On Modeling of Self-organizing Systems*

Richard Holzer  
University of Passau  
Innstrasse 43  
94032 Passau  
Germany  
holzer@fim.uni-passau.de

Hermann de Meer  
University of Passau  
Innstrasse 43  
94032 Passau  
Germany  
demeer@fim.uni-passau.de

ABSTRACT
A goal of computing and networking systems is to limit administrative requirements for users and operators. A technical systems should be able to configure itself as much as possible to increase the usability. This leads to the design of self-organizing systems. Self-organizing systems emerge as an increasingly important area of research, not only for computer networks but also in many other fields. For analyzing properties of complex systems, a mathematical model for these systems may be useful. Whether a model with discrete time or with continuous time fits better, depends on the properties of the system and which analysis should be done in the model. In this paper we give a comparison between discrete and continuous models and we give a formal definition for modeling continuous complex systems. Then this theory is applied to model slot-synchronization in wireless networks.

Categories and Subject Descriptors  
I.6.5 [Model Development]: Modeling methodologies

General Terms  
Modeling Systems

Keywords  
Self-Organization, Mathematical modeling, Systems

1. INTRODUCTION
In computing and networking systems, self-organization becomes more and more important. Self-organization cannot only reduce administrative requirements and configuration work, but a self-organizing system should also be able to detect and correct failures automatically if possible. Another trend in networked systems is to distribute the network control and data among the entities of a network such that the system becomes more independent of centralized servers and control instances.

A non-technical overview of self-organizing systems can be found in [6]. Self-organization can be seen as the increase of coherence or as the decrease of statistical entropy. The main properties of self-organization are:

- Autonomy: Nearly no external control is needed for the system.
- Emergence: Local interactions induce the creation of globally coherent patterns.
- Adaptivity: Changes in the environment have only a small influence on the behavior of the system.
- Decentralization: The control of the system is not done by a single entity or by just a small group of entities, but by all entities of the system.

Other definitions and properties of self-organizing systems can be found in thermodynamics [10], synergetics [5], information theory [11] and cybernetics [13], [1], [2], [7]. A good overview about modeling complex systems can be found in [3]. For modeling discrete self-organizing systems see [8].

In this paper section 2 gives a comparison between continuous modeling and discrete modeling with respect to self-organization. Section 3 proposes a formal method for the modeling of continuous systems. In section 4 we apply the definitions of section 3 to model the algorithm of [12] for slot-synchronization in wireless networks. Section 5 concludes this paper.

2. DISCRETE VERSUS CONTINUOUS
When we design a model for a complex system, then we first have to check, which parts of the system should be modeled discrete, and which parts should be modeled continuous. There are four major parts, where this decision has a large impact on the design and on the behavior of the model. They are: time, object set, states of objects, interaction.

In a model with discrete time, we only consider a finite or countable number of steps in time, so a model with discrete time usually is used, when behavior is event driven, i.e. for the time between two events there is no need to model anything of the system, so only the (finite or countable) events appear in the model. A model with continuous time usually

*The presented work has been supported by the EU projects Euro-NF (NoE, FP7, ICT-2007-1-216306) and Autol (STREP, FP7, ICT-2007-1-216404).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Autonomics 2008 September 23 - 25, 2008, Turin, Italy  
Copyright 2008 ACM 978-963-9799-34-9 ...$5.00.
is used, when changes in the system do not occur in form of 
events, but every time.

In a model with a discrete object set, we only consider a 
finit number of objects, which are interacting 
with each other. If the number of objects is uncountable, a 
continuous object set can be used in the model. Also if 
the number of objects is finite, but very large, such that an 
analysis of a system with a discrete object set would be very 
difficult (e.g. if the objects are the elementary magnets of 
a metal plate), then it might be more convenient to use a 
continuous object set.

In a model with discrete states of objects, at each point 
of time each object is in one of a finite or countable set of 
states. The future behavior of an object (change of state, 
interaction to other objects) depends on the state of the ob-
ject. States can for example be used to store information 
about the input, which the objects got from other objects 
from or from the environment in the past. Even if there are un-
countable many states of the object in the real world, it 
might be better to use a discrete set of states if only finite 
or countable information is needed to describe the behavior 
of the object. Only if such a discrete set does not suffice 
to store the needed information, a continuous set of states 
might be the better choice for the model.

In a model with discrete interaction, each object can inter-
act at each point of time with only finite or countable many 
other objects. If an interaction occurs not only between fi-
nite or countable many objects, but everywhere in the space 
edg. gravitation force of a planet), then the interaction is 
continuous. In a model with discrete interaction, the inter-
action can be seen as the usage of communication channels: 
Each communication channel connects two objects, and an 
interaction can be described as the transfer of data through 
this channel. A continuous interaction can also be seen as a 
force or an impression of one object to other objects.

While section 3 of this paper uses continuous methods 
for modeling systems, [8] describes how discrete methods 
can be used to model systems. In [8], graphs are used to 
describe which objects can communicate with which other 
objects. Each node in the graph represents one object and 
each edge represents one communication channel. The 
behaviors of the objects are described by stochastic automa-
tons. Since the time is discrete in [8], each automaton has a 
clock, which defines, when the successor state will be com-
puted. Modeling with discrete methods has the advantage 
that a concept of the information theory can be used to 
computed. Modeling with discrete methods has the advantage 
that a concept of the information theory can be used to 
computed. Modeling with discrete methods has the advantage. 
Each object state and interactions with other objects.

A state transition within an object depends on the current 
system has a current state, which can change over the time. 
A state transition within an object depends on the current 
object state and interactions with other objects.

Concerning the example in section 4, the definitions of [8] 
also show that the system is autonomous and it has a high 
level of emergence. Adaptivity and decentralization have 
not been formally defined in [8].

3. MODELING CONTINUOUS SYSTEMS

In this section we give a mathematical definition for mod-
eling systems. The time and the state set are modeled con-
tinuously. For the object set and the interactions, it depends 
on the system to be modeled, whether they should be mod-
eled discrete or not. The definitions in this section can be 
applied for both cases. For modeling systems with discrete 
time and discrete state sets see [8].

A system consists of a set of objects \( V \), which can interact 
with each other, i.e. at each point of time \( t \in \mathbb{R}^+ \) each 
object \( v \in V \) can send information to other objects. These 
objects can be of technical nature (e.g. computers connected 
by a LAN), of biological nature (e.g. a colony of ants) or of other 
kind. At each point of time \( t \in \mathbb{R}^+ \) each object \( v \in V \) of 
the system has a current state, which can change over the time. 
A state transition within an object depends on the current 
object state and interactions with other objects.

To describe also nondeterministic behavior of objects, we 
assume that we have a probability space \( \Sigma = (\Omega, A, P) \), 
where \( \Omega \) is the set of all random events, \( A \) is a \( \sigma \)-algebra 
and \( P : A \rightarrow [0,1] \) is a probability measure on \( A \). This 
probability space will be used to describe all random events 
of the objects occurring during the time. If each object 
\( v \in V \) has its own probability space \( \Sigma_v \), which is independent of the 
probability spaces of the other objects, then we can use 
the product space \( \Sigma = \prod_{v \in V} \Sigma_v \). If there are independent 
probability spaces \( \Sigma_v,t \) for each point of time \( t \in \mathbb{R}^+ \), then we 
can use the product space \( \Sigma = \prod_{v \in V, t \in \mathbb{R}^+} \Sigma_v \). A random 
map between two sets \( A, B \) is a family \( g = (g_v)_{v \in V} \) of maps 
\( g_v : A \rightarrow B \). We use also the notation \( g : A \rightarrow B \) for random 
maps, i.e. \( g \) is seen as a map, which depends on the random 
event. Analogously, a random set \( C = (C_v)_{v \in V} \) is seen as 
a set, which depends on the random event. Now we give a 
formal definition of the mathematical model of systems.

DEFINITION 1. Let \( \Sigma = (\Omega, A, P) \) be a probability space. 
A continuous system \( S = (V, S, A, f, h) \) consists of

- a set \( V \), where the elements of \( V \) are called objects of 
  the system;
A family $S = (S_v)_{v \in V}$ of sets, where each set $S_v$ is a subset of a normed vector space; the elements of $S_v$ are called states of the object $v$.

A set $A$, which is called alphabet;

A family $\lambda = (\lambda_v)_{v \in V}$ of random maps $\lambda_v : S_v \times V \rightarrow A$, where $\lambda_v$ is called output map of the object $v$.

A family $f = (f_v)_{v \in V}$ of random maps $f_v : A^V \times S_v \rightarrow S_v$, where $f_v$ is called change map of the object $v$.

A family $h = (h_v)_{v \in V}$ of random maps $h_v : A^V \times S_v \rightarrow S_v$, where $h_v$ is called hop map of the object $v$.

An initialization of the system $S$ is a random variable $I = (I_v)_{v \in V}$ with $I_v = (I_v)_{v \in V} \in \prod_{v \in V} S_v$ for all $v \in V$.

A family $(s_v)_{v \in V}$ of random maps $s_v : \mathbb{R}_0^+ \rightarrow S_v$ is called behavior of $S$ with respect to an initialization $I \in \prod_{v \in V} S_v$, if for all $v \in V$

- $s_v(0) = I_v$,
- $s_v$ is left-continuous,
- $\{ t \in \mathbb{R}_0^+ \mid s_v(t) \text{ is not differentiable in } t \}$ is a discrete random set,
- for each $t \in \mathbb{R}_0^+$, for which $s_v(t)$ is differentiable, we have $s_v(t) = f_v((s_v(s_v(t)), v))_{v \in V}$, $s_v(t)$,
- \[ \lim_{P \rightarrow +A} s_v(p) = s_v(t) + h_v((\lambda_v(s_v(t), v))_{v \in V}, s_v(t)) \text{ for } t \in \mathbb{R}_0^+ \]

The behavior of the system describes for each object $v \in V$ and each point of time $t \in \mathbb{R}_0^+$ the current state $s_v(t)$ of the object $v$. The initialization defines the states at time $t = 0$. The output map $\lambda_v$ defines the value $\lambda_v(s, w)$ that the object $v$ sends to the object $w$, when $v$ is in the state $s$. The change map $f_v$ describes how the state $s \in S_v$ continuously changes during the time. $f_v$ can be seen as the derivation of the state with respect to the time in dependency of the local inputs from other objects: For each object $v \in V$, the object $w$ can send some data $x_v \in A$ to the object $v$, so the object $v$ receives at the current point of time $t \in \mathbb{R}_0^+$ a family $x = (x_v)_{v \in V} \in A^V$ of values. If the current state of the object $v$ is $s$, then $f_v(x, s)$ describes the direction, in which the state is currently moving: $f_v(x, s) = s$. The hop map $h_v$ describes the changes of the state, when the state is not continuous, i.e. $h_v(x, s)$ can be seen as the value that has to be added to $s$ to get the new state after receiving the local input $x$, so we have the new state $s_{\text{new}} = s + h_v(x, s)$. Since the time is not discrete, we have no “successor state” like in the discrete case (see [8]), so $s_{\text{new}}$ is only the limit from the right: $s_{\text{new}} = \lim_{P \rightarrow +A} s_v(p)$, where $t$ is the current point of time.

Note that the output map $\lambda_v$ cannot depend on the current input of the other objects, because if we would define $\lambda_v$ as a random map from $A^V \times S_v \times V$ to $A$, this would mean that the value that is sent from an object $v \in V$ to another object $w \in V$ at time $t \in \mathbb{R}_0^+$ depends on the value that is sent from the object $w$ to $v$ at the same time $t$ and vice versa. Obviously this would lead to problems, so $\lambda_v$

We also use the notation $I \in \prod_{v \in V} S_v$.

can only depend on the current state but not on the current input values from other objects. But it is still possible to model the situation that $v$ sends a new value $b \in A$ to the object $w$ after it received a certain value $a \in A$ from $w$ by changing the state with the hop map $h_v$. Then the output map $\lambda_v$ can be used to send the new value to $w$, since $\lambda_v$ depends on the state.

It could also be possible to restrict the interactions, e.g. if some objects in the real world are too far away, such that a direct communication is not possible. This could be modeled by a graph, where the edges describe the possible communication channels. In [8] this has been done for discrete modeling. But since many systems in the real world (e.g. ant colonies) are not static, the graph could change during the time. This is the reason why we do not use a graph for the modeling. The problem of missing communication channels can be solved by using a special symbol $null \in A$ for “no communication”.

From Definition 1, it can easily be seen from the hop map $h_v$, in which situations the behavior $s_v$ of the object $v$ is not continuous:

**Lemma 2.** Let $S$ be a system, $s$ be a behavior and $t \in \mathbb{R}_0^+$. The following conditions are equivalent:

- $s_v$ is continuous at time $t$.
- $h_v((\lambda_v(s_v(t), v))_{v \in V}, s_v(t)) = 0$

### 4. Modeling Slot-Synchronization

In this section we apply the definitions of the previous section to model an algorithm for self-organized slot-synchronization in wireless networks [12]. In such a network, we have some objects which can communicate with the other objects. For this communication, the time is divided into time slots. Since there is no central clock, which defines when a slot begins, the objects need to apply a slot synchronization. An algorithm for this slot synchronization is proposed in [12]. It is based on the model of pulse-coupled oscillators by Mirollo and Strogatz [9].

The clock of each object is described by a phase function $\phi$ which starts at $0$ and increases over time until it reaches the threshold value $\phi_{th} = 1$. The object then sends a “firing pulse” to its neighbors for synchronization. After receiving the firing pulse, the other objects adjust their own phase functions by adding $\Delta \phi := (\alpha - 1)\phi + \beta$ to $\phi$, where $\alpha > 1$ and $\beta > 0$ are constants.

In [12] delays (e.g., transmission delay, decoding delay) are introduced into this algorithm. Let $T > 0$ be a constant. The duration of an uncoupled period (i.e. if no pulses are received from other objects) is $2T$. Now this period is divided into four states (see Figure 1). Let $\gamma \in [0, 2T]$ be a time instant. Then the object is in a

- waiting state, if $\gamma \in [0, T_{\text{wait}}) =: I_{\text{wait}}$
- transmission state, if $\gamma \in [T_{\text{wait}}, T_{\text{wait}} + T_{\text{rx}}) =: I_{\text{rx}}$
- refractory state, if $\gamma \in [T_{\text{wait}} + T_{\text{rx}}, T_{\text{wait}} + T_{\text{rx}} + T_{\text{refr}}) =: I_{\text{refr}}$
- listening state, if $\gamma \in [T_{\text{wait}} + T_{\text{rx}} + T_{\text{refr}}, 2T) =: I_{\text{rx}}$

where the constants are defined as follows:

- $T_{\text{rx}}$: Delay for the transmission of a value.
Therefore, a state \( s \) is a triple \( (\phi, \gamma, D) \). Now let us consider the output map \( \lambda_o \). Only the interval \( I_{Rx} \) is used for the transmission, so during this interval, the output value 1 is sent to all other objects \( w \in V \). During the intervals \( I_{wait}, I_{Rx}, I_{refr} \) there is no pulse, so the output value 0 is sent to all other objects.

The change map \( f_o \) describes the derivatio \( s' \) of the state \( s \). Let us first consider the intervals \( I_{wait}, I_{Rx}, I_{refr} \): During these intervals, the phase function stays constant 0, so \( \phi = 0 \). Also the value \( D_w \) for \( w \in V \) can stay constant at a negative value, so \( D_w = 0 \). Only the variable \( \gamma \) is changed. Since \( \gamma \) is the time elapsed since the beginning of the cycle, we have \( \gamma = 1 \). During the interval \( I_{Rx} \) we have a different change of the state: In this interval, the phase function \( \phi \) increases uniform (if no pulses arrive from other objects) until the threshold is reached. For an uncoupled system, the threshold \( \phi_{th} \) should be reached at the end of the listening period, so during the interval of length \( T_{Rx} \), the phase function \( \phi \) grows linearly from 0 to 1, which implies \( \phi = \frac{T_{Rx}}{D_v} \). The value \( \gamma \) still grows with gradient \( \gamma = 1 \) like above. During a pulse \( x_w = 1 \) of another object \( w \in V \), the value \( D_w \) stays constant at \( T_{dec} \), so we have \( D_w = 0 \). After the end of this pulse, \( D_w \) decreases with the gradient \( D_w = 1 \).

The hop map \( h_v \) describes the changes of the state, whenever the state is not continuous. In \( I_{wait}, I_{Rx}, I_{refr} \), the values \( \phi \) and \( D_w \) for \( w \in V \) are constant and \( \gamma \) is continuous, so in this case the hop map is 0. Now consider the interval \( I_{Rx} \). The value \( \gamma \) is still continuous. For a pulse from another object \( w \), the value \( D_w \) must be set to \( T_{dec} \). Since the hop map describes only the value that has to be added to the old state, this can be done by adding \( -D_w + T_{dec} \) to the old value \( D_w \) to get the new value \( T_{dec} \). After the decoding delay also an adjustment of \( \phi \) has to be done. If the threshold is not reached by this adjustment, this is done by adding \( \Delta \phi \) to the current value of \( \phi \) for each pulse that has been decoded, i.e. for \( D_w = 0 \). For \( D_w > 0 \), the pulse has not yet been decoded, so the adjustment of \( \phi \) need not be done yet. \( D_w < 0 \) indicates irrelevance (either the adjustment has already been done or no pulse has been received from \( w \)), also in this case no adjustment to \( \phi \) has to be done. After reaching the threshold \( \phi_{th} = 1 \), the phase function must be set to 0, and also \( \gamma \) must be initialized to 0 to begin the new cycle. In this case we use the hop map to get into the state \((0, 0, (1-w_{<}w_{<}V))\).

Now we give a formal definition of these maps \( \lambda_v, f_v, h_v \). Let \( S_v = [0,1] \times [0,2T] \times (-\infty, T_{dec}] \) and \((\phi, \gamma, D) \in S_v \). Definition of the output map \( \lambda_v \):

- For \( \gamma \in I_{wait} \cup I_{refr} \cup I_{Rx} \) define \( \lambda_v(\phi, \gamma, D, w) = 0 \).
- For \( \gamma \in I_{Rx} \) define \( \lambda_v(\phi, \gamma, D, w) = 1 \).

Definition of the change map \( f_v \):

- For \( \gamma \in I_{wait} \cup I_{Rx} \cup I_{refr} \) define \( f_v(x, \phi, \gamma, D) = \left( 0, 0, (1-w_{<}w_{<}V) \right) \).
- For \( \gamma \in I_{Rx} \) define \( f_v(x, \phi, \gamma, D) = \left( \frac{1}{T_{Rx}}, 1, (x_w - 1_{w_{<}w_{<}V}) \right) \).

Definition of the hop map \( h_v \):

- For \( \gamma \in I_{wait} \cup I_{Rx} \cup I_{refr} \) define \( h_v(x, \phi, \gamma, D) = (0, 0, 0) \).
- For \( \gamma \in I_{Rx} \) let \( \phi' = \Delta \phi \cdot [w \in V \setminus \{v\} \mid D_w = 0] \) with \( \Delta \phi := (\alpha - 1)\phi + \beta \). For \( w \in V \) let \( D_w = -D_w + T_{dec} \) for \( x_w = 1 \) and \( D_w = 0 \) for \( x_w = 0 \). Let \( h_v(x, \phi, \gamma, D) = (\phi', 0, D') \) for \( \phi' < 1 \) and \( h_v(x, \phi, \gamma, D) = (\phi', -\gamma, (1-D_w - 1_{w_{<}w_{<}V}) \) for \( \phi' \geq 1 \).

The following Theorem shows that we have successfully modeled the system [12].

THEOREM 3. For each initialization \( I \), the system described above has exactly one behavior \( s = (s_v)_{v \in V} \). The value \( \gamma \) of
the current state \( s_v(t) \) for an object \( v \) runs cyclic through the intervals \( I_{\text{wait}}, I_{\text{rx}}, I_{\text{refr}}, I_{\text{refr}} \). After the end of the first cycle, the states \( (\phi, \gamma, D) = s_v(t) \) of the behavior have the following properties:\(^2\)

1. In each cycle, \( \gamma \) starts at 0 and grows linearly with \( \dot{\gamma} = 1 \) until the end of the cycle and then restarts at 0.
2. During the intervals \( I_{\text{wait}}, I_{\text{rx}}, I_{\text{refr}}, \) the object \( v \) does not send a pulse.
3. During the interval \( I_{\text{rx}}, \) the object \( v \) sends a pulse.
4. During the intervals \( I_{\text{wait}}, I_{\text{tx}}, I_{\text{refr}}, \) the phase function \( \phi \) is constant 0.
5. During the intervals \( I_{\text{wait}}, I_{\text{tx}}, I_{\text{refr}}, \) the value \( D_w \) is constant –1.
6. If the interval \( I_{\text{rx}} \) starts at time \( t \) (i.e. \( \gamma = T_{\text{wait}} + T_{\text{rx}} + T_{\text{refr}} \)) then \( \phi \) is continuous in \( t \).
7. During the interval \( I_{\text{rx}}, \) if \( \phi = 1 \), then the current cycle ends and the next cycle starts.
8. During the interval \( I_{\text{rx}}, \) with \( 0 < \phi < 1 \), the phase function \( \phi \) is not differentiable at time \( t \) iff \( s_w(t+T_{\text{dec}}) \) was a listening state and a pulse from another object \( w \) ended at time \( t + T_{\text{dec}} \).
9. During the interval \( I_{\text{rx}}, \) if the phase function \( \phi \) is differentiable, then \( \dot{\phi} = \frac{1}{T_{\text{rx}}} \).
10. If a pulse from another object \( w \in V \) ends during the interval \( I_{\text{rx}} \) of the object \( v \) and \( s_v(t+T_{\text{dec}}) \) is still in the listening state, then the phase function \( \phi \) is adjusted by adding \( \Delta \phi \) to \( \phi \) (if the new value is smaller than 1) at time \( t + T_{\text{dec}} \). This is done for each object \( w \neq v \), for which the pulse ends at time \( t \). If the new value is greater or equal to 1, then the current cycle ends and the next cycle starts.
11. During the interval \( I_{\text{rx}}, \) the value \( D_w \) decreases linearly with gradient –1 during the time, where no pulse arrives from \( w \).
12. During the interval \( I_{\text{rx}}, \) the value \( D_w \) is constant \( T_{\text{dec}} \) during the time, where a pulse arrives from \( w \).

Proof. A formal proof of this theorem would be out of scope of this paper, but it is straightforward to check the following conditions:

- Every behavior of \( S \) satisfies the properties 1-12, because of the definitions of \( h_v, f_v \) and \( h_v \) of \( S \).
- The random maps \( s_v \) for \( v \in V \) are uniquely determined by the initialization and the properties 1-12, i.e. for a given initialization, there is only one family of random maps \( s = (s_v)_{v \in V} \) satisfying 1-12, because these properties describe the whole course of the random maps.
- For each random map \( s_v \) that satisfies 1-12, the properties (B2)-(B5) of Definition 1 are also satisfied.

After verifying these conditions, the assertion of this theorem can be easily deduced. □

\(^2\)If we assume a meaningful initialization, these properties are also satisfied during the first cycle.

The simulation results in [12] show that during the run of the system, groups of synchronizations are built, i.e. inside each group we have good synchronization (each object of the group fires at nearly the same time like the other objects of the group), and if we wait long enough, then there are only two groups left firing \( T \) time units apart from each other.

This system satisfies the four main properties of self-organization:

- The system is autonomic, because no external control is needed.
- The synchronization of the objects is emergence, since this is a global property of the system, which is induced by the local interactions.
- The system is adaptive to small changes in the environment (adding more objects, removing some existing objects, etc.).
- The control of the system is completely decentralized.

5. CONCLUSION AND FUTURE WORK

For analyzing properties of complex systems, the real world system can be transformed into a mathematical model. In this paper we compared continuous modeling with discrete modeling and we gave a formal definition for modeling continuous complex systems. Then this theory has been applied to model the slot-synchronization algorithm of [12].

While a level of autonomy and the level of emergence can be defined formally in discrete systems (see [8]) with the concept of entropy, it is still an open problem, how they can be defined formally for continuous systems. Also a formal definition of the level of adaptivity and the level of decentralization is left for future work.

6. REFERENCES


