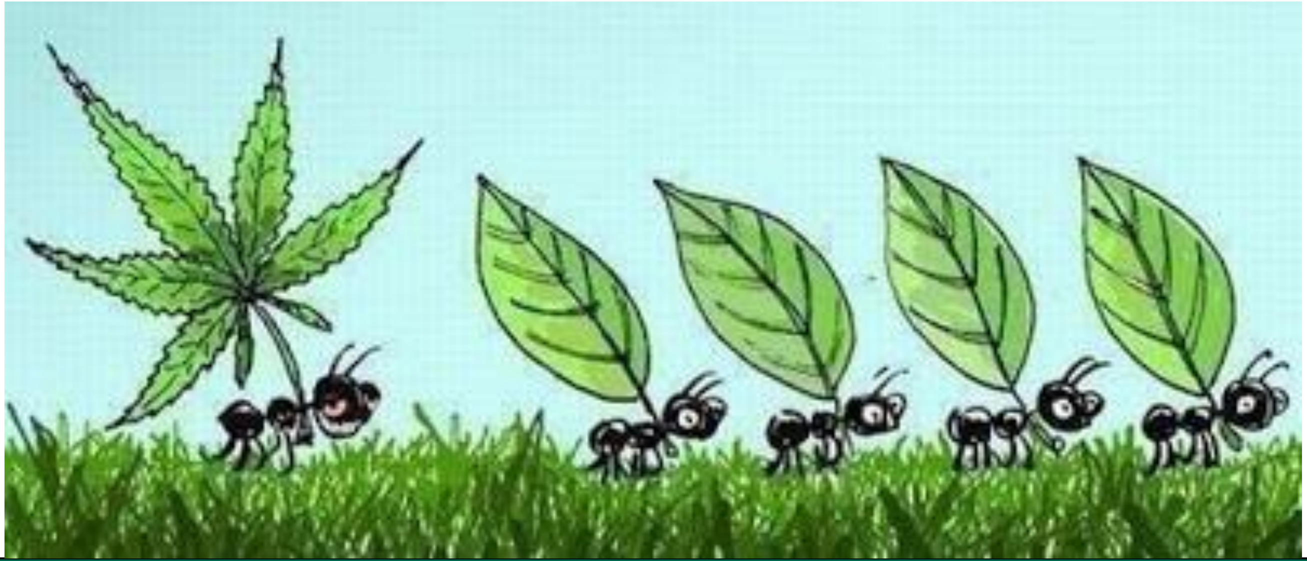
CSCE 990 Seminar:

Go to the Ant: Engineering Principles from Natural Multi-agent Systems



Mamur Hossain

02/04/2013

Citation of the Article

Parunak, H. V. D., 1997, "Go to the Ant: Engineering Principles from Natural Multi-agent Systems", *Annals of Operations Research*, 75 (1997) 69-101.

Outline

- Motivation
- Natural agent systems
- Engineering principles
- Agent design strategies
- Conclusion
- Praises
- Critiques
- Applications

Multiagent software systems can be designed using bio-inspiration

Traditional top down approach is complex and is not adaptable to changes.

Naturally occurring systems of agents are much simpler and such a system can adapt to changes.

Bio-inspired design is based on comparison, contrast and simplification.

The motivation behind a bio-inspired MAS is to have a robust system dealing with uncertainty

Top down approaches are applicable to predictable environments.

Uncertainty in a system requires higher flexibility that the top down approaches cannot offer.

Naturally occurring agent systems (such as insects) do not need system operator.

Overview of the Paper

Theory

- Agents
- Environment
- States & Processes
- Coupling

Natural Agent Systems

- Ants & Termites

- Wasps

- Birds & Fishes

- Wolves

Bio-inspiration

Engineering Principles & Agent Design Strategies

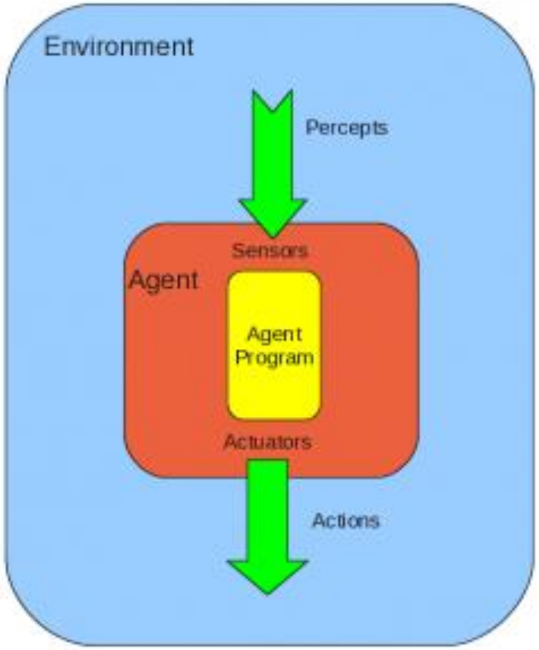
An MAS consists of some agents, an environment and coupling

MAS = <Agents, Environment, Coupling>

Agents = $\{Agent_1, Agent_2 \ldots Agent_n\}$
Agent = $\langle State_i, Input_i, Output_i, Process_i \rangle$

Environment = $\langle State_j, Process_j \rangle$

Models of State & Process
- Discrete-event
- Time-based / Continuous

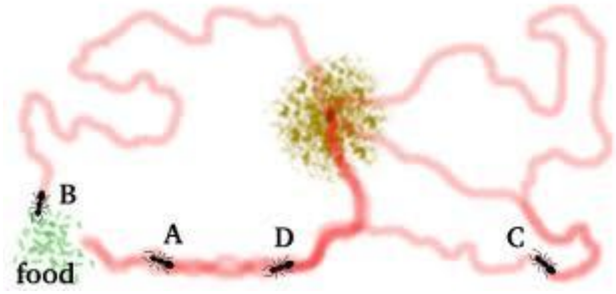


Coupling = {Discrete-event, Time-based} $\times$ {Agent, Environment}

Coupling
- Homodynamic
- Heterodynamic

Ants' path planning shows the emergence of local actions to a global outcome

System behavior: Networks of paths connecting the nest with food sources with minimum spanning tree network.



Responsibilities:

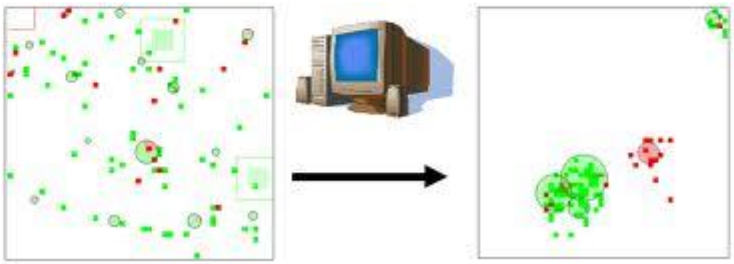
1. Avoid obstacles
2. Wander randomly – general direction towards pheromone
3. If holding food then drop pheromones at a constant rate
4. If not holding food and the ant is near the food then pick it up
5. If holding food and the ant is at the nest then drop the food

Integration: Resulting network of a minimal spanning tree.

Ant City Excavated

Ants' brood sorting explains how wandering ants examine all objects in the nest

System behavior: Sorting of larvae, eggs, cocoon and food in the nest w/o using any sorting algorithm.



Responsibilities:

1. Wander randomly around the nest
2. Sense nearby objects and maintain a short memory (10 steps)
3. If not carrying and encounters an object > decide stochastically whether or not to pick up. $p(pickup) = (k^+ / (k^+ + f))^2$
4. If carrying anything > decide stochastically whether or not to drop the object. $p(putdown) = (f / (k^- + f))^2$

   $k^- > k^+$ *; So that the clusters form faster than they dissolve*

Integration: Stochastic pick up and drop enables multiple concentrations of sorted elements to merge.

Tropical termites construct multi story nests w/o any centralized planning / management

System behavior: Termites make huge mounds that are complex, durable and effective. However, no termite serve the role of a chief engineer or planner.

Responsibilities:

1. Metabolize bodily waste that gives pheromone – this is the construction material for the mound
2. Wander randomly but prefer direction with strongest pheromone concentration
3. At each time step, decide stochastically whether or not to deposit the current load of waste

Integration: Probabilistic algorithm leads to the generation of:

Initial deposits > columns > arches > floors> complete mound

Wasps perform task differentiation in a decentralized manner

System behavior: Mature wasps in a nest divide into three groups: A single chief, a group of foragers and nurses for broods. These groups are not decided even by the chief!

Responsibilities:
Force parameter > How mobile a wasp is
Foraging threshold > How likely for the wasp to go seek food

1. When two wasps meet > engage in a face off and choose winner stochastically
2. When the brood receives food > reduce its demand
3. When a wasp is near the brood > determine stochastically whether or not to forage

Integration: High force, low threshold > foragers
Low force, low threshold > nurses
High force, high threshold (only one) > chief

Birds and fishes coordinate their movements locally leading to global dynamics of the flock

System behavior: Flocks of birds stay together, coordinate turns and avoid collisions with obstacles and each other. Schools of fishes exhibit similar coordinated behavior.

Responsibilities:
Each bird or fish maintain these simple rules -

1. Maintain a specified minimum separation from nearest object or other birds
2. Match velocity (magnitude and direction) with nearby birds
3. Stay close to the center of the flock

Integration: Individual bird or fish's behavior creates the global flock / school motion.

Wolves surround a prey w/o any long range communication – using local behaviors

System behavior: A single wolf cannot kill a moose, they coordinate with each other to surround it.

Responsibilities:
a predator-prey system with hexagonal grids where -

1. Moose: Move to the neighboring cell that is farthest away from the nearest wolf
2. Wolves: Move to the neighboring cell with the highest score: $S = d(moose) - k \times d(wolf)$
   where d(moose) is the distance to the moose
   d(wolf) is the distance to the nearest wolf
   k is a tuning constant i.e, repulsive force between two wolves

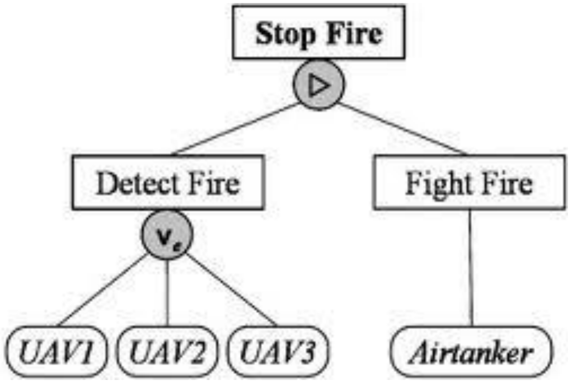
Integration: When repulsion an attraction are suitably balanced, wolves inevitable surround the moose.

Artificial systems can be engineered by mimicking natural systems

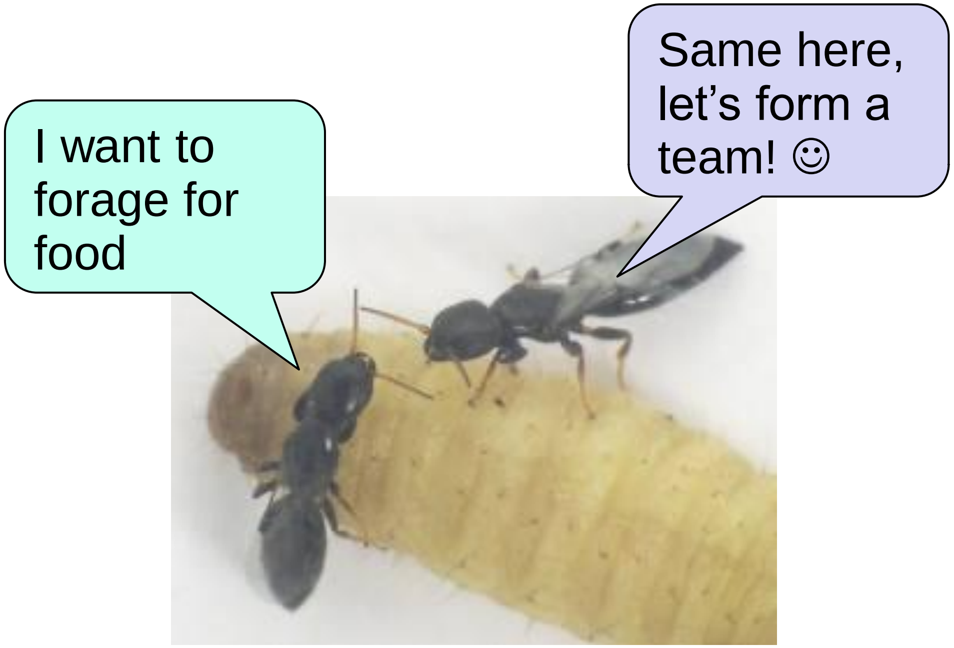
- Common principle of self-organization
- AARIA – a shop floor scheduling and control system
- CASCADE – a self routing material handling system
- CAS – a complex adaptive system similar to MAS
- Resnick on how people think about decentralized systems
- Kevin Kelly on bottom up control and chunking

Functional approaches are not well suited for naturally occurring distributed systems

Classical software engineering technique – functional decomposition

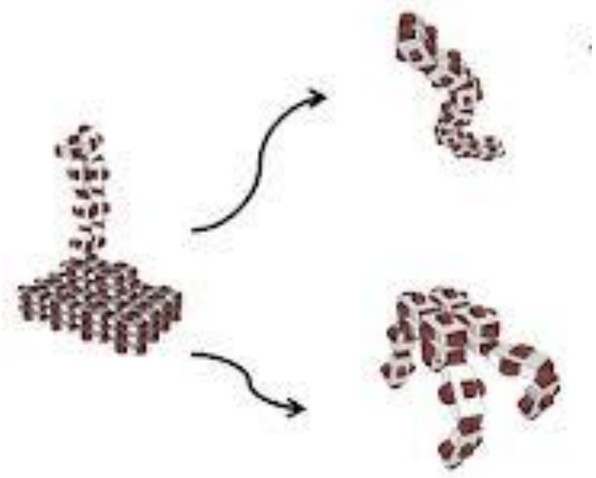


In natural systems functions are important but they emerge from individual components



Keeping agents small compared to the environment results in an adaptive system

Small in mass – negligible compared to the environment leading to a robust and wider scope for emergent behavior.



Small in time (forgetful) – smaller memory requirement and easy manipulation with the most recent data.

Small in scope (local sensing and action) – limiting recipients of messages to maintain local perspectives.

A decentralized system offers better robustness and flexibility

A central agent is vulnerable to total system failure

A central agent can work well in structured environments but it cannot expand beyond a boundary

Central systems become large software artifact that is difficult to understand and maintain

A population of diverse agents will provide better performance

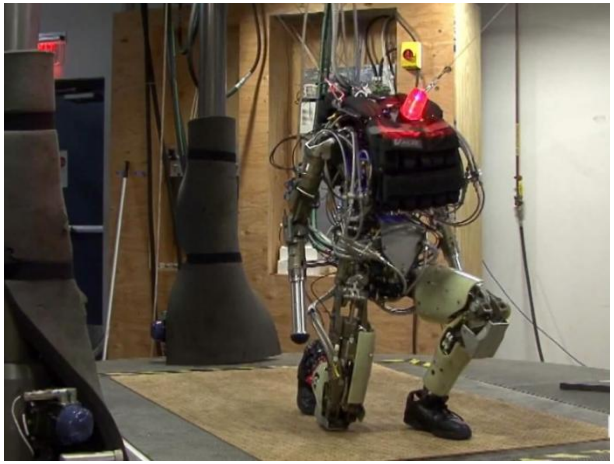
Ecological models show the importance of a diverse population

Diversity can be established using random process and repulsive fields

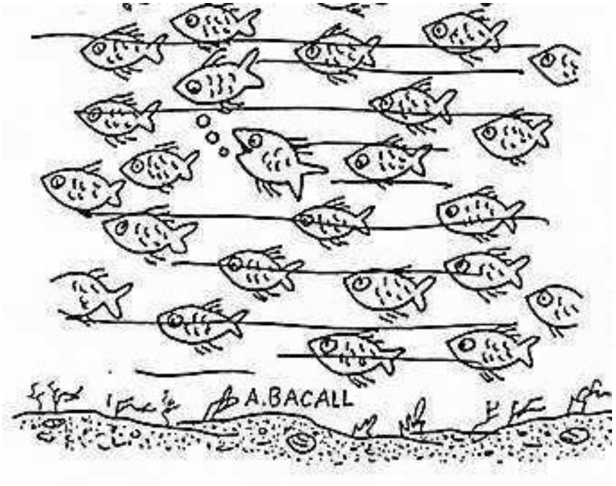
Example: diversity of location enables two fishes to be in two different places

Randomness and repulsion simplify an individual agent and add diversity

Randomized agents attack a problem in a Monte Carlo fashion – does not require an advanced model of the domain.



Wolf and bird examples show how a simple repulsive force among agents can maintain a diversity of location.



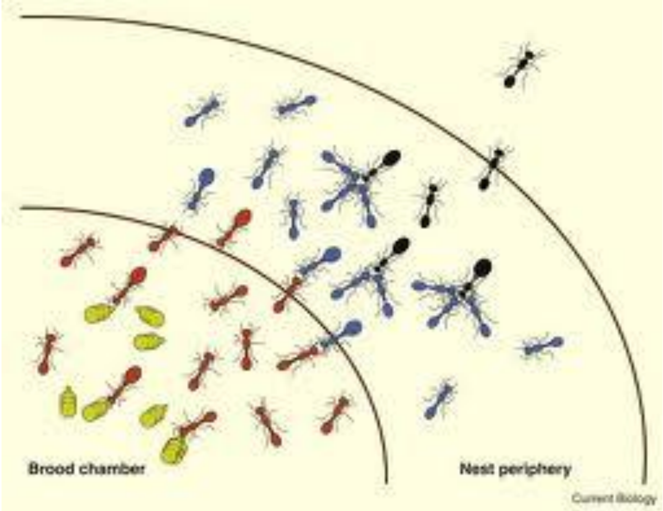
Risk and redundancy are useful in coping with an unstable environment

Natural systems work in an uncertain environment – need risk taking behavior and redundancy in the system

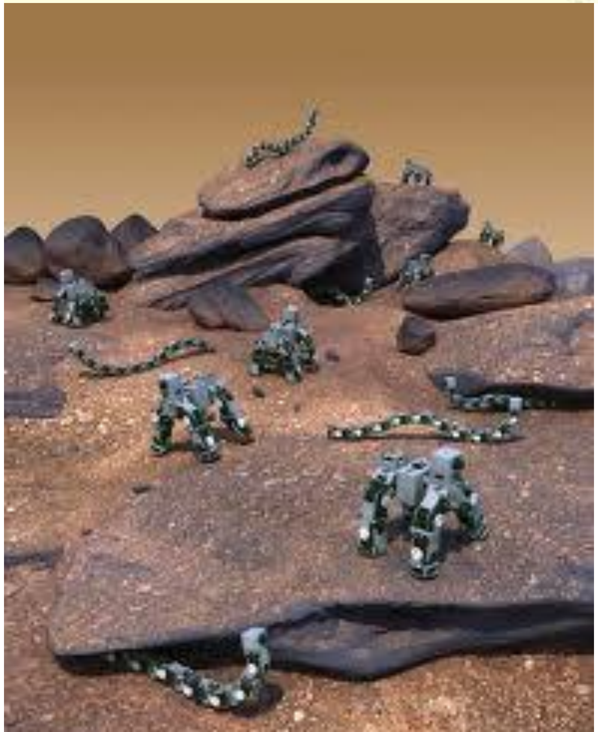
Risk taking behavior at local level is justified by the redundancy of the agents

A dissipative process at micro level ensures an organized macro behavior

Random movement of ants at micro level is a dissipative process / entropy leak

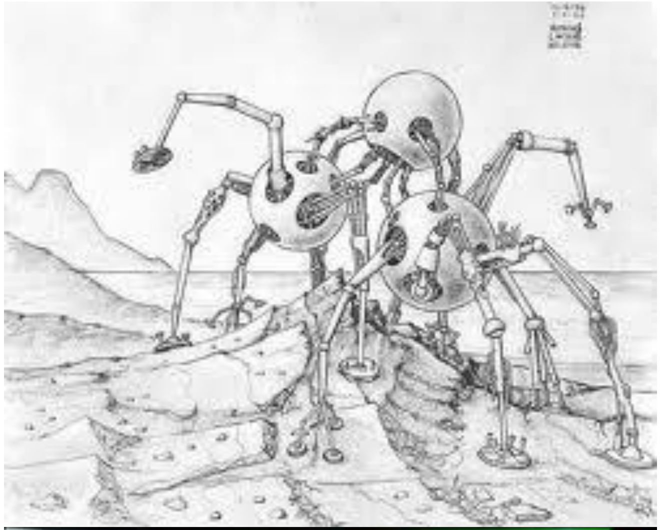


An artificial agent system can be benefitted by entropy leak – agents' actions must reinforce the field



Information sharing among the agents enable better response to changed conditions

Learning in a single agent will need sophisticated techniques

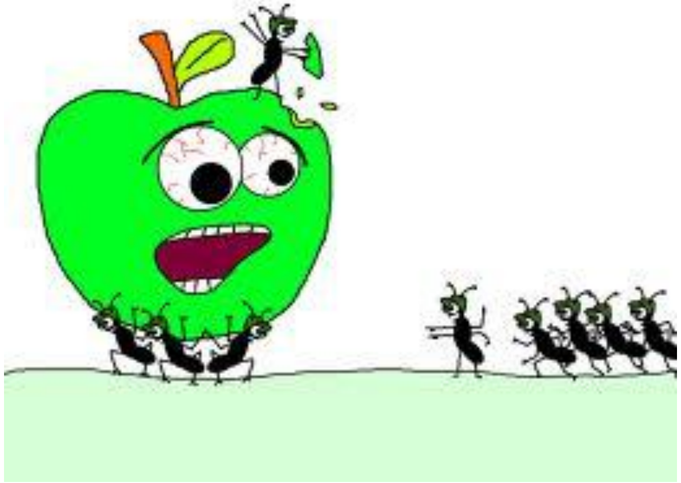


Information sharing such as evolutionary programming enables better response in case of changed environments

Plan and execution should be concurrent to cope with a changing environment

Traditional systems alternate planning and execution because of waiting for a central command.

Natural systems do not plan in advance and they do planning and execution concurrently.



Concurrent planning and execution results in faster response to changed conditions.

Ants Boat Amazon

Agent-based systems can be evaluated as useful in the cases of unstable environments

In a stable environment, centralized systems outperform agent based systems.

Unstable environments and uncertainty need randomness, redundancy and concurrent planning and execution that an MAS has.

Conclusion and Recent Related Works

- Centralized systems are vulnerable to changes in environment.
- Naturally occurred MAS such as ants, termites, birds, fishes etc. have the following properties that can be mimicked to an MAS software –
  - Redundant
  - Decentralized
  - Risk taking
  - Small agents
  - Dissipative local actions
  - Efficient global actions

- Evolutionary Robotics: Cornell
- Biorobotics Lab: CMU
- CSAIL: MIT

Discussion

- Praises in favor of bio-mimicking in MAS
- Critiques against the approach
- Applications of the nature-inspired MAS to the class project

Praises

- The paper introduces sufficiently detailed methods to engineer artificial systems following bio-inspiration
- Natural systems with multiple agents were described thoroughly along with multiple examples
- Natural systems were explained in a way so that it can be represented in a software system
- The paper discussed applications of a decentralized software system to shop floor and manufacturing environment
- The properties of a bio-inspired software system were explained in great details

Critiques

- The paper did not discuss related works to sufficient extent
- Although a natural agent is termed to as simple, their hardware is not as simple as the software – example: body of an ant
- Implementation of an MAS is challenging – specially because of tolerance, efficiency of individual agent etc.
- The paper could exemplify practical applications of nature-inspired MAS to a greater details

Summary

Functional decomposition	Vs	Correspond to problem domain
Large sized agents	Vs	Small sized agents
Long term memory agents	Vs	Short term memory agents
Long range sensing / action	Vs	Short range sensing / action
Homogeneous / incompatible	Vs	Diverse agents
Accurate at local level	Vs	Dissipative at local level
Sequential planning/execution	Vs	Concurrent planning/execution
Centralized software system		**Decentralized nature inspired MAS software**

Applications on Class Project

- Class project: *Multiagent Cooperative Payload Transportation with Modular Self-reconfigurable Robots*
- Individual robot agents should be small, with short memory and with short-range sensor and actuators
- Robot agents will stay close to the centroid of the mass (load)
- Robot agents will coordinate their velocity (magnitude and direction) with the nearest other agents
- Number of agents will be redundant in case the load carrying robots need help in case of failure of one of them

ModRED Gaits

Q / A and References

[Resnick 1994] M.Resnick, *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds.* MIT Press.

[Kelly 1994] K.Kelly, *Out of Control: The Rise of Neo-Biological Civilization.* Addison-Wesley.