I}$ is a pattern if for any $s \in S$ and $I \in \mathcal{P}$ we have also $sI \in \mathcal{P}$. We shall be particularly interested in *minimal patterns* that have no pattern as a proper subset. This concept leads to the simple

**Proposition.** The finest pattern partition $\{\mathcal{P}^p\}$ of $\mathcal{I}$ is uniquely determined and can be generated by templates $T^p \in \mathcal{P}^p$ such that $\mathcal{P}^p = ST^p, \forall p$.

**Proof:** Let us define a relation on $\mathcal{I}$ by saying that $I_1$ and $I_2$ are similar, formally written as $I_1 \equiv I_2(modS)$ if there exists an $s \in S$ such that $sI_1 = I_2$. It is obvious that this relation is reflexive, $I \equiv I$, symmetric, $I_1 \equiv I_2 \implies I_2 \equiv I_1$, and transitive, $I_1 \equiv I_2$ and $I_2 \equiv I_3 \implies I_1 \equiv I_3$. Hence it is an equivalence and induces a partition of $\mathcal{I}$ into equivalence classes.

From each such equivalence class select one representative, a *template*, and denote it $T^p$ with some suitable superscript $p$. This choice is, in general, quite arbitrary, but in particular situations there is often a natural choice of the representation $T^p$.

94 *Chapter 4: Analyzing Patterns*

Setting $\mathcal{P}^p = ST^p$ is clear that $\mathcal{P}^{p_1}$ and $\mathcal{P}^{p_2}$ are disjoint if $p_1 \neq p_2$. Indeed, if they had one element in common, say $I$, we would have $I = s_1 T^{p_1} = s_2 T^{p_2} \implies T^{p_1} = s_1^{-1} s_2 T^{p_2} \implies T^{p_1} \equiv T^{p_2}$ against our construction. It is also obvious that $\bigcup_p \mathcal{P}^p = \mathcal{I}$ and that each $\mathcal{P}^p$ is $S$-invariant so that $\{P^p\}$ is a partition pattern family.

But $\{P^p\}$ is also minimal, since otherwise some pattern, say $\mathcal{P}^p$, must have an $S$-invariant proper subset $P_0^p$. The latter is impossible since then, if $I_0 = sT^p \in P_0^p$,

$$\mathcal{P}_0^p = S\mathcal{P}_0^p = SI_0 = SsT^p = ST^p = \mathcal{P}^p.$$

This shows the existence of a minimal pattern partition.

Uniqueness is immediate, since if $\{\mathcal{P}^p\}$ is an arbitrary invariant pattern partition and if $I \in \mathcal{P}^p$ then $SI \subseteq S\mathcal{P}^P = \mathcal{P}^p$. But $SI$ cannot be a proper subset of $\mathcal{P}^p$ which implies uniqueness.

Q.E.D.

Grenander says that what we are most interested in are minimal patterns which contain no sub-patterns and that we need to distinguish the templates that produce these minimal patterns in each case. Of course, in actuality this may be very difficult and may need to be done in different ways in each case. But what is good about Grenander's definition is that it shows most of the features that we have been discussing. For instance, his definition steps through the Peirce-Fuller categories. It displays the four kinds of patterning. We can see how it relates to meta-levels of movement and learning. His mathematical definition aligns well with what we have been saying about the aspects of Being and the kinds of Being. The mathematical definition of the pattern schema is driven by the sorts of metaphysical considerations that we have been elaborating. Pattern's become the basic schema upon which all other schemas are built as we move up through there emergent levels of unfolding.