Self-Organization in Networking

Advanced Mobile Communication Networks
Motivation for Self-Organization

- Problems of today’s networks
  - Heterogeneity, dynamics, scalability
  - Managability
  - Operational cost

- Self-organization offers
  - Autonomous optimization and repair
  - Automation of control
  - Scalability

- Some applications
  - Network management in LTE (SON: self-optimization and self-healing)
  - Protocol functions: TCP congestion control, 802.11 PCF, AODV, …
  - Autonomous operation of UAV-supported DTNs
From Central to Self-organized Systems

- **Monolithic/centralized systems**
  Monolithic: Systems consisting of a single computer, its peripherals, and perhaps some remote terminals
  Centralized: systems with a well-defined centralized control process

- **Distributed systems**
  A collection of independent subsystems that appears to the application as a single coherent system

- **Self-organizing autonomous systems**
  Autonomously acting individual systems performing local programs and acting on local data but participating on a global task, i.e. showing an emergent behavior
Self-organization: Definitions

Yates et al. (1987)

“Technological systems become organized by commands from outside, as when human intentions lead to the building of structures or machines.

But many natural systems become structured by their own internal processes: these are the self-organizing systems, and the emergence of order within them is a complex phenomenon that intrigues scientists from all disciplines.”

Camazine et al. (2003)

“Self-organization is a process in which pattern at the global level of a system emerges solely from numerous interactions among the lower-level components of a system.

Moreover, the rules specifying interactions among the systems’ components are executed using only local information, without reference to the global pattern.”
Self-organizing Systems – Operational Principles

Local system control

Local interactions (environment, neighborhood)

Simple local behavior
Example from Nature: Ants (Double Bridge Experiment)

Ants start with equal probability

The ant on shorter path has a shorter time from it’s nest to the food.

The density of pheromone on the shorter path is higher

After some time, the shorter path is almost exclusively used
Example from Nature: Swarm Intelligence (SI)

- A computational technique for solving distributed problems inspired from biological examples provided by
  - social insects such as ants, termites, bees, and wasps and by swarm, herd, flock, and
  - shoal phenomena such as fish shoals

- An approach for controlling and optimizing distributed systems

- Resilient, decentralized, self-organized technique
Swarm Intelligence – Notable Features

**Autonomy:**
- The system does not require outside management or maintenance. Individuals are autonomous, controlling their own behavior both at the detector and effector levels in a self-organized way.

**Adaptability:**
- Interactions between individuals can arise through direct or indirect communication.

**Scalability:**
- SI abilities can be performed using groups consisting of a few, up to thousands of individuals with the same control architecture.

**Flexibility:**
- No single individual of the swarm is essential, that is, any individual can be dynamically added, removed, or replaced.

**Robustness:**
- No central coordination takes place, which means that there is no single point of failure.

**Massively parallel:**
- Tasks performed by each individual within its group are the same.

**Self-organization:**
- The intelligence exhibited is not present in the individuals, but rather emerges somehow out of the entire swarm.
Example from Chemistry

Pattern formation in the Belousov-Zhabotinski reaction

Photography by Juraj Lipscher
Self-organization – More Examples

**Examples** of self-organized systems:

- **biology**: Ants building a ant colony, a flock of birds flying in some joint and coordinated way
- **economy**: capitalistic economy with minimal rules and lots of individuals with ideas and goals representing local algorithms
- **human society**: social networks, new political forces, communities, associations, etc.

**But**: From time to time there may be unexpected breakdowns or crashes of the system, e.g.

- **human societies**: war, civil war
- **economy**: crash of financial markets
- **nature**: natural catastrophes
Self-organizing Systems – Operational Principles

Local system control

Local interactions (environment, neighborhood)

Simple local behavior
## Self-organization – Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| No central control    | No global control system  
No central information, information is distributed  
Subsystems perform completely autonomous |
| Emerging structures   | Global behavior or functioning of the system emerges in form of observable pattern or structures                                           |
| Resulting complexity  | Even if the individual subsystems can be simple as well as their basic rules, the resulting overall system becomes complex and often unpredictable |
| High scalability      | No performance degradation if more subsystems are added to the system  
System performs as requested regardless of the number of subsystems |
Properties: Self-organization and Emergence

Definition **Self-Organization** (Camazine rephrased)

Self-organization is a process in which structure and functionality (pattern) at the global level of a system emerge solely from numerous interactions among the lower-level components of a system without any external or centralized control. The system's components interact in a local context either by means of direct communication or environmental observations without reference to the global pattern.

Definition **Emergence**

Emergent behavior of a system is provided by the apparently meaningful collaboration of components (individuals) in order to show capabilities of the overall system (far) beyond the capabilities of the single components.
## Properties: Self-X Capabilities

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-configuration</td>
<td>Methods for (re-)generating adequate configurations depending on the current situation in terms of environmental circumstances</td>
</tr>
<tr>
<td>Self-management</td>
<td>Capability to maintain systems and devices depending on the current system parameters</td>
</tr>
<tr>
<td>Adaptability</td>
<td>Ability of the system's components to adapt to changing environmental conditions</td>
</tr>
<tr>
<td>Self-diagnosis</td>
<td>Mechanisms to perform autonomous checks and to compare the results with reference values</td>
</tr>
<tr>
<td>Self-protection</td>
<td>Capability to protect the system and its components against unwanted or even aggressive environmental influences</td>
</tr>
<tr>
<td>Self-healing</td>
<td>Methods for changing configurations and operational parameters of the overall system to compensate failures</td>
</tr>
<tr>
<td>Self-repair</td>
<td>Similar to self-healing but focusing on actual repair mechanisms for failing system parts</td>
</tr>
<tr>
<td>Self-optimization</td>
<td>Ability of the system to optimize the local operation parameters according to global objectives</td>
</tr>
</tbody>
</table>
Characteristics of Self-organized Systems

Self-organizing systems are dynamic and exhibit emergent properties.

Since these system-level properties arise unexpectedly from nonlinear interactions among a system’s components, the term emergent property may suggest to some a *mysterious property that materializes magically.*

**Example: growth rate of a population**

\[ x_{n+1} = r x_n (1 - x_n) \]

- \( 0 < r < 1 \): extinction
- \( 1 < r < 3 \): constant size after several generations
- \( 3 < r < 3.4 \): oscillating between two values
- \( 3.4 < r < 3.57 \): oscillating between four values
- \( r > 3.57 \): “deterministic” chaos
Consequences of Emergent Properties

A small change in a system parameter can result in a large change in the overall behavior of the system

- Adaptability
- Flexibility

Role of environmental factors

- Specify some of the initial conditions
- Positive feedback results in great sensitivity to these conditions

Self-organization and the evolution of patterns and structure

- Intuitively: generation of adaptive structures and patterns by tuning system parameters in self-organized systems rather than by developing new mechanisms for each new structure
- However: the concept of self-organization alerts us to the possibility that strikingly different patterns result from the same mechanisms operating in a different parameter range

Simple rules, complex patterns – the solution to a paradox?
Design of Self-organized Systems

- Positive and negative feedback
- Interactions among individuals and with the environment
- Probabilistic techniques

Local interactions (environment, neighborhood)

Local system control

Simple local behavior
Design: Positive and negative Feedback

Simple feedback

Feedback

Input -> System state -> Output

Amplification problems

Snowballing effect

Implosion effect
Design: Interactions among Individuals and Environment

- Direct communication among neighboring systems
- Indirect communication via the environment (stigmergy)
- Interaction with (stimulation by) the environment

Stigmergy is a mechanism of spontaneous, indirect coordination between agents or actions, where the trace left in the environment by an action stimulates the performance of a subsequent action, by the same or a different agent. Stigmergy is a form of self-organization. It produces complex, apparently intelligent structures, without need for any planning, control, or even communication between the agents. As such it supports efficient collaboration between extremely simple agents, who lack any memory, intelligence or even awareness of each other.
Design: Probabilistic Techniques

- Examples: stochastic processes, random walk
- Objectives: leaving local optima, stabilization

![Simulation results]

---

Pick-up chip → no → carry a wood chip → no → drop chip

 Initialization → random walk → bumped into a wood chip → yes → carry a wood chip → yes
Design Paradigms for Self-organized Systems

Paradigm #1:
Design local behavior rules that achieve global properties

Paradigm #2:
Do not aim for perfect coordination: exploit implicit coordination

Paradigm #3:
Minimize long-lived state information

Paradigm #4:
Design protocols that adapt to changes
Design Paradigms for Self-organized Systems

Paradigm #1: Design local behavior rules that achieve global properties
To design a network function that establishes a global property, this paradigm is to distribute the responsibility among the individual entities: no single entity is "in charge" of the overall organization, but each contributes to a collective behavior. Following this paradigm, localized behavior rules must be devised, if applied in all entities, automatically lead to the desired global property (or at least approximate it). "localized" mean that entities have only a local view of the network and interact with their neighbors and changes in the network have initially only local consequences.

Paradigm #2: Do not aim for perfect coordination: exploit implicit coordination
Implicit coordination means that coordination information is not communicated explicitly by signaling messages, but is inferred from the local environment. A node observes other nodes in its neighborhood; based on these observations, it draws conclusions about the status of the network and reacts accordingly.

Paradigm #3: Minimize long-lived state information
To achieve a higher level of self-organization, the amount of long-lived state information should be minimized. In general, the more localized the interactions are and the less coordination is needed, the less state information must be maintained.
Design Paradigms for Self-organized Systems

Paradigm #4: Design protocols that adapt to changes

The need for the capability of nodes to react to changes in the network and its environment typically arises from changed resource constraints, changed user requirements, node mobility, or node failures. Since there are no centralized entities that could notify the nodes about changes, each node has to continuously monitor its local environment and react in an appropriate manner. Three levels of adaptation can be distinguished.

– Level 1: A protocol is designed so that it can cope with changes, such as failure and mobility.
– Level 2: A protocol is designed to adapt its own parameters (e.g., value of timers, cluster size) as a reaction to changes in order to optimize system performance.
– Level 3: A protocol is designed so that it realizes if the changes are so severe that the currently employed mechanism is no longer suitable. To detect such situations the environment is observed, and significant changes in major parameters trigger a fallback or alternative solution.

Typically, these levels of adaptation are combined using control loops.

Design Paradigms for Self-organized Systems

**Paradigm #1**
- Local properties (divide and conquer)
- Local behavior rules

**Paradigm #2**
- Tolerable conflicts and inconsistencies (conflict detection and resolution)
- Implicit coordination

**Paradigm #3**
- Synchronized state (discovery mechanisms)
- Reduced state

**Paradigm #4**
- Definition of severe changes and reactions (monitoring and control)
- Adaptive algorithms

**Resulting protocol**
- Behavior rules, messages, state, and control

Required functionality – system behavior (objectives)
Limitations of Self-organization

- **Controllability**
  - Predictability vs. scalability

- **Cross-mechanism interference**
  - Composition of multiple self-organizing mechanisms can lead to unforeseen effects

- **Software development**
  - New software engineering approaches are needed

- **System test**
  - Incorporation of the unpredictable environment
Self-organized vs. Centralized Control

Self-organizing networks

- Local state
- Neighbor information
- Probabilistic methods

Networking functions (e.g. for global connectivity and efficient resource usage)

Centralized networks

- Global state (globally optimized system behavior)

Implicit coordination

Explicit coordination
Applications of Self-organization to Networking

• Media Access Control:
  – CSMA/CD (802.3/Ethernet)
  – 802.11 DCF
  – Cognitive radio in interweaved mode with infrastructure network

• Load/congestion control:
  – Exponential backoff (CSMA systems)
  – TCP flow control

• Routing: AODV route discovery

• Election: cluster head selection in sensor networks (LEACH)

• Energy management: 802.11 ad hoc mode

• System management:
  – LTE load control
  – LTE energy saving
  – LTE mobility robustness optimization (HO parameter optimization)
  – 2-4G inter-system load balancing

• ...
Example: 802.11 DCF MAC

Distributed Coordination Function (DCF) with Exponential Backoff

- **DIFS**
- **bo**
- **bor**
- **busy**
- **boe**
- **boe**
- **boe**
- **boe**
- **boe**

**Packet arrival at MAC**

**busy medium not idle (frame, ack etc.)**

**boe elapsed backoff time**

**boe residual backoff time**
Example: TCP Congestion Control and RED

TCP flow control

Loss of a single packet (≥3 Dupacks)

Major problem (Timeout)

Random Early Detection (RED)

Always drop

Non-zero likelihood of drop

Never drop
Example: Information Management

Gossip-based Aggregation
- Compute a (global) aggregate value from distributed data
  - load balancing in networks
  - monitoring in industrial control applications
- Each node
  - maintains a local approximate value of the aggregate
  - communicates periodically with randomly chosen neighbors and updates the approximate value
- Only local pairwise interaction!

![Diagram of nodes and connections]

Average value?
1. 2,5
2. 3,25
3. 3,75
4. 3,5
Example: Cluster Head Election in Sensor Networks

LEACH: Low-Energy Adaptive Clustering Hierarchy

Principles

- nodes elect themselves to become cluster heads at any given time with a certain probability (in HEED extension based on remaining energy)
- the cluster heads broadcast their status to the other nodes in the network
- each node determines to which cluster it wants to belong by choosing the closest cluster head (e.g. which requires the minimum communication energy)
Example: Inter-system Load Balancing

Force-based selection of target cell

- Criteria/forces:
  - supported QoS in target cell (user view)
  - free capacity of target cell
  - cost per bit in target cell
  - migration penalty and handover cost

Problem: Stability and Predictability

Fires don’t matter!

Yield = density - loss

Cold

Density
Problem: Stability and Predictability

Everything burns!
Problem: Stability and Predictability
Problem: Stability and Predictability

Edge Of Chaos (EOC)
Self-Organized Criticality (SOC)

Claims:
• Interesting phenomena is at criticality (or near a bifurcation)
• Life, networks, the brain, the universe and everything are at “criticality” or the “edge of chaos.”
Conclusions

Self-organization: what does it mean?
– distributed, autonomous control of systems
– simple local algorithms or behavior
– interaction with neighbors and/or environment only

Lots of applications of self-organization already
– self-organization keeps today’s Internet together
– development and tuning of self-organized algorithms takes time
– stability is an issue

Nature
– provides a rich source of inspiration for computer systems.
– has many features that are desirable for distributed computing such as auto-configuration, auto-organization, autonomy, scalability, flexibility, robustness, emergent behavior, and adaptability
References

Self-organization in general


Application of Self-organization to communication networks


Bioinspired Algorithms