

Generation of Higher Order Emergent Structures

Nils A. Baas
Michael W. Olesen
Steen Rasmussen

SFI WORKING PAPER: 1996-08-057

SFI Working Papers contain accounts of scientific work of the author(s) and do not necessarily represent the views of the Santa Fe Institute. We accept papers intended for publication in peer-reviewed journals or proceedings volumes, but not papers that have already appeared in print. Except for papers by our external faculty, papers must be based on work done at SFI, inspired by an invited visit to or collaboration at SFI, or funded by an SFI grant.

©NOTICE: This working paper is included by permission of the contributing author(s) as a means to ensure timely distribution of the scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the author(s). It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may be reposted only with the explicit permission of the copyright holder.

www.santafe.edu



SANTA FE INSTITUTE

Generation of Higher Order Emergent Structures

Nils A. BAAS^a, Michael W. OLESEN^b, and Steen RASMUSSEN^{b,c}

^aDepartment of Mathematical Sciences
University of Trondheim, NTH
N-7034 Trondheim
NORWAY

^bTSA-DO/SA MS M997 and CNLS MS B258
Los Alamos National Laboratory
Los Alamos, NM 87545
U.S.A.

^cSanta Fe Institute
1399 Hyde Park Road
Santa Fe, NM 87501
U.S.A.

August 9, 1996

1 Introduction

In the study of complex systems - in particular self-organizing systems - the notions of emergence and higher order structures come up. A framework for studying them was given in [1, 2] where the notion of a hyperstructure was introduced. In biological systems higher order hyperstructures occur both in an intuitive and a formal sense. But in formalizations of these systems it has turned out to be quite difficult to produce higher order emergence from first principles. The first problem is of course to agree on what a higher order structure is. It is known that second order structures occur relatively easy, the problem is to know how to proceed to third order without external interference. The goal of this paper is to discuss these problems in the light of the discrete field automata formulated in [19] and [12]. Then we shall try to extract some of the formal principles involved in the generation of higher order (hyper-) structures and relate it to dynamical systems.

2 Biological examples

In biology we know that hierarchies and higher order structures occur in many ways. Clearly we have the coarse fundamental order hierarchy

organisms < organs < tissues < cells < organelles < molecules

which have many refinements and substructures [8]. It seems like evolution has favored systems that have been built up by subunits at several levels. In that respect even the simplest cells are extremely complicated. In order to understand the basic principles leading to such structures we should start looking at the molecular level where we find many examples of supra-molecular structures built up by subunits at several levels and hence qualify for being intuitively called higher order structures. Some guiding examples:

- (i) The formation of polymers from monomers as for example ethylene molecules forming polyethylene.
- (ii) The monomeric protein actin, spontaneously associates into linear, helical polymers in the presence of ATP. These polymers are called actin filaments. The filaments are cross-linked by other proteins—fodrin or filamin to side-by-side aggregates like bundles or meshworks. These certainly qualify to be called natural 3rd order structures. They greatly increase the viscosity of the medium in which the filaments are suspended [8].
- (iii) Microtubules spontaneously form their monomeric subunits. The process starts with the tubulin α - and β -subunits forming α and β -dimers which then form linear polymers associating side by side to form the hollow microtubules [7].
- (iv) Viruses are also built up through aggregation levels of subunits. This applies to the simple tobacco mosaic virus and the much more complicated bacteriophage T4 [5].
- (v) Lipids dispersed in water form microscopic aggregates. Lipid molecules cluster together with their hydrophobic moieties in contact with each other and their hydrophilic groups interacting with surrounding water. These clusters may be micelles, in other cases bilayers, liposomes or even more complicated forms [11].

Many other and much more complicated examples of higher order structures exist [22], but it seems reasonable to start with the simpler structures and try to model the generation of them first. In the following we shall study a formal example which is chosen because of its conceptual clarity and not because it replicates the known dynamics for any particular biomolecular system. Our

system have hydrophobic-like and hydrophilic-like monomers which can polymerize. These polymers in turn can aggregate to micelle-like structures. Thus, a system with three distinctive levels: monomers, polymers, and aggregates.

3 Higher Order Emergence

We have in examples described real systems with higher order structures. But in order to understand them better, their use and synthesis, it is important also to have formal systems in which such structures can be generated. Then one can do experiments with the formal systems. In general when higher order structures occur, new properties arise for each level — for example through aggregation. This means that in this context we will be looking for objects or aggregates with *new* properties. An obvious question is then what *new* means? This really brings us into the basic discussion of emergence and the notion of an observer. For a discussion of emergence and higher order structures we refer to [1] where a suitable framework is given in which these concepts can be discussed. How this connects to dynamics we refer to [20].

Let us just recall briefly a few basic notions. We consider families of objects or structures S_r^1 of *first order*

$$S_r^1 = S_q^1(f_{rs}, s_r, \tau_r), \quad r, s = 1, 2, \dots, n \quad (1)$$

where s_r is the state of the object, f_{rs} defines the object-object interactions, and τ_r is the local object time. In addition we need to define an update functional U which schedules the object updates (e.g. parallel, random, event driven), which together with the interaction rules – given by f_{rs} – defines the dynamics. Also the important notion of an observer O^1 needs to be introduced. With O^1 we can measure explicit system properties as for instance internal object states.

The system dynamics may now generate a new structure S^2 through the interactions

$$S^2 = R(S_r^1), \quad r = 1, 2, \dots, n \quad (2)$$

where R is the process that generates S^2 . This is what we call a *second order* structure which may be subjected to a possible new observer O^2 . Then we say that a property P is *emergent* iff

$$P \in O^2(S^2), \text{ and } P \notin O^2(S_r^1). \quad (3)$$

Clearly emergence depends on the observer in use which may be *internal* or *external*. Also note that the emergent properties may be *computable* or *non-computable*.

The above process can be iterated in a *cumulative*, not necessarily a *recursive*, way to form higher order emergent structures which we shall call *hyperstructures* of e.g. order N :

$$S^N = R(S_{r_{N-1}}^{N-1}, S_{r_{N-2}}^{N-2}, \dots). \quad (4)$$

It should be noted that the definition of an observation function is no more – or just as – arbitrary as the definition of the objects and their interactions.

For more details we refer to [1] and [20] where the concept of emergence and the relation between emergence and dynamics is discussed.

4 Discrete field automata

4.1 The basic discrete field automata concept

Discrete field automata are based on the assumption that all molecular interactions can be modeled by mediating particles [19, 12]. We model both matter and fields as “information particles” that propagate locally along the edges of a lattice and interact with one another at nodes, as in a Lattice Gas (LGA) [4]. Thus, the rules that generate the dynamics are the rules that propagate the information particles and define the local reactions depending on the current state of the site together with an update functional U that schedules the object updates. Unlike a standard LGA, the different Lattice Polymer (or Molecular) Automata, LPA (LMA), we study here use several different types of information particles, so the structure of a node is more complicated than the simple six bit register required for a minimal LGA. The models to be discussed here are all formulated on a 2-D hexagonal lattice, see figure 1.

Figure 1 here.

The transmission of the force particles between the molecules enables an update of each individual molecule using only local information. After the information particle transport steps, each lattice site can be updated independently. The force-communicating particles propagate locally, that is, between neighboring lattice sites. A variety of molecular interactions may be formulated by choosing the mediating particles properly. For instance a polymer must obey a connectivity constraint between its monomers and all molecules must obey an excluded volume constraint.

A lattice automaton cell (object) has several variables (registers) which serve to represent the current state of the current location (object). The total state space \mathbf{X} consists of a set of n sequences (internal states of the objects) ¹ of the type

$$(i, j, s_q) = (i, j, x_1, \dots, x_q) \quad (5)$$

where x_1, \dots, x_q are values of the variables associated with each lattice site - a data structure - and where (i, j) indicates the lattice site location.

The object-object interactions are defined by functions or lookup tables f_{qr} which have one or more of the data structure variables as arguments. Intuitively we may describe a data structure \mathcal{D} (the internal states of an object) as a box (see figure 2) with a finite number q , of sub-boxes.

¹To be more precise: The number of molecular objects is n , but since vacuum is used to carry (transfer) the emitted information particles the rest of the lattice sites are just empty sequences, sequences with $x_1, \dots, x_q = 0$. More about the information mechanics later.

Figure 2 here.

Our “physical-space”, where the molecular configurations form, is a lattice \mathcal{L} directly coordinatized by pairs $(i, j), i, j \in N$ (integers). The contents of sub-box k is an element of a set (or space) $X_k, k = 1, 2, \dots, q$. A data structure \mathcal{D} is then formally defined by

$$\begin{aligned}\mathcal{D} &\in N \times N \times X_1 \times X_2 \times \dots \times X_k \times \dots \times X_q \\ &= N^2 \times \prod_{k=1}^q X_k \\ &= \mathcal{S}.\end{aligned}\tag{6}$$

Each space X_k contains a distinguished element, $*$, meaning that the corresponding box is empty. It is of course in principle possible that $q \rightarrow \infty$. If $q = \infty$ by the product we mean that in every element only a finite number of components can differ from $*$.² Recall that if objects r is at lattice location (i, j) a projection of the data structure $\mathcal{D}_{(i,j)}$ corresponds to s_r in equation (1). The total state space \mathbf{X} can now be expressed as

$$\begin{aligned}\mathbf{X} &= \mathcal{S} \times \mathcal{S} \times \dots \times \mathcal{S} \\ &= \mathcal{S}^n.\end{aligned}\tag{9}$$

If we for simplicity assume that the update is time stepped (as in the lattice automata system we shall discuss), which means that all objects always have the same local (= global) time t , then we may define $X(t) \in \mathbf{X}$ as the current state of the system (1) at time t . That is

$$X(t) = (s_1(t), s_2(t), \dots, s_n(t))\tag{10}$$

The system dynamics is generated by applying an update functional U to the family of objects given in (1)

$$\{S_1(t+1), \dots, S_n(t+1)\} = U(\{S_1(t), \dots, S_n(t)\}),\tag{11}$$

²Note that in formalizing the data structures on our lattice \mathcal{L} , we see that \mathcal{S} , the space of states for each object, describes a data-field over \mathcal{L} , since we have the projection p

$$\begin{array}{c} \mathcal{S} \\ \downarrow p \\ \mathcal{L} \end{array}\tag{7}$$

A data-field is then specified by a section of this map meaning that

$$c : \mathcal{L} \rightarrow \mathcal{S}\tag{8}$$

such that $c \circ p = \text{identity}$. This just expresses that c assigns a data structure to each site and hence defines a configuration.

and the local object-object interactions are defined by the f_{rs} 's.

Note that in general no *explicit*, closed form function $F : \mathbf{X} \rightarrow \mathbf{X}$ exists that takes the current global state

$$X(t) = (s_1(t), \dots, s_n(t)) \in \mathbf{X} \quad (12)$$

and maps it into some other state in the state space \mathbf{X}

$$F(X(t)) = X(t+1). \quad (13)$$

Such a function is only *implicitly* given through (11). F is a *composition* of the local f_{rs} s and the scheduling (ordering) of these interactions defined by U . Obviously, the classical dynamical systems which can be explicitly written in the form (13) are special cases of the form given in (1) and (11). This is true, because a system that explicitly can be written in the form (13) can be viewed as a *single* object S_1 from (1) which is iterated by f_{11} . Since there is no scheduling with only a single object, the update functional U , becomes the identity.

The problem of the local dynamics is how a data structure $\mathcal{D}(t)$ at lattice site (i, j) changes with time t

$$\mathcal{D}(t) = (i, j, x_1, \dots, *, \dots, x_q). \quad (14)$$

We can now write

$$\mathcal{D}(t+1) = (i, j, y_1, \dots, *, \dots, y_q) \quad (15)$$

where

$$\begin{aligned} y_k &= Y_k[(i, j, x_1, \dots, x_q), \\ &\quad (1(i, j), x_1, \dots, x_q), \\ &\quad \dots, \\ &\quad (6(i, j), x_1, \dots, x_q)] \\ &= f_{k0}(i, j, x_1, \dots, x_q) \circ \\ &\quad f_{k1}(1(i, j), x_1, \dots, x_q) \circ \\ &\quad \dots \circ \\ &\quad f_{k6}(6(i, j), x_1, \dots, x_q). \end{aligned} \quad (16)$$

$$\quad (17)$$

This means that $f_{k0}(i, j, x_1, \dots, x_q)$ is the part of the interaction which depends on the data structure information at lattice site (i, j) and $f_{kl}(l(i, j), x_1, \dots, x_q)$ is the part of the local interaction function that depends on the data structure information at the neighboring lattice site *in direction* l , where $l = 1, \dots, 6$. Note that each part of the data structure update is made with only local information, e.g. $f_{k3}(3(i, j), x_1, \dots, x_q)$ is computed at the neighboring site to (i, j) , in direction 3, and the result (operatively written by “ \circ ”) is then *written* into the variable x_k at box k in the data structure at site (i, j) . What our detailed

updating rules do is exactly to compute the only *implicitly* given function Y_k . Note that Y_k depends on U , since it depends on which sequence neighboring sites are updated in which e.g. can be parallel or random. The dynamics will preserve the length q of the data structures, but that this is a convention, which is not necessary.

4.2 Lattice automata dynamics

The data structure we use in this particular application has dimension $q = 7 (+2)$ ($= 9$ if also counting the lattice coordinates) and the object-object interaction algorithm, f_{rs} , consists of 10 substeps (5 major steps) for each of two scheduling colors. Formally, the dynamical system is of the type as defined in equation (16).

The data structure space \mathcal{S} on the lattice is defined as follows:

- x_1 defines the scheduling color. There are two colors, which are random for free monomers and always alternating in a polymer to ensure that neighboring monomers in a polymer can be updated independently in each half of the update cycle.
- x_2 defines vacuum and molecules. 0 indicates vacuum and water in some fraction (a mean field approximation), 1 indicates a hydrophilic monomer (a monomer that likes water), 2 indicates a hydrophobic monomer (a monomer that dislikes water).
- x_3 defines the incoming excluded volume particles ("Repellons") which are being propagated from every monomer.
- x_4 defines the incoming force particles ("Forceons") which are being propagated from every monomer. The hydrophobic monomers propagate positive force particles and hydrophilic monomers propagate negative force particles.
- x_5 defines the current velocity which is being influenced by the random kicks of the only implicitly given solvent (water).
- x_6 defines the bond directions. A monomer can either be free, have one bond (at the end of a polymer) or have two bonds (inside of a polymer).
- x_7 defines the incoming binding force particles ("Gluons").

Each full update cycle consists of the following main steps for each scheduling color:

- (1) Propagation of information particles, both excluded volume particles ("Repellons") to neighborhood two, negative or positive force particles ("Forceons") depending on whether the molecule is a hydrophobic or a hydrophilic

molecule also to neighborhood two, random kicks of the monomers from the only implicitly given water molecules, and propagation of binding force particles (“Gluons”) for polymerization.

- (2) Creation of possibly new bonds.
- (3) Computation of the most proper move direction of each monomer.
- (4) Move all molecules that need to move.
- (5) Clear lattice for propagated information particles.
- (6) Repeat step (1) - (5) for the other scheduling color.

After all objects have been updated once the global update is complete.

Figure 3 here.

The information particle propagation is discussed in figure 3 and an example of a single update that defines the molecular dynamics is shown on a small 8×8 lattice with periodic boundary conditions in figure 4. A complete, explicit description of the dynamics is given in the appendix. A different cellular automata formulation of polymer dynamics can be found in [16].

Figure 4 here.

In the next section we discuss some of the constructive aspects of the dynamics. In section 7 we generalize the above formalization together with a discussion of the constructive dynamics and put it into a traditional dynamical system framework.

5 Third order emergence in lattice automata systems

5.1 Second and third order interactions

All interactions between the first order objects, the molecules, are by definition first order interactions. The first order interactions are the interactions that generate the second order structures. For example the “glue” that makes up a polymer is generated by the monomer-monomer interactions that make up the bonds. The details of these interactions are defined in the appendix and in [19, 12].

Second order interactions are, for instance, the interactions we find between polymers. These interactions are generated from a *composition* of first order interactions, since each pairwise interaction always occur between the first order objects. When a polymer communicates with another polymer the interaction is given by the monomer-monomer interactions; the interactions between the monomers from the two different polymers as well as the interaction between the monomers making up each of the polymers. The second order interactions are therefore only *implicitly* given.

The local use — or *interpretation* — of the different kinds of communicated information defines the *operational semantics* of the information [19]. The solvent particles are for instance interpreted differently by the monomers depending on the context the monomer currently are in. If the monomer is free, the solvent particle will induce a movement in its direction, but if the monomer is polymerized it becomes a bit more complicated. The effect will depend on whether such a movement would violate bond restrictions.

Figure 5 here.

Thus, the *meaning* of an interaction with a solvent molecule (a kick) depends on the context in which it occurs. In a certain way, we can interpret the data structure and the functions operating on it as an *internal* observer. In particular, this can be stated as follows: The interpretation of the information depends on which hyperstructural level the communicating objects belong to. In this system we have three levels; L_1 the level of the free monomers and the solvent, L_2 the level of the polymers, and L_3 the level of the polymer aggregates. In general, *new hyperstructural levels support new means of communication*, both within new levels and between old and new. Note that different levels in general need not be part of a strict hierarchy. This is why the *object complexity is bound to increase* as more hierarchical levels are to be generated - at least for the first

hierarchical levels.³

Interactions between first order structures and second order structures are of course also possible, but here the interactions are not symmetric. The first order to second order interaction is of second order whereas the second order to first order interaction is of first order because of the reasons given above. This interpretation of the information depending on the context (e.g. in the hyperstructure) of the interacting objects has profound consequences for how higher order (≥ 3) emergent structures can self-assemble in formal self-programmable or constructive dynamical systems.

5.2 Third order emergence

Let us emphasize that the following example is chosen because of its *conceptual clarity* and not because it replicates the known dynamics for any particular molecular system. For a physico chemical discussion of the lattice automata dynamics please see [12].

In figure 6 it is clear that genuine third order structures are generated in the lattice automata system through a generation of second order structures which interact and form third order structures. At the present state of the lattice automata system we only look for and observe polymer aggregates. As we know from the previous discussion each of these aggregates are generated through the interactions between first and second order objects as well as through the interaction between second order objects. Examples of second order properties the polymers carry are: elasticity and polarity. Examples of third order properties the aggregates carry are: permeability - and an inside and an outside. All of these properties do not have any meaning at the level(s) below. They are, however, generated or caused due to the lower level interactions. Also note that the higher order structures cause the lower level structures to behave in a different manner. The dynamics of the monomers are more restricted once they form a polymer and polymers are more restricted once they form an aggregate. Thus, there is a clear *downward causation* in such systems.

A formal identification of emergent properties of the second as well as the third order structures can of course also be defined in terms of external computational algorithms that inspect the data structures and how they communicate. These are examples of external, algorithmic observational functions of order two O^2 , and three O^3 . It does not matter whether the observational mechanism is an algorithm or a human.

³Whether more explicit internal object complexity always will be necessary to obtain yet higher order emergent structures - and whether there is an object complexity limit above which any functional property can be produced, we do not know. We would at least like to believe that the latter is true. The (limited) number of molecules involved in the biochemistry of life may also suggest that.

Figure 6 here.

A thorough discussion of the lattice automata dynamics including scaling of the radius of gyration of the polymers, hydrophobic cluster formation (phase separation), micelle stability as a function of polymer length, and membrane stability, will be done elsewhere.

6 Object complexity

It is clear from the observations we have made of the dynamics of the discrete field automata systems that their ability to produce emergent structures is highly dependent on the degree of detail - or fidelity - of the objects. As more and more interactions - and more and more different molecules - are taken into account, the more complex emergent structures the lattice automata systems are able to produce.

Only allowing a simple molecule-molecule interaction without an excluded volume, enables us to define Lattice Gasses which can generate a variety of macroscopic fluid dynamics phenomena (\mathcal{D}_1 in figure 7)⁴. Defining an excluded volume for the monomers allows the production of a little more detailed monomer-monomer interaction dynamics which is what the simplest LPA/LMA system has (\mathcal{D}_2 and \mathcal{D}_3 in figure 7). By the addition of binding and scheduling information to each of the data structures it becomes possible to generate polymer dynamics (\mathcal{D}_4 in figure 7). These are examples of second order emergent phenomena, as we have discussed earlier.

The polymers, however, can in a direct way be generated by the dynamics if we allow an interaction which is specific for a polymerization process. Thus, it becomes possible to produce second order structures from first order structures (\mathcal{D}_5 in figure 7). If no binding information is present in the data structure it becomes possible to generate a phase separation or produce aggregates of hydrophobic monomers surrounded by water and hydrophilic monomers (\mathcal{D}_6 in figure 7). These aggregates are also examples of second order structures.

If binding information is present and the initial configuration is a random configuration of polymers with hydrophilic heads and hydrophobic tails the formation of micelle-like aggregates becomes possible (\mathcal{D}_7 in figure 7). In such a situation the system has produced third order structures from the interactions of second order structures (the polymers). If bond information together with polymerization interactions as well as hydrophobic/hydrophilic molecules all are defined, it becomes possible to generate polymer aggregates from an initial condition of random monomers (\mathcal{D}_8 in figure 7). Intermediate configurations of the dynamics will then be dominated by the newly polymerized hydrophobic/hydrophilic polymers. Thus, it is possible to produce third order structures from first order structures.

Figure 7 here.

Another part of the local object complexity is given by the the complexity of the f_{rs} 's measured as the number of rules or functions together with their

⁴Note that x_2 and x_5 defined in section 5 has been merged into x_2 in this section for a more clear comparison.

arguments which define the local transformation of the variables in the data structures. The number and the nature of the variables in \mathcal{S} together with their transformation rules $f_{r,s}$ (and U) characterizes the computational complexity (in terms of time, space, and algorithmic complexity)[14, 6] of the dynamical system. So our observed relation between the object complexity and the generative power of the dynamics can also be stated in the form: The higher order structures we want to generate the more complex objects the system must have.

Another way of saying this, is that a more detailed description of the Physics⁵ is necessary to allow the formal system to produce higher order structures. Weaker effects also have to be taken into consideration if more complex biomolecular structures have to be explained.

The above may at a first glance seem contradictory to the findings of e.g. very simple cellular automata rules capable of generating arbitrary complex behavior – behavior equivalent to a Universal Turing Machine [9]. However, this is only an apparent contradiction, since these findings only show that there *exist* simple cellular automata rules (objects with low complexity), capable of generating (any) complex structures. It does not say that *every* simple rule is capable of generating arbitrary complex structures. Obviously, the dynamics of any of the above rather complex (molecular) object descriptions given in figure 7 could be reduced to one of these simple cellular automata rules (because any formal description can). But what would that give us? It would very likely not have any *explanatory* power in terms of what we know about the Physics, since the interpretation of these new, simple rules would (by definition, because they are simpler) be quite different from what they currently are. Also such a compression of the rules would probably have to be compensated by increase time and space complexities.

⁵We use Physics to denote principles of Nature independent of our description or knowledge of it. We use physics to denote our formal understanding and models of these principles.

7 A formal dynamical systems description

Recall that $\mathbf{X} = S^n$ is the state space and $X(t)$ the current state of the system. Since the current system state $X(t)$ at any time t can be obtained by an inspection of all the internal states of the objects $S_r(f_{rs}, s_r(t))$, via some appropriate observational functions O^I , a state dynamics function of the form (13) is always *implicitly* defined. Note, however, that an explicit expression of the global state space mapping in general will not be obtainable. What *explicitly* produces the dynamics, is the local application of the f_{rs} functions, defining the object-object interactions — together with an appropriate update functional U , which schedules these object-object interactions. Thus

$$\begin{aligned} \{S_1(t+1), \dots, S_n(t+1)\} &= U(\{S_1(t), \dots, S_n(t)\}) \\ &= U \circ \{f_{rs}\}(\{(s_r(t), s_s(t))\}), \end{aligned} \quad (18)$$

where $r, s = 1, \dots, n$. Thus, (18) becomes the *natural form* for the dynamics and not (13). By “ \circ ” we mean a composition which e.g. can have the form given in section 4.1. For a discussion of some of the mathematical consequences of this more general form of a dynamical system — which we have defined as a *simulation* — we refer to Rasmussen and Barrett [20].

In X the data structures may have different complexity as measured by the number of non-empty boxes and the number of states in each box. The complexity of the f_{rs} s may also differ measured as the number of rules or functions together with their arguments, that define the local transformation of the variables in the data structure.

The general dynamical system question is: If we start out with a random lattice configuration, how will the dynamics transform it – what is the transient and what is the asymptotic behavior? Our goal with this system is to look for the production of polymers (2nd order structures) and even more important; aggregates of polymers (3rd order structures) within one formal dynamical system (recall \mathcal{D}_7 in section 6). Therefore our observers are external mechanisms (algorithms or humans) looking for monomer-configurations, polymer-configurations, and aggregate-polymer-configurations. The interactions are all determined by the local object models, but are at the same time using the data structures as a kind of internal observers which makes the interactions context dependent.

In the simulation we observe that polymers and polymer aggregates are created. We now want to put this into a traditional dynamical system scheme. We suggest that the observed dynamics can be interpreted as follows: There exists a subspace $\mathcal{Y} \subset \mathcal{C}$ (= space of configurations) such that $\tilde{F}(\mathcal{Y}) \subset \mathcal{Y}$.

$$\mathcal{Y} = M_1 \cup M_2 \cup M_3 \cup P_1 \cup P_2 \cup A_1 \quad (19)$$

where M_1 , M_2 and M_3 are sets of monomer configurations of different complexity in a well defined sense; M_1 lowest complexity, then M_2 and then M_3 .

Similarly P_1 and P_2 are sets of polymer-configurations with P_2 more complex than P_1 . A_1 is a set of polymer-aggregate-configurations.

$\tilde{\mathcal{Y}} = A_1 \cup P_1 \cup M_1$ is invariant and the dynamics follows the following scheme - which can easily be generalized to arbitrarily number of levels:

$$\begin{array}{ccccc}
 M_3 & \cup & M_2 & \cup & M_1 \hookrightarrow \\
 \downarrow & & \downarrow & & \\
 P_2 & \cup & P_1 & \hookrightarrow & \\
 \downarrow & & & & \\
 A_1 & \hookrightarrow & & &
 \end{array} \tag{20}$$

This scheme should be interpreted as follows:

- (i) The low complexity monomers in M_1 , simply remain within their class under iteration of the dynamics. They do not have enough structure (object complexity) to evolve.
- (ii) The more complex monomers in the M_2 class form polymers of type P_1 , but under further iteration they just remain in the class and do not evolve further.
- (iii) The most complex monomers in M_3 , transform into P_2 -polymers, but only as an intermediate stage. The dynamics now give rise to new, context interpreted interactions which under further iteration form polymer-aggregates A_1 . This class remains stable under iteration. The evolution from monomers has stopped. The elements of A_1 represent the ultimate power of the system to evolve higher order complexity.

The elements of A_1 having been formed by the process:

$$M_3 \rightarrow P_2 \rightarrow A_1 \tag{21}$$

by interactions and observations and form a genuine hyperstructure of order 3 in the sense of Baas [1].

It would be interesting to look into general dynamical systems with this kind of structure — especially the scheme given by (20) — whether they would be natural generators of higher order structures. In general dynamical systems one could say that this scheme would correspond to:

M_1 : An invariant set.

P_1 : Attracting domain of M_2 .

P_2 : “Semi-attracting” domain of M_3 - to be mathematically defined.

A_1 : Attracting domain of P_2 and second attracting domain of M_3 .

8 Discussion

We have demonstrated how the lattice automata can be formulated so that they fit into a broader dynamical systems context. We have also seen how the dynamics of these mappings are defining higher order structures that can be generated in simulation. It follows that genuine third order structures can be generated using discrete field automata.

We have seen how increasing the object complexity of the primitives, the first order objects, is crucial for the discrete field automata to produce higher order emergence. Another way of saying this is that a more detailed description of the system is necessary to allow the system to produce higher order structures. Also weaker effects has to be taken into consideration if more complex structures have to be explained.

Since the very beginning of the study of Complex Systems the dogma described in figure 8 has been dominating.

Figure 8 here.

Is this really true? Can our “complex” hyperstructures be created only starting with very simple object states and interaction rules without any external interaction with the system? ⁶ We doubt it. We do not doubt, of course, that complex structures in formal systems *can* arise from very simple rule and state descriptions which are much simpler than the structures they generate. However, we question a stronger version of the dogma that essentially holds that a common minimal simplicity underlies all emergent structures. Furthermore, there are complexities that require levels of construction.

In the lattice automata described we have two levels of emergence leading to third order structures. We start with monomers and observe the interactions and first the polymers emerge with new properties relative to an external observational mechanism. The polymers then interact and the subtle point is that they actually interact with more than just 1st level monomer-monomer interactions - they also interact via 2nd order polymer-polymer interactions. When monomers from different polymers interact, part of the data structure acts as a kind of internal observer and relate the information for both polymers so that it really becomes a polymer-polymer interaction. The dynamics eventually lead to micelle-like polymer aggregates and as aggregates they qualify for 3rd order structures.

This approach can of course be questioned. By introducing more object complexity do we not then “script” the outcome of the dynamics? Yes, it is

⁶Note that we are considering fixed objects with no self-programming or “mutations” in the explicit rules and variables. Our objects do not learn, which seems reasonable, since they model simple invariant molecules.

scripted in the sense that we try only to include information relevant for the processes under investigation. But this is the case in any scientific inquiry. More importantly, our object models do not “cross” the levels and reference polymers or polymer aggregates. They all reference monomers or monomer properties. Thus, the interactions these rules together with the update functional define are non-trivial to the production of the higher order structures we are investigating.

The next step is to extend the formal system so that 4th and higher order structures can be generated from first principles. This should indeed be possible using the lattice automata following the principles we have outlined. We could for instance imagine the generation of what corresponds to ion channels in membranes, so that an active transport would be possible. The generation of “tissues” or sheets of vesicles could perhaps also account for yet another non-trivial level of structures.

We would also like to ask the question whether there are more direct types of second order (e.g. polymer-polymer) interactions. Interactions more directly given by observable polymer-polymer properties. This might correspond more to a system where a production of the protein filaments (recall the example in section 2) are joined by a third kind of molecule binding the two filaments together and thus acting as the main object mediating the 2nd order interactions. For the time being we leave these questions open, but we hope to extend our scheme to other examples - not necessarily of a biological nature ⁷

Finally it should be noted, that it is of course also possible to create systems similar to the ones presented here, where the data structures are not on the lattice and where they are able to change structurally - not only in contents - but also in length and in size of contents as well as the functions transforming the contents. Thereby the system objects themselves can undergo an evolution through yet another kind of a self-programming of the data structures and their interactions. The dynamics of such systems has been studied in [17, 3, 10, 18]. However, in the present framework the higher order structures and their generation become more apparent, we believe.

Acknowledgements

We thank Chris Barrett, Bernd Mayer, Kai Nagel, Maya Paczuski, Christian Reidys, and Jay Riordon for constructive comments and discussions either during the work that eventually led to this paper or in the presentation of the results. We are in particular grateful for a number of clarifying discussions of the central issues in this paper we have had with Chris Barrett. We also thank the Norwegian Research Council for financial support.

⁷For instance, to include the generation of dynamical hierarchies in socio-technical systems as we are currently studying in the TRANSIMS[15] project at Los Alamos National Laboratory.

Appendix

A Notation

- $D = \{1, \dots, d_{\max}\}$, the set of directions. In the hexagonal lattice $d_{\max} = 6$.
- $d_d : N \times N \rightarrow N \times N$, compute the position of the site neighboring site (i, j) in direction $d \in D$.
- $\varrho : D \rightarrow D$, computes the direction number to the right of its argument.
- $\lambda : D \rightarrow D$, computes the direction number to the left of its argument.
- $\omega : D \rightarrow D$, computes the direction number of the opposite direction to its argument.
- $\#_b : \mathcal{D} \rightarrow \{0, 1, 2\}$, computes the number of bond directions actually “glued” to a neighbor, that is, $\#_b(i, j) = \#(x(i, j)_{6,k} \neq 0)$.

B Update

B.1 Propagation of information particles

This step defines the propagation of information particles. This is done by *all* particles, it is thus *not* regarding scheduling color. The step looks like:

FOR all particles $x(i, j)$ AND FOR all directions d DO:

- (1) IF $x(i, j)_{6,1} \neq d$ AND $x(i, j)_{6,2} \neq d$ THEN $x(d_d(i, j))_{3,\omega(d)} = x(d_d(i, j))_{3,\omega(d)} + 1$.
 - (2) IF $x(i, j)_2 = 2$ THEN
 - $x(d_d(i, j))_{4,\omega(d)} = x(d_d(i, j))_{4,\omega(d)} - 2$.
 - $x(d_{\varrho(d)}(d_d(i, j)))_{4,\omega(\varrho(d))} = x(d_{\varrho(d)}(d_d(i, j)))_{4,\omega(\varrho(d))} - 1$.
 - $x(d_d(d_d(i, j)))_{4,\omega(d)} = x(d_d(d_d(i, j)))_{4,\omega(d)} - 1$.
 - $x(d_{\lambda(d)}(d_d(i, j)))_{4,\omega(\lambda(d))} = x(d_{\lambda(d)}(d_d(i, j)))_{4,\omega(\lambda(d))} - 1$.
 - (3) IF $p < \rho_{\text{water}}$ THEN $x(i, j)_{4,q} = x(i, j)_{4,q} + 6$.
 - (4) IF $x(i, j)_2 = 1$ AND $\#_b(x(i, j)) < 1$ THEN FOR all directions d DO:
 - $x(d_d(i, j))_{7,\omega(d)} = -1$.
- ELSE IF $x(i, j)_2 = 2$ AND $\#_b(x(i, j)) < 2$ THEN FOR all directions d DO:
- $x(d_d(i, j))_{7,\omega(d)} = \#_b(x(i, j)) + 1$.

B.2 Creation of new bonds - polymerization

DO $\mathcal{L}_{\text{row}} \times \mathcal{L}_{\text{col}}$ times:

Choose a site (i, j) randomly. IF $x(i, j)_2 \neq 0$ THEN:

$n_{\text{candidates}} = 0$.

IF $x(i, j)_2 = 1$ THEN

IF $\#_b(x(i, j)) < 1$ THEN FOR all directions d DO

IF $x(i, j)_{7,d} = 1$ THEN

$n_{\text{candidates}} = n_{\text{candidates}} + 1$.

$c(n_{\text{candidates}}) = d$.

ELSE IF $x(i, j)_2 = 2$ THEN

IF $\#_b(x(i, j)) = 0$ THEN FOR all directions d DO

IF $x(i, j)_{7,d} = -1$ OR $x(i, j)_{7,d} = 2$ THEN

$n_{\text{candidates}} = n_{\text{candidates}} + 1$.

$c(n_{\text{candidates}}) = d$.

ELSE IF $\#_b(x(i, j)) = 1$ THEN FOR all directions d DO

IF $x(i, j)_{7,d} = 1$ THEN

$n_{\text{candidates}} = n_{\text{candidates}} + 1$.

$c(n_{\text{candidates}}) = d$.

IF $n_{\text{candidates}} > 0$ THEN

$q = \text{random}(1, n_{\text{candidates}})$.

IF $x(i, j)_2 = 2$ AND $\#_b(x(i, j)) = 0$ THEN

$x(i, j)_1 = \neg x(d_q(i, j))$.

IF $x(d_q(i, j))_2 = 2$ AND $\#_b(x(d_q(i, j))) = 0$ THEN

$x(i, j)_1 = \neg x(d_q(i, j))$.

FOR all directions d DO $x(d_d(i, j))_{7,\omega(d)} = 0$.

IF $x(i, j)_2 = 1$ OR $x(d_q(i, j)) = 1$ THEN $b = 0$.

ELSE IF $\#_b(x(i, j)) = 1$ THEN

IF $x(i, j)_{6,1} = 0$ THEN $b = 0$.

ELSE $b = 1$.

ELSE

IF $x(d_q(i, j)) = 0$ THEN $b = 0$.

ELSE $b = 1$.

$x(i, j)_{6,b} = q$.

$x(d_q(i, j))_{6,b} = q$.

B.3 Computation of direction

This is only done for molecules with the right scheduling color.

FOR all particles $x(i, j)$ with the right scheduling color DO:

$v_1 = x(i, j)_{4,1} - x(i, j)_{4,4}$.

$v_2 = x(i, j)_{4,2} - x(i, j)_{4,5}$.

$v_3 = x(i, j)_{4,3} - x(i, j)_{4,6}$.

The preferred direction d is computed as the direction in which there is the biggest "push".

IF $x(i, j)_2 = 2$ THEN $d = \omega(d)$.
 IF $x(i, j)_5 > 0$ AND $(x(i, j)_{6,k} > 0$ AND $x(i, j)_5 \neq \varrho(x(i, j)_{6,k})$ AND
 $x(i, j)_5 \neq \lambda(x(i, j)_{6,k})$ ($k = 1, 2$)) THEN $x(i, j)_5 = 0$.

B.4 Move particles

Move particles with the right scheduling color appropriately and update bonds.

References

- [1] N.A.Baas, Emergence, hierarchies and hyperstructures, Artificial Life III, proceedings, ed. C.G. Langton, Addison-Wesley/Santa Fe Institute Studies in the Sciences of Complexity Vol XVII (1994) 515-537.
- [2] N.A.Baas, Hyperstructures as a tool in nanotechnology. Nanobiology **3**, 49-60, (1994).
- [3] W. Fontana and L.W. Buss, The arrival of the fittest: Toward a theory of biological organization, *Bull. Math. Biology*, **56**, 1-64 (1994).
- [4] U. Frisch, B. Hasslacher, and Y. Pomeau, Lattice Gas Automata for the Navier-Stokes Equation, *Physical Review Letters* **56**, 1986, 1722- 1728.
- [5] N.S. Goel and R.L. Thompson, Movable finite automata (MFA): A new tool for computer modeling of living systems. Artificial Life, proceedings, ed. C.G. Langton, Addison-Wesley/Santa Fe Institute Studies in the Sciences of Complexity Vol VI (1988) 317-340..
- [6] J. Hopcroft and J. Ullman, Introduction to automata theory, languages, and computation, Addison-Wesley (1979).
- [7] H. Hotani, R. Laholz-Beltra, B. Combs, S. Hameroff and S. Rasmussen, *Nanobiology* **1**, 67 (1992).
- [8] A.L. Lehninger, P.L. Nelson, and M.M. Cox, Principles of Biochemistry, Worth Publishers, N.Y., 1993.
- [9] K. Lindgren and M.G. Nordahl, Universal computation in simple one dimensional cellular automata, *Complex Systems* **4**(3), 299-318, (1990).
- [10] K. Lindgren and M.G. Nordahl, Artificial food webs. Artificial Life III, proceedings, ed. C.G. Langton, Addison-Wesley/Santa Fe Institute Studies in the Sciences of Complexity Vol XVII (1994) 515-537.
- [11] P.L. Luisi, P. Walde, and T. Oberholzer, Enzymatic RNA synthesis in self-replicating vesicles: An approach to the construction of a minimal cell. *Ber. Bunsenges. Phys. Chem.*, **98** (No. 3), (1994) 1160-1165.
- [12] B. Mayer, G. Koehler, and S. Rasmussen, Simulation and dynamics of entropy driven molecular self-assembly processes. To appear in *Phys Rew E*.
- [13] B. Mayer and S. Rasmussen, Lattice Molecular Automata (LMA): A physico-chemical simulation system for constructive molecular dynamics. LA-UR/SFI preprint 96-1732. Available from the authors.
- [14] M. Minsky, Computation - finite and infinite machines, Prentice-Hall, 1972.

- [15] K. Nagel, C.L. Barrett, and S. Rasmussen, Network traffic as a self-organized critical phenomena. LANL/SFI preprint 96-659. To appear in the Proceedings of The International Twin Conference on Self-Organization and Complexity, Stuttgart and Berlin, Germany, September 20-28, 1995, ed., F. Schweitzer, World Scientific Publ.
- [16] O. Ostrovsky, M. A. Smith, and Y. Bar-Yam, "Applications of Parallel Computing to Biological Problems", *Annu. Rev. Biophys. Biomol. Struct.*, **24**, 239-67 (1995).
- [17] S. Rasmussen, C. Knudsen, R. Feldberg, and M. Hindsholm, The core-world: Emergence and evolution of cooperative structures in a computational chemistry, *Physica D* **42**, 111-134, 1990.
- [18] S. Rasmussen, C. Knudsen, and R. Feldberg, Dynamics of programmable matter. Artificial Life II, proceedings, ed. C.G. Langton, C. Tayler, J.D. Farmer, and S. Rasmussen, Addison-Wesley/Santa Fe Institute Studies in the Sciences of Complexity Vol XVII (1991) 211-254.
- [19] S. Rasmussen and J.R. Smith, Lattice Polymer Automata, *Ber. Bunsenges. Phys. Chem.*, **98** (No. 3), (1994) 1185-1193.
- [20] S. Rasmussen and C. Barrett, Elements of a theory of simulation, Lecture Notes in Artificial Intelligence, Proceedings eds F. Moran et. al., Springer Verlag, Vol. 929 (1995) 515-529.
- [21] S. Rasmussen, N.A. Baas, C.L. Barrett, and M.W. Olesen, A note on simulation and dynamical hierarchies. LA-UR/SFI preprint 96-661. To appear in the Proceedings of The International Twin Conference on Self-Organization and Complexity, Stuttgart and Berlin, Germany, September 20-28, 1995, ed., F. Schweitzer, World Scientific Publ.
- [22] A.C. Scott, Stairway to the mind. Springer-Verlag (1995).

Figures

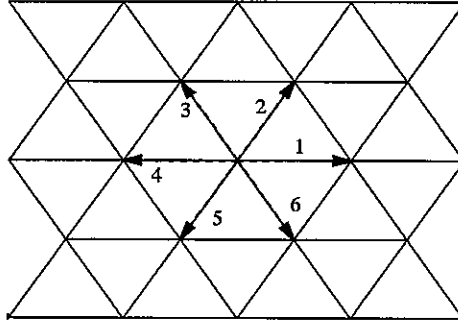


Figure 1: *Definition of the hexagonal lattice \mathcal{L} and the 6 principal lattice directions.*

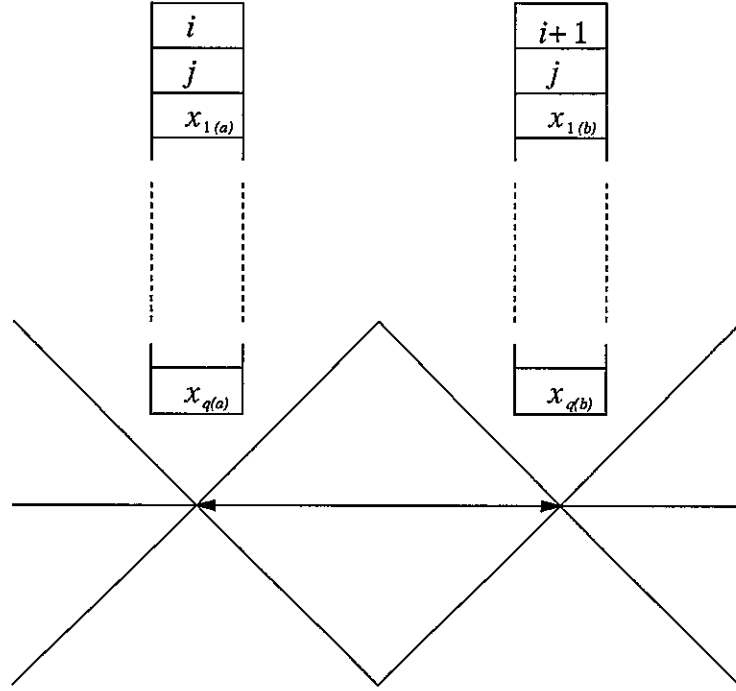


Figure 2: Two neighboring data structure $\mathcal{D}_{(i,j)}$ and $\mathcal{D}_{(i,j+1)}$ on the lattice \mathcal{L} .

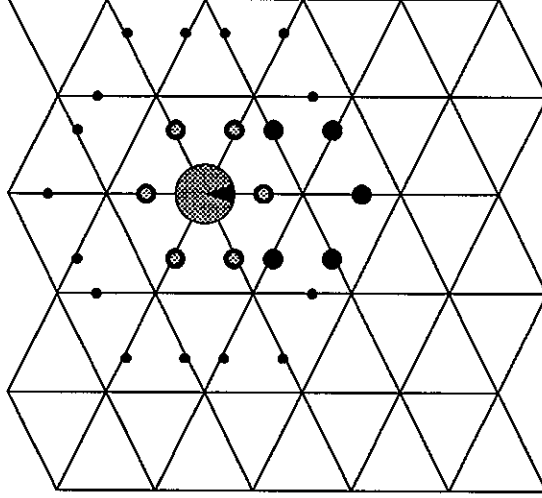


Figure 3: *Force particle propagation. A monomer is propagating excluded volume particles, “Repellons”, and force particles, “Forceons”. The Repellons (indicated by the large grey information particles) are propagated to distance one except in the direction where the monomer intend to move where they are propagated out to neighborhood two. The move direction is indicated by the black “sector”. The Forceons (indicated by the small information particles) are propagated to neighborhood two, but are also present at neighborhood one together with the Repellons (the Repellons cover them). The binding force particles (“Gluons”), which are not shown, are also propagated to neighborhood two. (The force particles are depicted near, and not directly at, the corresponding lattice site, because a molecular object at the same site then would “cover” the information particles.)*

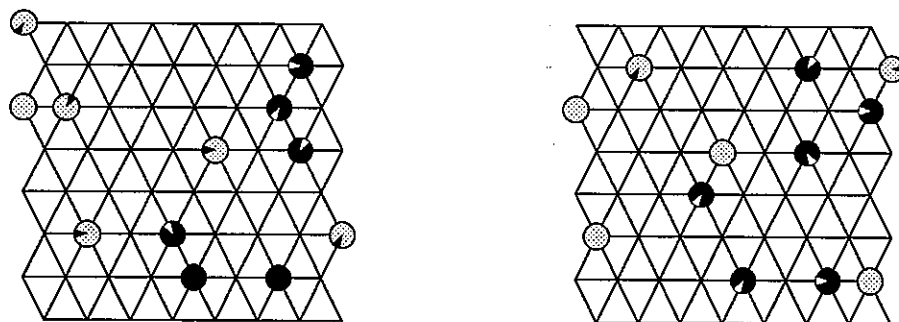


Figure 4: *A single update of free monomers, hydrophobic (grey) and hydrophilic (black) on a 8×8 lattice with periodic boundary conditions. Only one scheduling color is used here. Note how random kicks from the (non explicit) solvent changes the velocity of some of the monomers. Eventually the hydrophobic monomers will cluster.*

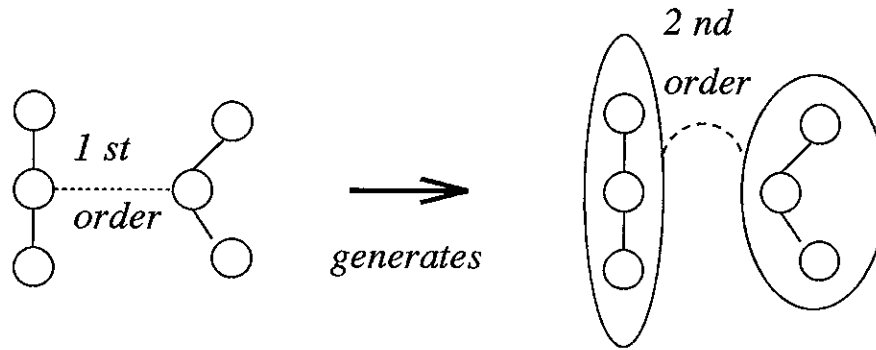
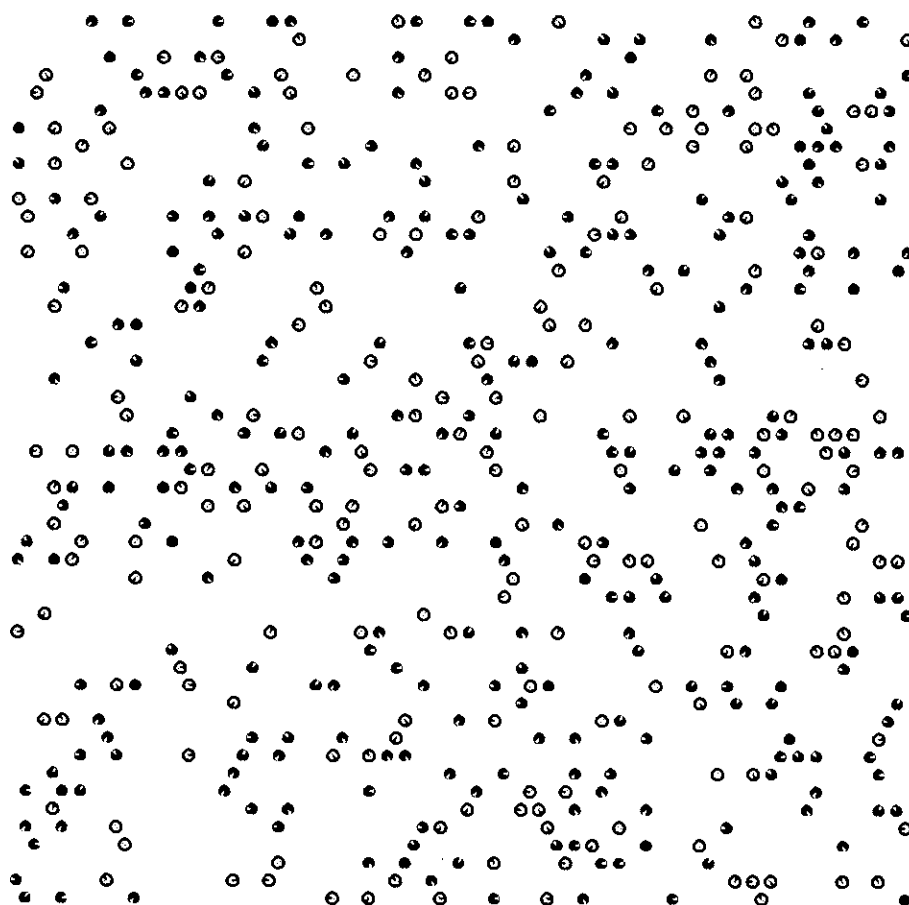
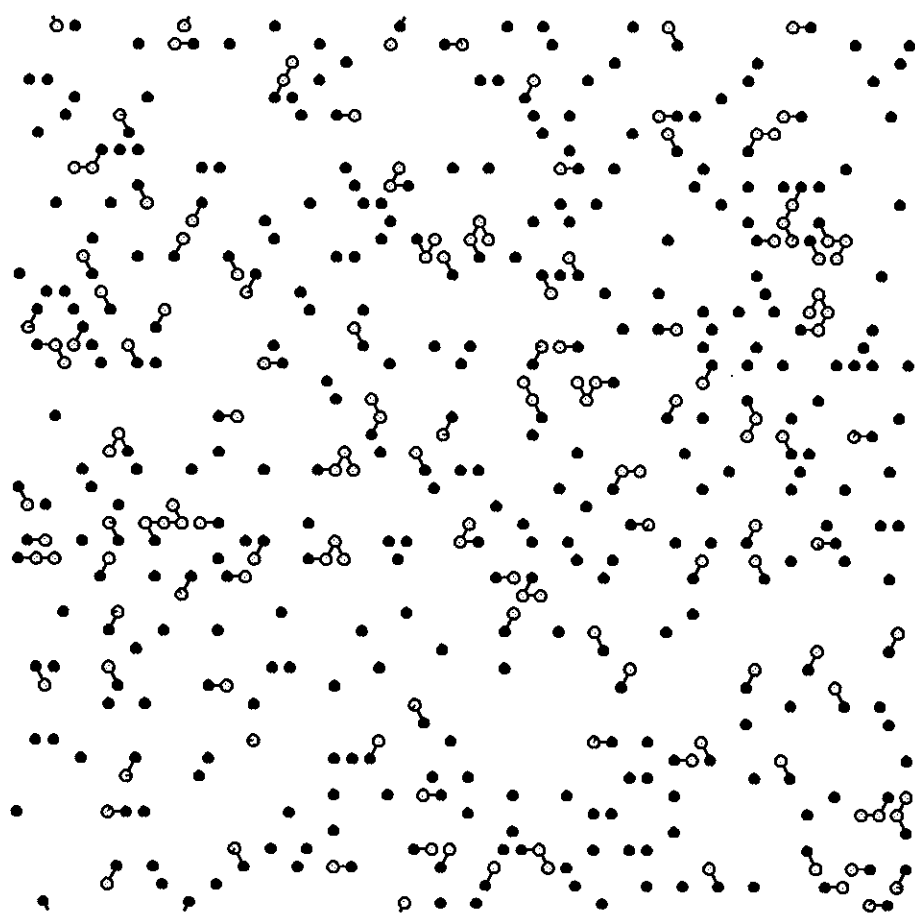
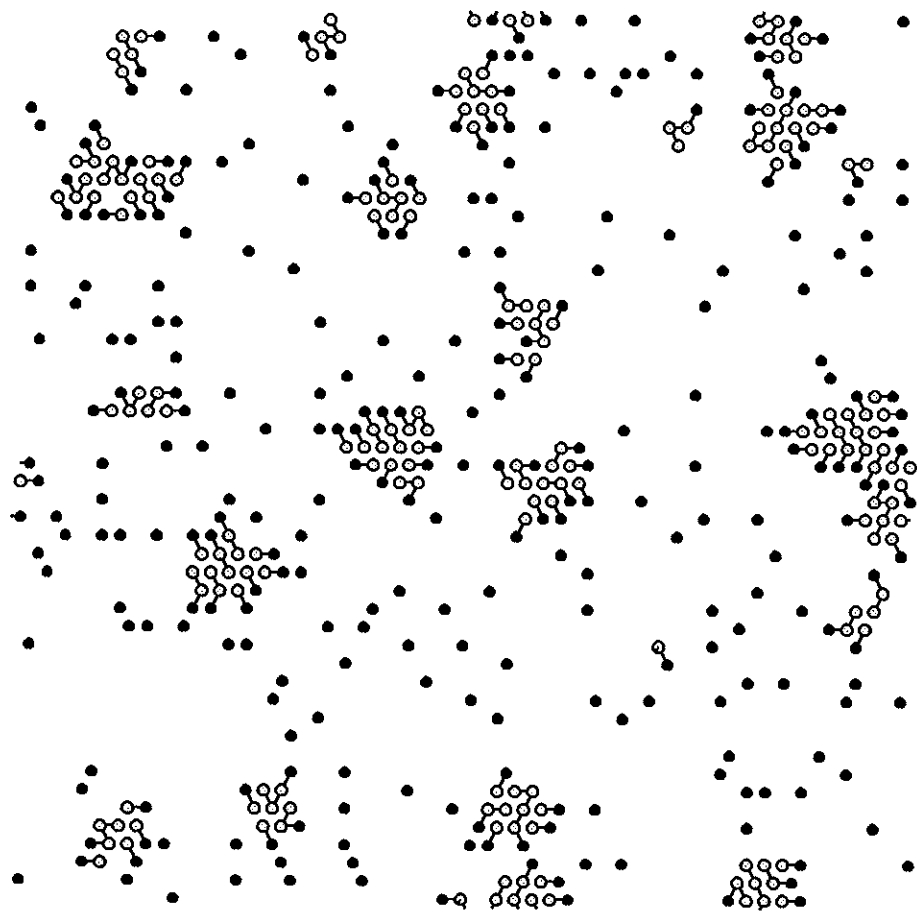


Figure 5: *Data structures acting as internal observers (as opposed to external observers) generating second order interactions. A 1st order interaction giving a communication channel between aggregates. The internal 1st order interaction within the aggregates communicate this interaction through the aggregate. A 2nd order interaction between the aggregates is therefore induced by the 1st order interactions - external as well as internal.*







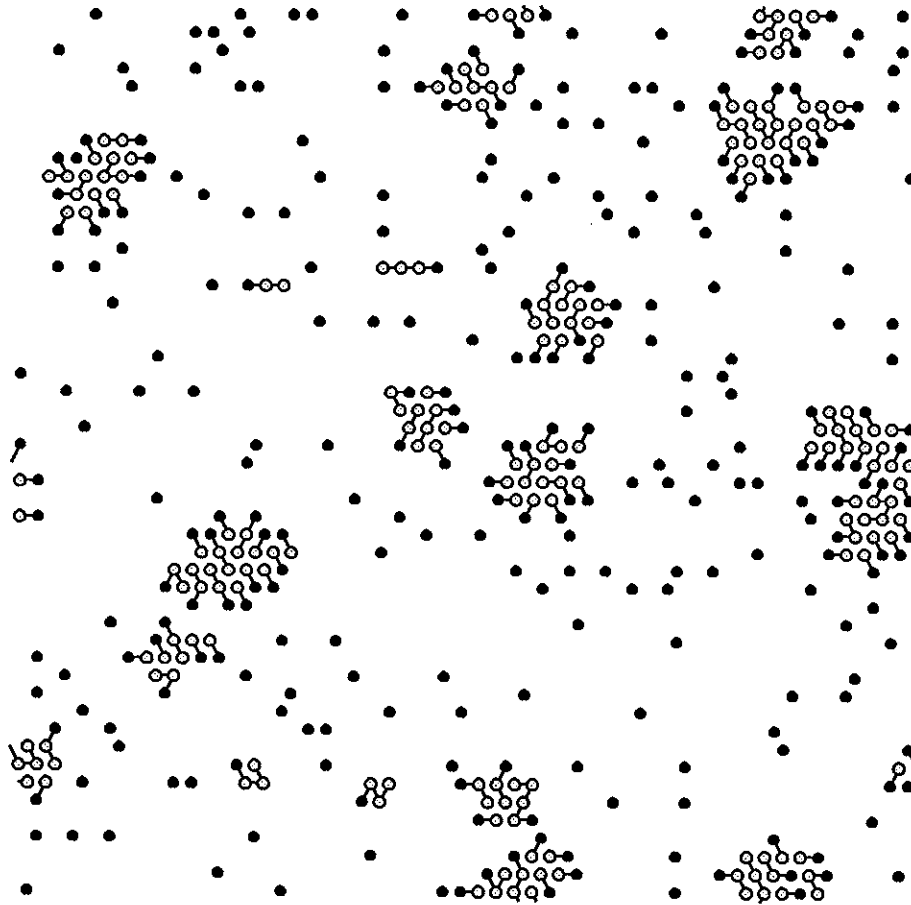


Figure 6: *Generation of third level structures in the lattice automata system. L_1 (monomers at time $t = 0$) \rightarrow L_2 (polymers at time $t = 30$) \rightarrow L_3 (micelle-like aggregates at time $t = 20,000$ and $60,000$). Initially there are approximately the same number of hydrophobic (black) and hydrophilic (white) monomers. The polymerization occurs such that a hydrophobic monomer (which is the nucleation center) can form a bond to a hydrophilic monomer which in turn can polymerize another hydrophobic monomer etc. At time $t = 30$ there are still a few free hydrophilic monomers not yet polymerized. Note that the formed micelle-like clusters are quite stable for these initial conditions and parameters, since they do not change much between $t = 20,000$ and $60,000$.*

----	----	----					scheduling color
*	*	*					vacuum and molecules
							excluded volume particles
*							hydrophobic/philic forces
*	*		*		*		bond directions
*	*	*					polymerization
*	_*_	_*_	_*_	_*_			

\mathcal{D}_1 \mathcal{D}_2 \mathcal{D}_3 \mathcal{D}_4 \mathcal{D}_5 \mathcal{D}_6 \mathcal{D}_7

Figure 7: Definition of object complexity as the number of active variables in the data structure. The by * marked variables are empty. \mathcal{D}_1 produces fluid dynamics as defined through the Lattice Gas. \mathcal{D}_2 produces monomer dynamics with excluded molecular volumes. \mathcal{D}_3 produces aggregates of hydrophobic monomers surrounded by water. \mathcal{D}_4 produces polymer dynamics. \mathcal{D}_5 produces polymer aggregates, vesicles and membranes starting from polymer interactions. \mathcal{D}_6 produces polymers from monomers through a polymerization process. \mathcal{D}_7 produces vesicles and membranes from polymers that are polymerized from the initial monomers.

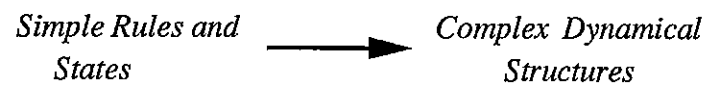


Figure 8: *Complex Systems Dogma.*