Self-organizing Networked Systems

Wilfried Elmenreich

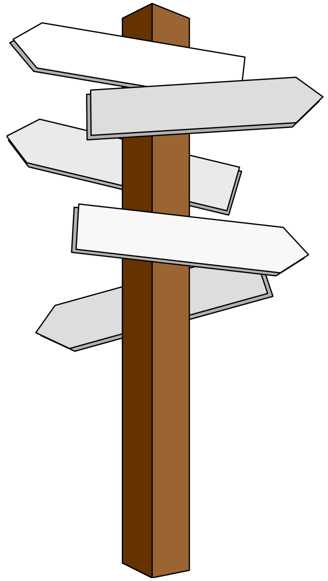
Lakeside Labs / Mobile Systems Group
Alpen-Adria Universität Klagenfurt

ITG Meeting - Cooperation and Self-Organization in Communication Networks

June 29 2009

Overview

- Motivation for self-organized systems
- What is a self-organizing system
- How to design self-organizing systems
- Example applications
- Conclusion and summary



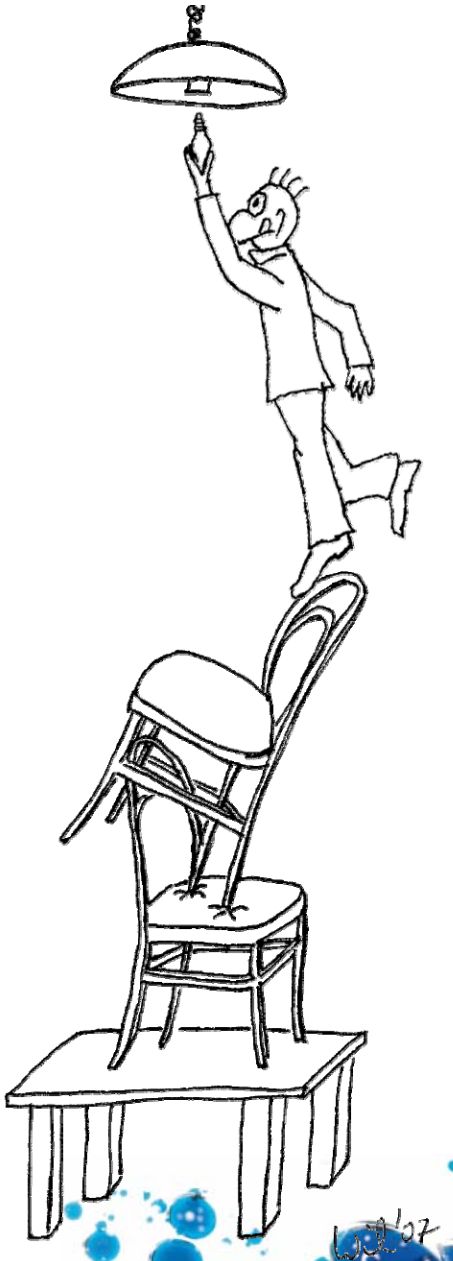
Motivation

Image from Wikimedia Commons

- Increasing complexity in applications (more nodes, modes, protocols, …)
- Traditionally, applications are built like a puzzle – all parts have to be exactly in the right place

Motivation

- Many applications are fragile and single-purpose
- Small changes can lead to a crash of the system
- Requirement for robust networked applications
- …and scalable
- …and adaptive
- …supporting cheap hardware
- …

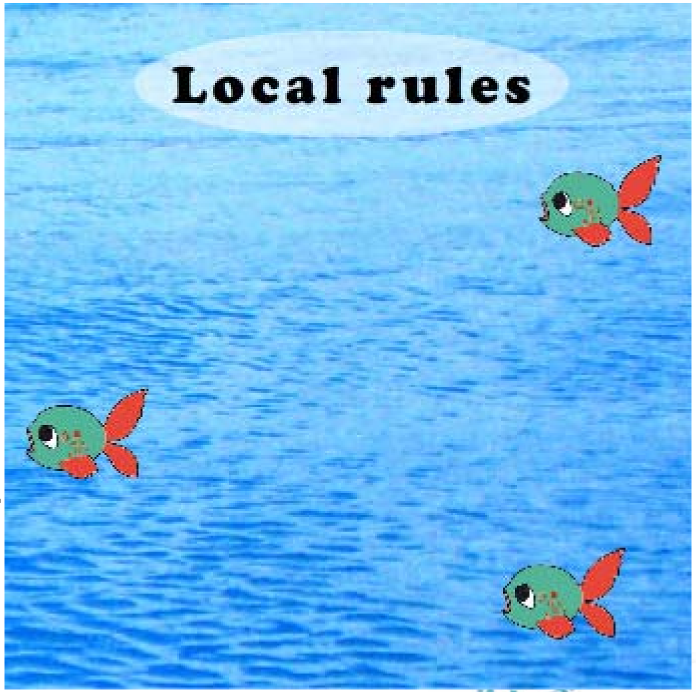


What is Self-Organization

- *„A system is called self-organizing if it is autonomous, adaptive and its organization is an emergent property."* de Meer, Uni Passau
- *„Self-organization is a process of attraction and repulsion in which the internal organization of a system, normally an open system, increases in complexity without being guided or managed by an outside source. Self-organizing systems typically (but not always) display emergent properties."* Wikipedia, 2009
- *„A self-organizing system (SOS) is a set of entities that obtains global system behavior via local interactions without centralized control."* Research Days 2008
- *"Self-organization is a way of observing systems, not an absolute class of systems."* Gershenson, Heylighen, 2003

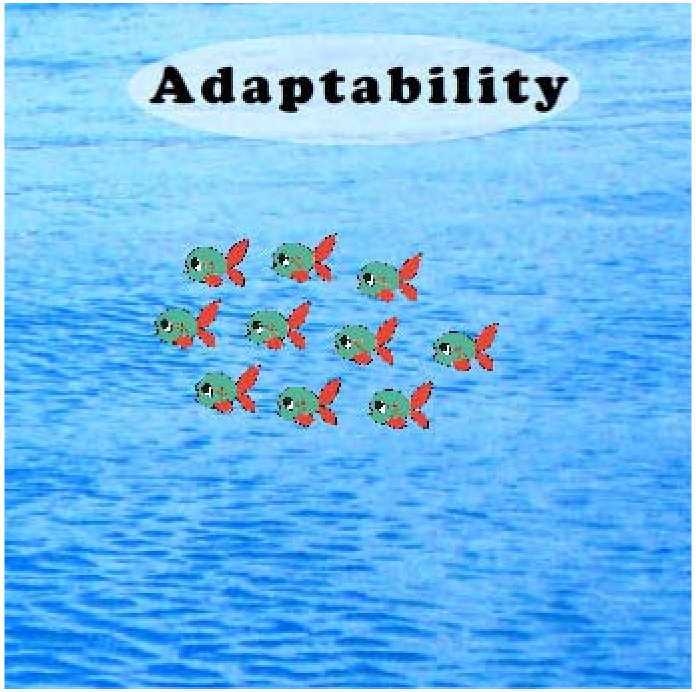
A School of Fish as SO example

- Several fish with simple behavior ("local rules")
  1) Swim where other fishes are
  2) Avoid coming too close
  3) Being attracted by food
  4) Flee from predators

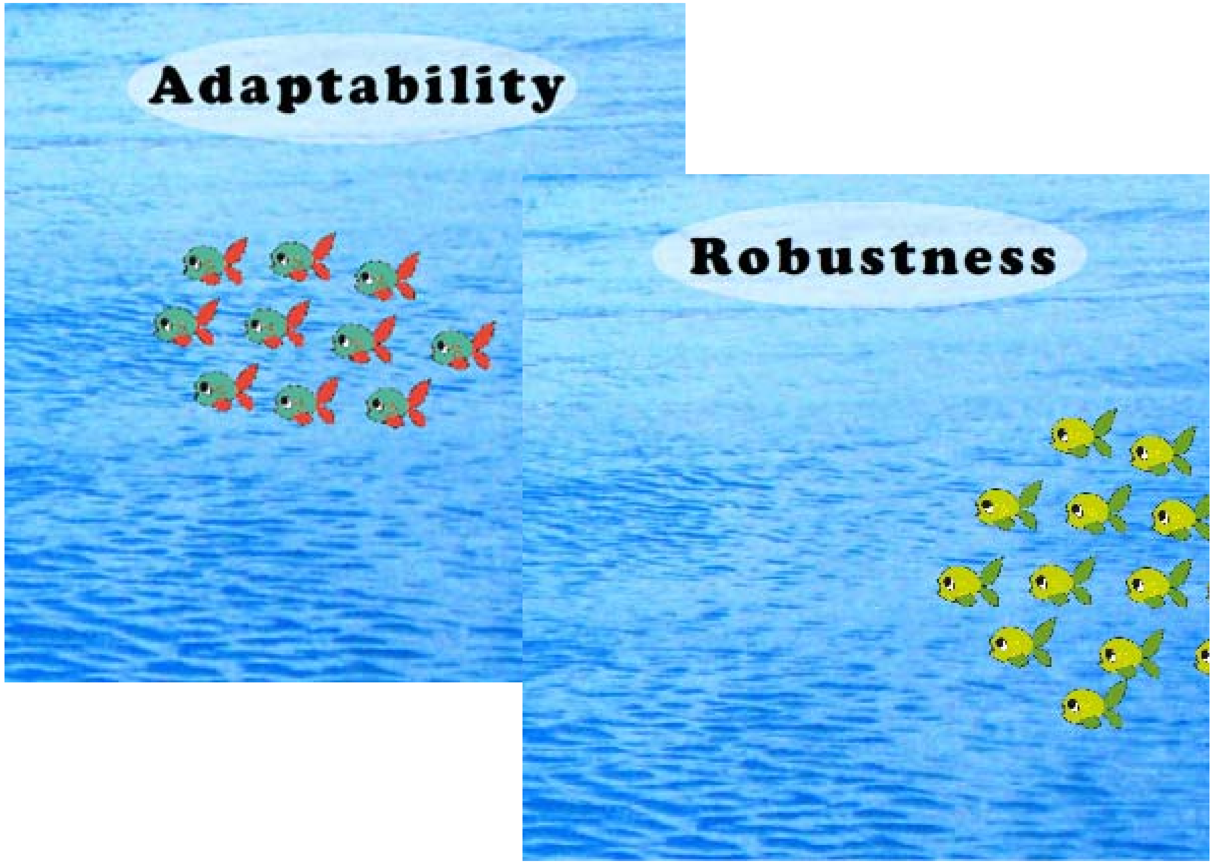


- Example from Prehofer/Bettstetter
- Animation by Fehérvári

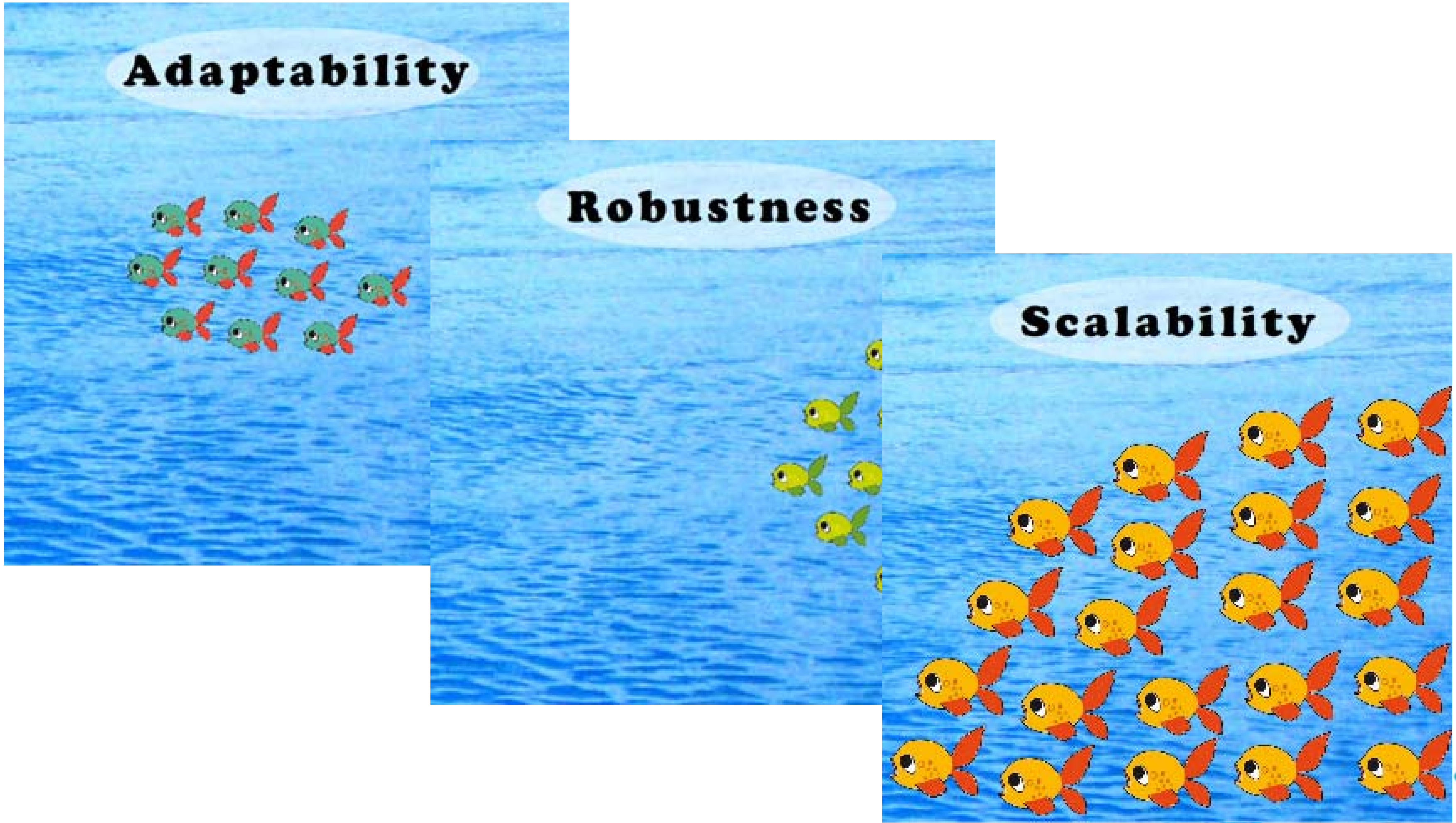
Emergent Behavior of Fish School



Emergent Behavior of Fish School



Emergent Behavior of Fish School



Fish School Example Analysis

- **Individuals ("Fish")**
- Local observations
- Interaction with other fish
- No centralized control
- Simple behavior

⇩ Emergence

- **Overall system ("Fish school")**
- Complex behavior
- Robust, adaptive and scalable

Properties of Self-Organizing Systems

- Decentralized control
- Localized perception/actions
  - Not necessarily identical components
- Implicit communication/stigmergy
  - Navigation system tells fastest way – automatically avoids jammed roads
- Competing forces that tend towards an equilibrium
- At least one attraction state where a wide set of parameter settings move to
  - The emerging structure can be often observed as being interesting/complex, but this is not a primary property

Design Methods for SOS

- Main Problem: define local interaction properties in order to achieve desired system behavior
- Tuning the system manually
  - Test different settings
- Tuning the system using heuristic search
- Bio-inspired design
  - Top-down approach
  - Bottom-up approach
- Analyzing a „perfect“ solution

Tuning the system manually…

- SO systems are typically *non-linear*, i.e. the resulting behavior cannot be extrapolated by summing up the behavior of the single components
- Designer must get an understanding what a specific rule change might achieve
- Experience/intuition to identify the critical parameters
- C. Gershenson introduced notion of local and global *friction* to gain understanding

Tuning the system manually… (2)

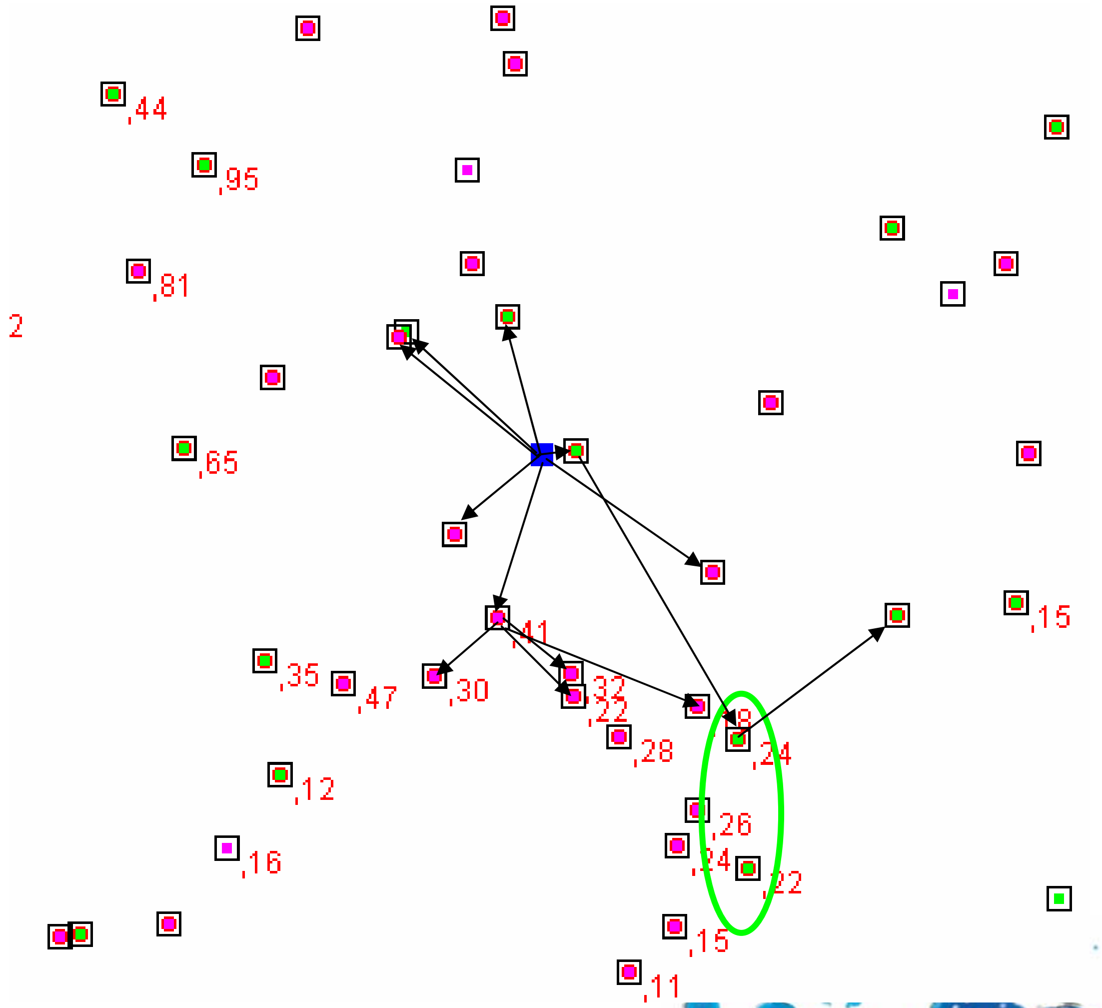
- Problems
  - A minimum on local friction does not mark a minimum on global friction
  - Changes in local behavior often have counter-intuitive results (cf. Resnik‘s experiment at MIT)
- As a consequence, often trial-and-error are used
- Large search space ☹

Example: defining backbone networks for ad hoc wireless sensor networks

- Idea: have two distinct backbone networks that connect every node to a base station
- Only one backbone needs to be active, the other nodes can sleep and save energy
- Algorithm to generate backbones uses
  - A decision function for each node to join one of the backbones
  - Attraction (a backbone should extend in order to keep connectivity)
  - Repulsion (a node with many neighbors of one type should better become a node of another type in order to keep balance)

Example: defining backbone networks for ad hoc wireless sensor networks

- Network grows beginning at base station
- Nodes that are determined to one backbone become colored
- Application: WSN detecting events

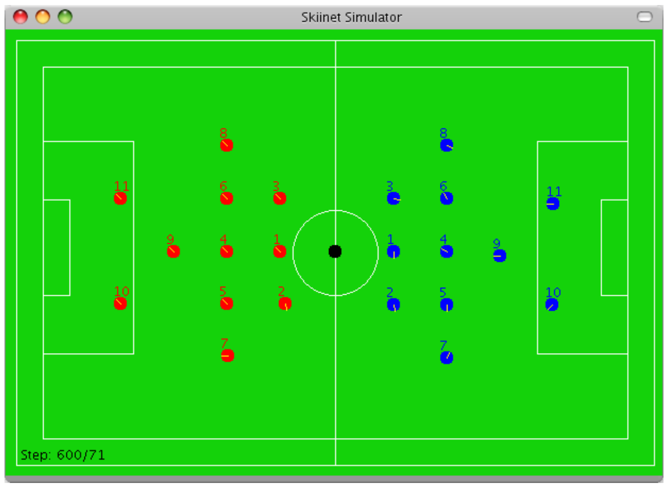


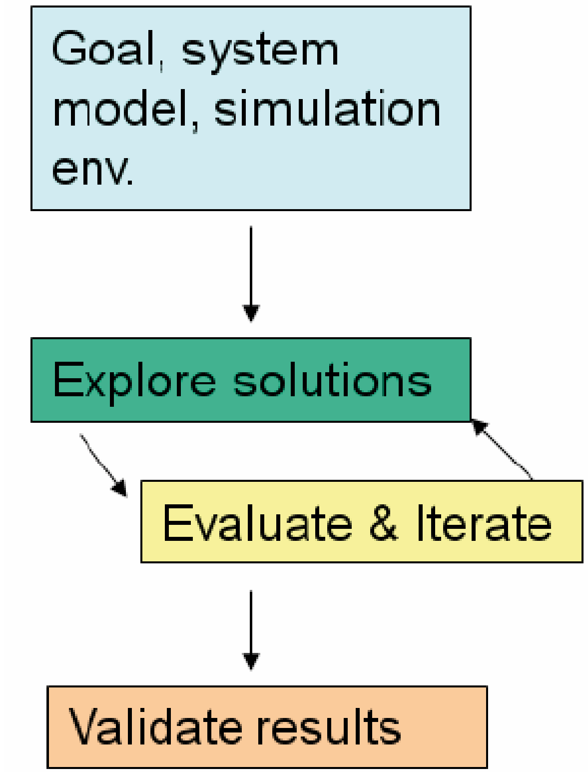
Using heuristic search

- Conquer the search space using heuristic search methods
- E.g., genetic algorithms, simulated anneahling, swarm-based optimization
- Representation of rules must be evolvable (i.e., mutation and Xover operations must be defined)

Example for heuristic search approach

- Robot soccer simulation as a testbed
- Using a neural network for modeling the behavior
- Evaluation of fitness via simulator





Top down-approach to bio-inspired design

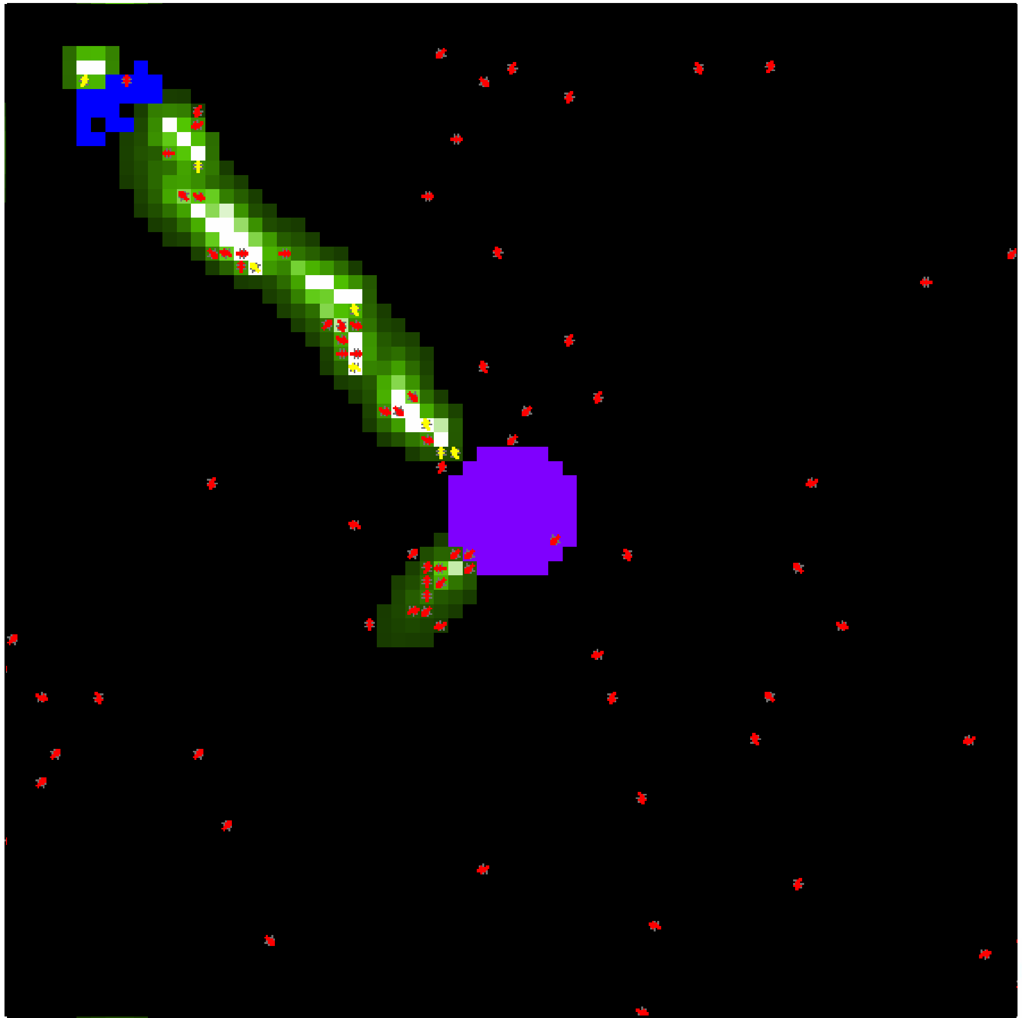
- Look for natural examples that solve your problem
- Analyze the solution and its principles
- Re-build the solution in a technical application
- Examples:
  - Gliding flight of birds -> aeroplane
  - Form of wings -> winglets
  - Robots with autonomously controlled legs

Bottom up-approach to bio-inspired design

- Derive principles by analyzing natural systems of different purposes (basic research)
- Abstract principle from biological context
- Use principles in technical applications
- Examples
  - Ant foraging -> packet routing
  - Seeds distribution by help of animals -> velcro
  - Artificial neural networks

Example: Foraging of Ants

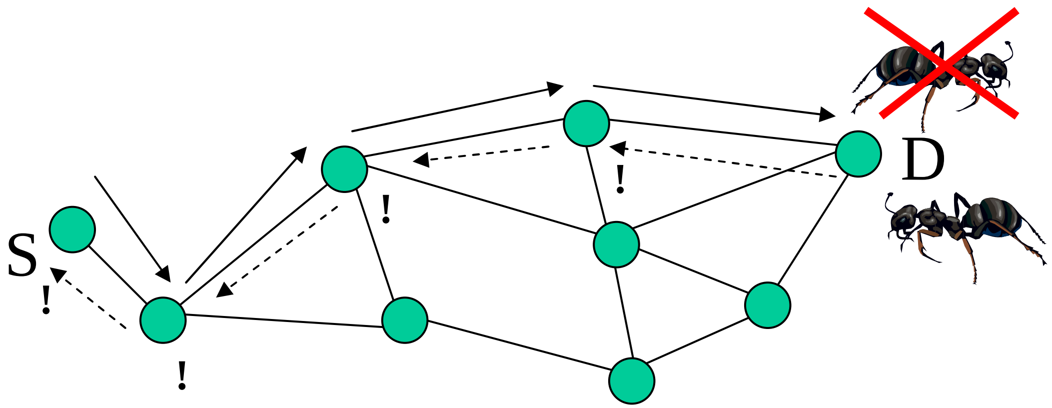
- Ants swarm out to search for food sources
- Successful ants mark their way back using pheromone
- Other ants follow the pheromone track
- Most frequented (nearest) food source has strongest trail
- Pheromone trails disappear over time



Screenshots aus StarLogo-Projekt „ants“

Ant Packet Routing Algorithm in Networks

- Forward ant and backward ant
- Forward ant searches (randomly) for destionation
- Backward ant travels back to source and grades path
- „Pheromone“ corresponds to measured link costs
- Future ants prefer routes with more pheronmone
- System adapts to breakdowns and overload situations



Analyzing a „perfect“ solution

- First build an omniscient component with perfect knowledge
  - E.g., a poker player that looks into the others‘ hand
- Learn the behavior of this solution, typically using statistical methods
- (Try to) reproduce the behavior for a realistic non-omniscient component

Example for learning from omniscient solution

- Work by de Meer et al. for iterated prisoneers dilemma
- Each player has two choices: cooperate/defect
- Reward matrix:

	Cooperate	Defect
Cooperate	3, 3	0, 5
Defect	5, 0	1, 1

- Perfect player knows what the oppenent will play and chooses the best option
- Perfect player has been statistically analysed
- Result: *Tit-for-Tat strategy with forgiveness*

Conclusion

- Self-organizing systems are interesting solutions for current and future networking problems
- Many possible applications, attractive properties (Robustness – Adaptability – Scalability)
- Bad news: no unique method that explains how to design a self-organizing system
- Good news: we already know several promising design methods
- Important to exchange ideas on the general principles of self-organizing networks and their application in communication networks

Thank you very much for your attention!

Summary by wordle.net

Alpen-Adria-Universität Klagenfurt
Institute for Embedded and Networked Systems
Mobile Systems Group

Wilfried Elmenreich