Cyber Biosphere (CBS) for Future Embedded Systems

Franz J. Rammig
Heinz Nixdorf Institute
University of Paderborn, Germany
franz@upb.de

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

What Cyber Biosphere is not about

Source: Hartmut Schmeck

Outline

- Motivation
- Lessions learned from biology
- Examples
  - Ant colony algorithms
  - Artificial hormone systems
  - Artificial immune systems
- Towards creating a Cyber Biosphere
  - Basic principles of CBS
  - CPS vs. CBS
  - CPS and CBS in synergy
- Conclusion

Worldwide Intrusion by Interlinked IT

- Over 20 billion processors
- A billion computers
- A billion of internet access points
- Billions of kilobytes moved per second
- 1.5 million new domains per day
- A billion of potential nodes in worldwide self-coordinating software development

- 100 million production sites
- 20 million containers
- Billions of temporary trade connections
- Millions of potential nodes in worldwide self-coordinating logistics networks

- Half a billion vehicles
- A trillion of kilometers of traffic per year
- Billions of decisions per second
- Billions of potential nodes in self-coordinating driver assistance systems

Lessons Learned from Nature ??

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

The classical approach of engineering:

- Build highly efficient systems
- Build highly reliable systems
- Build highly deterministic systems

Typical solutions:

- Deterministic real-time scheduling
- Schedulability analysis
- Collision-free communication protocols
- Time-triggered architectures
- Formal proof techniques
- …

- **Biosphere seems to be completely different**
- **Follows completely different principles**
- **Getting inspirations from biosphere seems to be a strange idea**

Lessons Learned from Nature !!

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Biological systems are highly robust

- Human body consists of billions of cells
- Is exposed to billions of enemies (antigens)
- Attacking strategy is changed continuously and in unpredictable manner
- MTBF analysis would conclude a human's lifetime of not more than some hours
- In reality its up to 100 years (or more)
- Same holds for any complex bio-system

Biological systems have proven to be

- extremely robust in a hostile environment
- Even when confronted with rapidly and unpredictably changing attacking strategies

© Heinz Nixdorf Institute, University of Paderborn

Lessons Learned from Nature ??

Of course, the classical approach enables as well extremely robust systems:

- Today's SoC with billions of transistors
- Telephone switching systems
- But what about reaction on changing environments?

Of course the robustness of biological systems is limited as well:

- Beyond a certain limit of flexibility the respective species just disappears
- The corridor of homeostasis, however, seems to be broader compared to classical technical artifacts

Not surprising: One of the

- most robust
- most stable
- most adaptive

technical artifact is the Internet

Self-Organization/Self-Coordination as Key Property of Biological Systems

Common to all self-coordinating systems:

- **Decentralization**
  - decentralized control and components acting autonomously
- **Volatility**
  - Network of rapidly changing structure, communication behavior, and component behavior
- **Scoped knowledge**
  - Unobservable global system state and thus components with only local knowledge

- **Selfishness**
  - Optimization of own benefits being the driving force of a component's cooperation or potential competition
- **Adaptability**
  - Adapting to and learning from environmental changes as a universal ability of components

- **Dependability**
  - Limited availability of resources together with required security and safety of the highest degree being the hard **constraints**

Ant Colony Algorithms

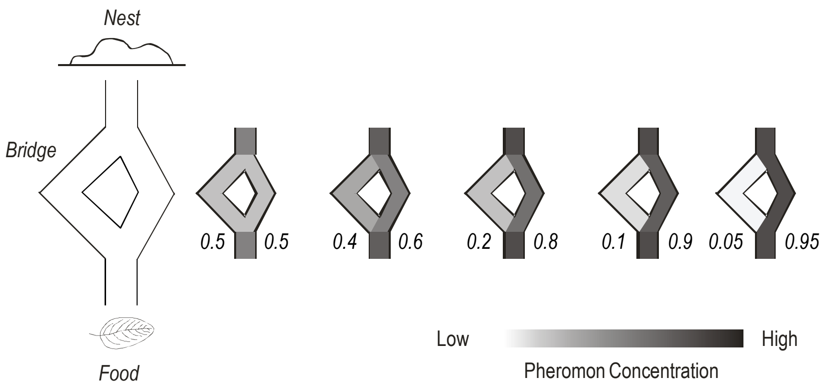
Something general about ACO

- Introduced by Marco Dorigo in 1992
- Universal meta-heuristic for optimization problems
- In comparison to other meta-heuristics shows robustness against changing environments

Basic principles:

- Based on random search of „ants“ for food
- Communication via *Stigmergy*
- Ants leave *Pheromone* when moving
- Prefer following paths marked with pheromone
- Pheromone evaporates over time

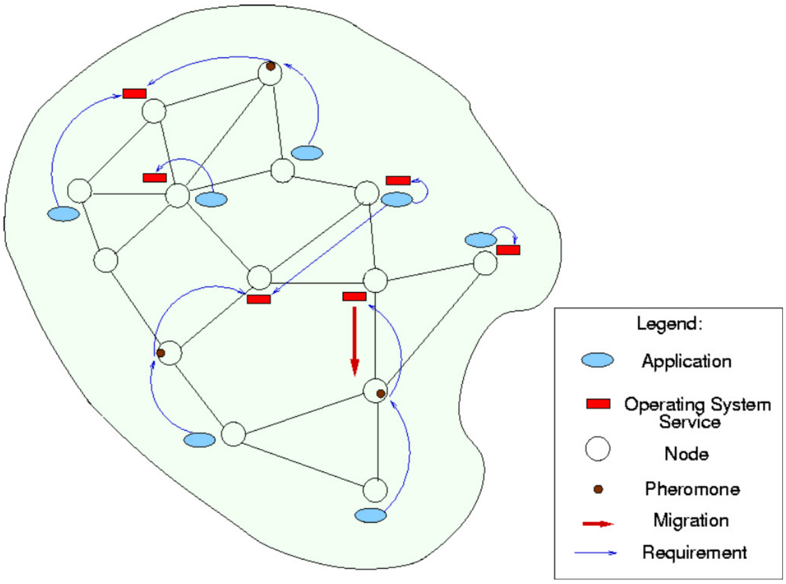
Ant Colony Routing



§ **Pheromon estimation und adaptation based on transport cost function based on connection ratings**

Application of ACO: Service Distribution in a Distributed OS for WSN

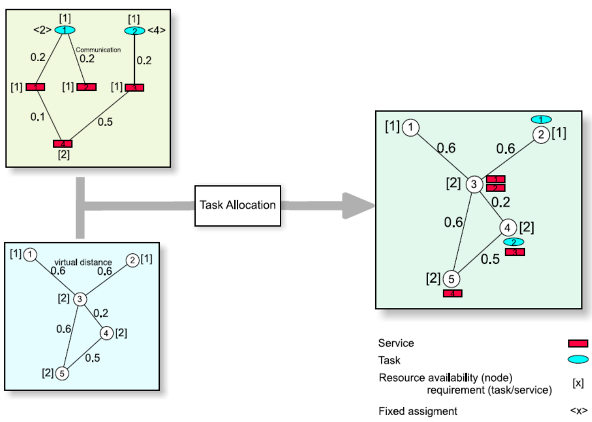
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Problem Definition



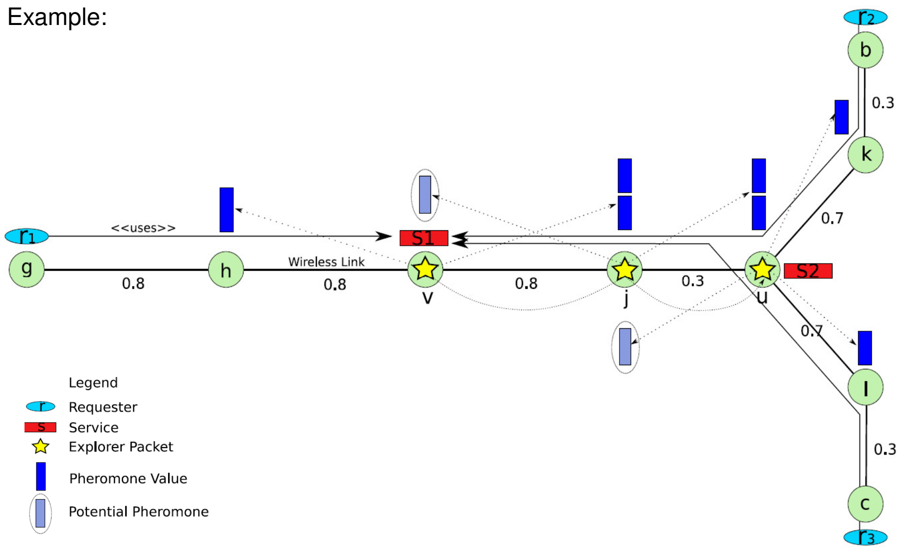
$$\text{cost} = \sum_{\{m_1,m_2\}\in C} b(\{m_1,m_2\})\cdot D(q(m_1),q(m_2))$$

Ant Based Service Distribution

- **Using an analogy with the ant foraging behaviour:**
  - In our approach services are the equivalent of food sources
  - Service locations are the equivalent of shortest paths
  - Calls made by the requesters are the ants
  - Requesters are the nests
  - Wireless links form the paths which the ants can use for movement
  - While the requests are being routed to the destination service, they leave pheromone on the nodes.
  - Pheromone evaporates over time

Basic Ant Based Service Distribution

Example:



Artificial Hormone Systems

Something general about Artificial Hormone Systems

- Early papers by Trumler et al. at BICC 06, Oberthür et al. at BICC06, Brinkschulte et al. at ISORC08
- Models the broadcasting technique used in the hormone system of biospecies
- Application to distributed computing systems

Basic principles:

- Use destination-independent broadcasting
- Distribute just messages (data, facts)
- Received by many (all) members of a population.
- Each receiver decides individually whether it feels to be addressed and how to react (Publish/Subscribe principle)
- Strictly delegating responsibility how to react

Fight or Flight

How did humans survive in "dangerous times"?

Fight or flight response = acute stress response

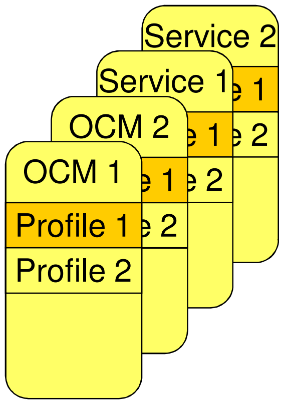
Reaction
- Clever resource management

RTOS
- How can this "technique" be useful in an RTOS?

Self-optimizing RTOS
Profile Framework

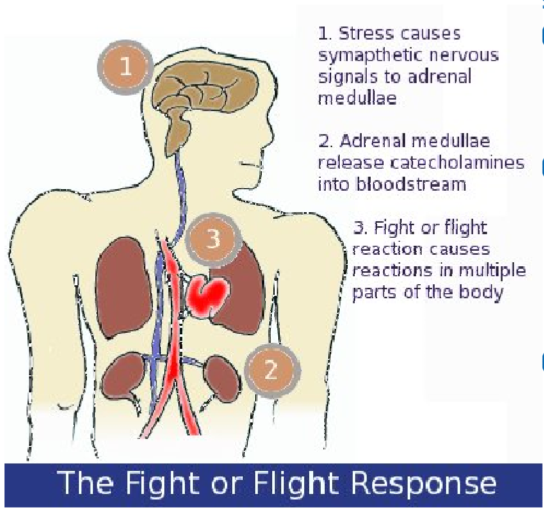
§ **Application & system services**
- Defines profiles, comparable to:
  - § Different execution levels / service levels
  - § Every profile fulfills the minimal required task, (eventuality) with a different quality

§ **Used by**
- Applications
  - § Model implementation alternatives with different resource requirements
- System services
  - § Model a service status per profile:
    - activated/degraded/deactivated



OCM: specific application modules
(Operator Controller Modules)

Acute stress response
How is it "activated"?



§ **Transfer to our scenario**

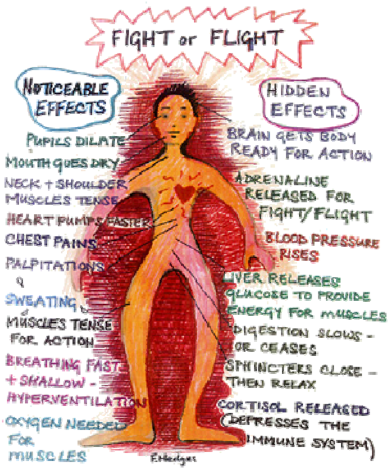
(1) A "danger" is detected
- by application or monitoring system

(2) Virtual adrenalin is released
- Distributed to the applications
- Carries additional information about the threat

(3) Fight of flight response
- Reaction in multiple parts of the virtual organism (mechatronic system)

Acute stress response

What happens to deal with the threat?



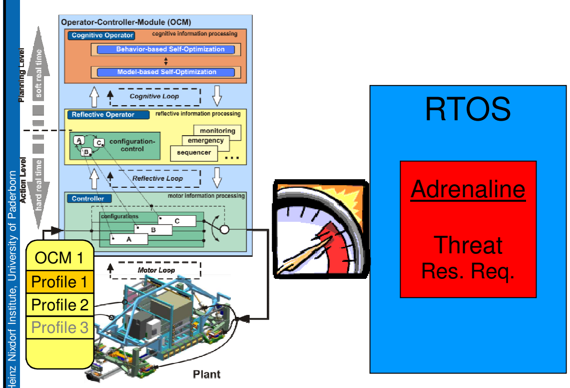
§ **What is the goal in our scenario?**

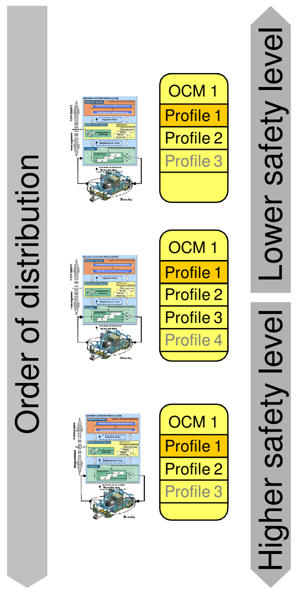
- **Components, which take steps against the threat, get more resources**
  - Special profiles with higher resource consumption
- **Other components try to free resources but remain functional on "low level"**
  - Special profiles with low resource requirements

Acute stress response

Our heuristic

- **Distribute adrenalin from lower safety level to higher safety level**
  - § **OCM decides whether it has a "low-level" profile to "offer"**
- **Update adrenaline**





Artificial Immune Systems

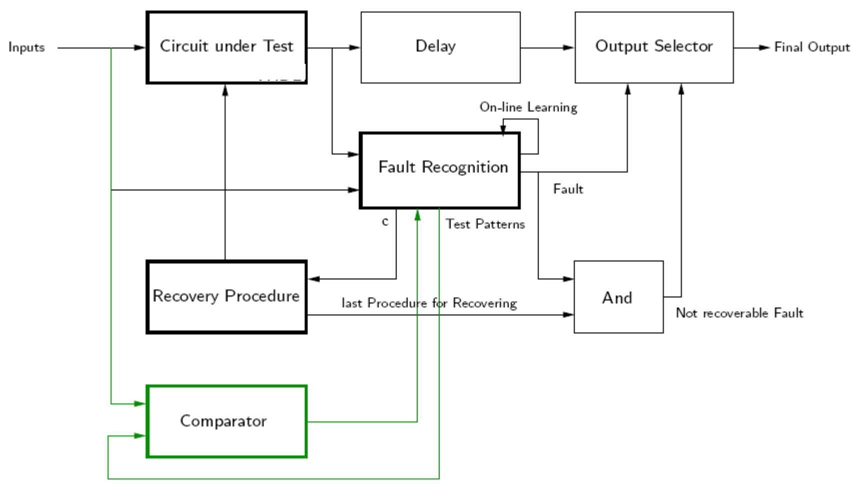
Something general about Artificial Immune Systems

- First approaches back in the 80's
- Universal meta-heuristic for optimization and intrusion detection problems
- Immune network theory as meta-heuristic for learning societies

Basic principles:

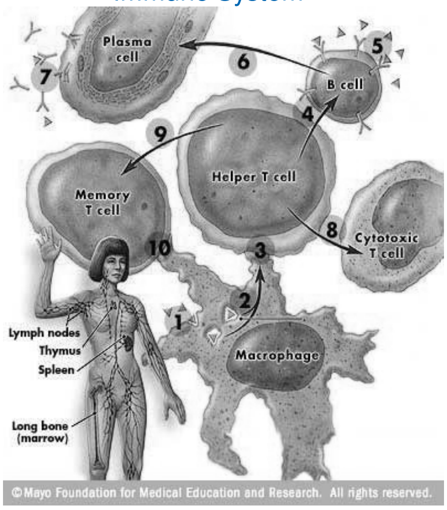
- Pattern recognition by modeling antigenes and antibodies and a proper distance measure
- Either abstract models of the biological immune system or using detailed models
- Hypermutation (mutation only inside immune cells) for adaptation to previously unknown antigenes
- Improvement of protection capabilities by building immune networks

Towards fault-immune FPGAs



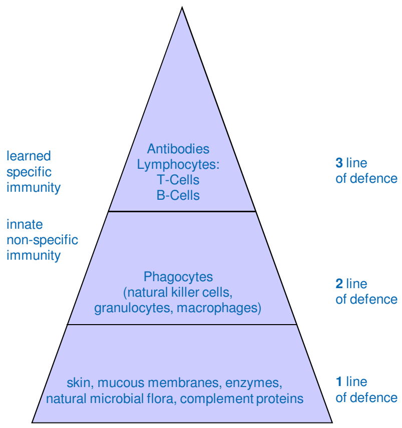
- **Immune System**
  - Tissues
  - Organs
  - Network of cells
    - Lymphocytes
      - T Cells
        - T Helper Cells
        - T Killer Cells
      - B Cells
    - Paghocytes

Immune System



- **Immune System**
  - Tissues
  - Organs
  - Network of cells
    - Lymphocytes
      - T Cells
        - T Helper Cells
        - T Killer Cells
      - B Cells
    - Phaghocytes

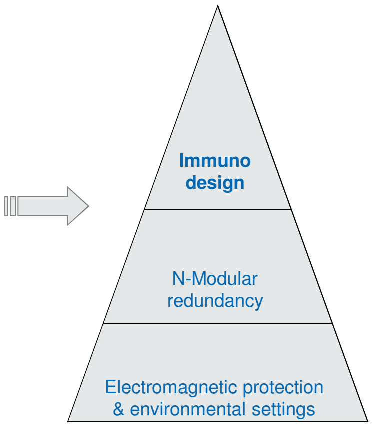
Immune System



Artificial Immune System

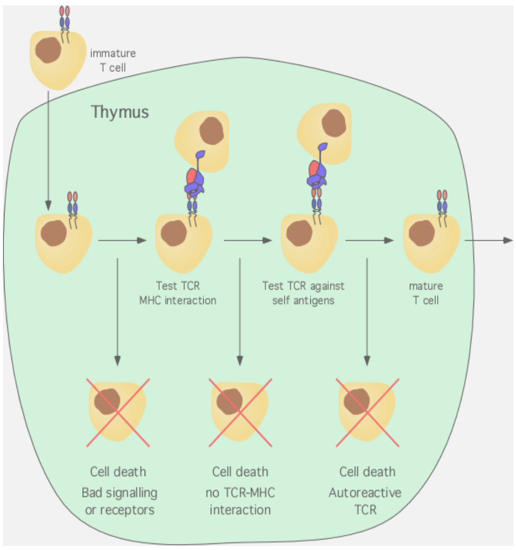
- **Immune System**
  - Tissues
  - Organs
  - Network of cells
    - Lymphocytes
      - T Cells
        - T Helper Cells
        - T Killer Cells
      - B Cells
    - Paghocytes

Artificial Immune System



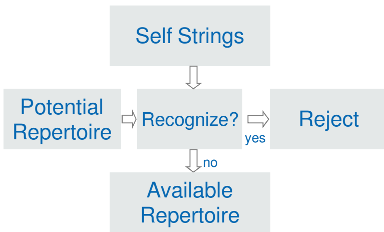
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

T Cell Maturation & "Self non self" algorithm

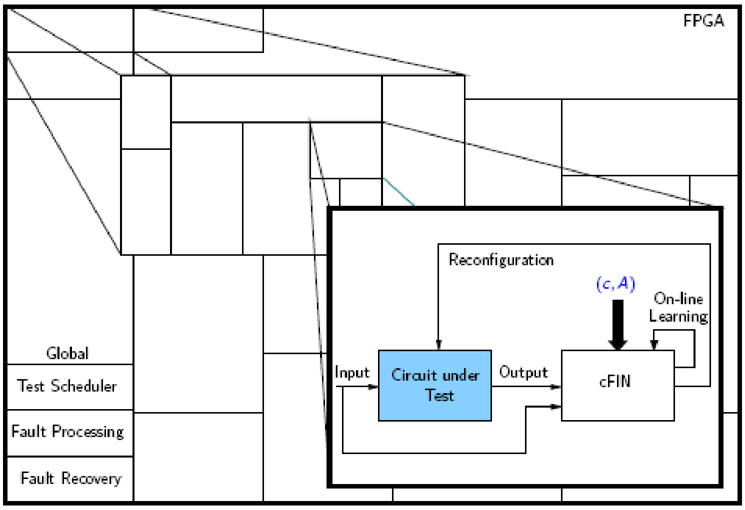


www.bact.wisc.edu

Negative selection



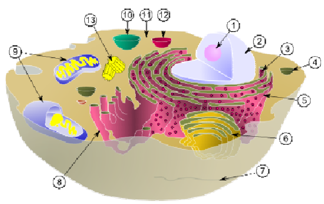
Partitioned circuit



Conceptual Inspirations from Nature (1)

Principle 1:
Follow a cell-based approach

- Nature invented life by inventing cells
- For 2.75 billion years life did exist only by means of single cell entities
- Whatever species or colony of species we are looking at, they always are built bottom-up as a collection of cells
- Cells include
  - intelligent I/O (cell membrane)
  - static code (nucleus)
  - reproduction mechanism
  - complete chemical plant (cell plasma)
  - energy management
  - motion mechanism
- I.e. cells are self-contained with
  - information
  - energy
  - material flow



Source: Wikipedia

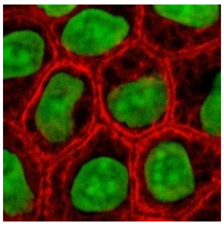
Conceptual Inspirations from Nature (2)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Principle 2:
Follow a federation approach

- Higher species are made by (large) collections of cells that may cooperate very closely
- Cells may be differentiated into highly specialized ones
- Cells never lost their autonomy, each cell keeps
  - intelligent I/O (cell membrane)
  - static code (nucleus)
  - reproduction mechanism
  - complete chemical plant (cell plasma)
  - energy management
  - motion mechanism
- I.e. biological systems are federated ones

- Same concept is maintained in social insects ("macro cells") and even higher up to human societies.



Source: Wikipedia

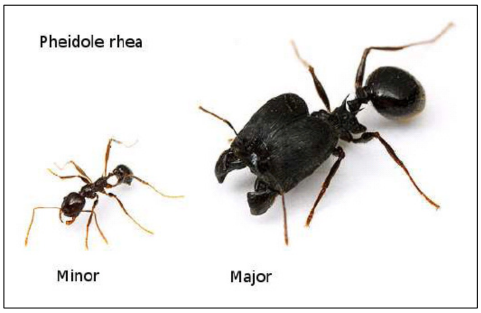
Source: Ant Hill Wood

Conceptual Inspirations from Nature (3)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Principle 3:
Elasticity

- Higher species are made by differentiated, highly specialized cells
- But there stays some degree of elasticity:
  - components dedicated to specific tasks can take over other tasks up to a certain degree
  - extremely valuable principle to achieve robustness
- Good compromise between
  - efficiency (division of labor) and
  - avoidance of single point of failure.



Conceptual Inspirations from Nature (4)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Principle 4:
Broad variation of communication techniques

- **Nature invented the entire bandwidth of communication**
  - **Broadcasting:**
    - Hormones
    - Cytokines
    - Pheromones
  - **Multicast/Unicast:**
    - Nerve system
  - **Wired:**
    - Nerve System
  - **Wireless**:
    - Stigmergy
  - **Power line communication**
    - Hormones

All these technique are based on basic capabilities of cells.

Common to all techniques is the principle of delegation.

Communication takes place by sending messages.

It is up to the receivers how to react (Publish/Subscribe)

Conceptual Inspirations from Nature (5)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Principle 5:
Delegation

- Large complex systems need a high degree of self-organization or even self-coordination
- Pre-planned communication seems to be no longer adequate in such a context.
- The principle of delegation reduces dramatically the amount of information to be communicated.
- Having intelligent receivers allows to send just data.
- The receivers can decide, whether and how to react.

© Heinz Nixdorf Institute, University of Paderborn

Cyber Physical Systems (CPS) vs. Cyber Biosphere (CBS)

Major principles of CPS:

- Build systems where the correctness can be guaranteed
- Made of potentially unreliable components
- Exposed to an unreliable environment
- Compensate unpredictability at a certain layer by robustness at the next higher layer
- Don't hide away essential properties by abstraction

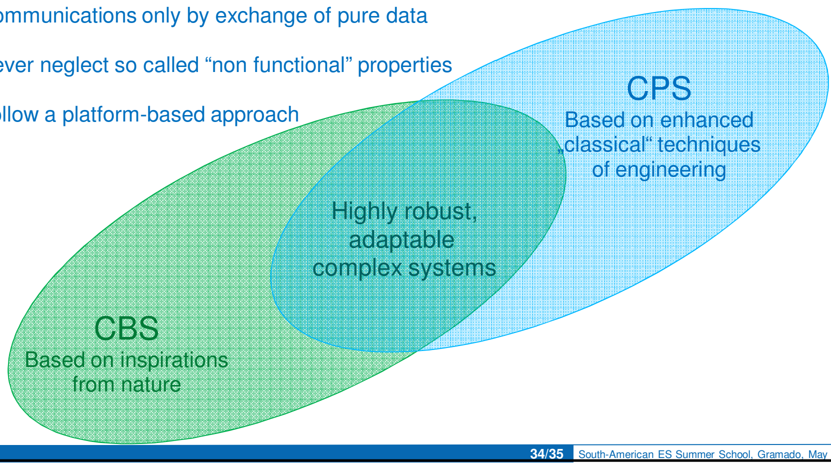
Major principles of CBS:

- Obtain robustness by continuous adaptation
- Compile systems out of semi-autonomous components
- Provide a high degree of local elasticity
- Delegate responsibility
- Federation instead of hierarchy

Cyber Physical Systems (CPS) meets Cyber Biosphere (CBS)

Common principles of CPS and CBS:

- Build systems that can act in a robust manner in unexpected situations
- Build systems following a strict component-oriented approach
- Communications only by exchange of pure data
- Never neglect so called "non functional" properties
- Follow a platform-based approach



"Principles encountered in the nature can be transferred to computers with satisfactory results"

Thank you for your attention.

Heinz Nixdorf Institute
University of Paderborn
Fuerstenallee 11
33102 Paderborn
Germany

Phone: +49 (0) 52 51/60 65 01
Fax: +49 (0) 52 51/62 65 02
Email: franz@upb.de
http://www.heinz-nixdorf-institut.de

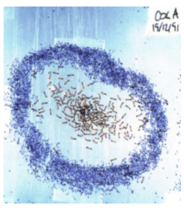
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Fundamental Design Principle of Biologically Inspired Systems: Self-Coordination

Franz J. Rammig

Self-organisation in biological systems

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Leptothorax ant nest between panes of glass

Wing patterns of rice paper butterfly

Leaf cutter ants in fungus garden

Introduction

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- 20 billions of microprocessors
- Give arbitrary technical artefacts the potential to become a *„Thing that Thinks“*
- All these *Things that Think* may be interconnected, forming the *„Internet of Things“* ➡ *Cyber Physical Systems*
- A *„Virtual Organism“* is coming into existance
  - Creating an *„Ambient Intelligence“*
  - Omnipresent, *„Ubiquitous Computing“*

  ➡ *Cyber Biospere*
- This virtual organism cannot be governed by any traditional means
- We need completely novel
  - Design
  - Evolution, and
  - Management

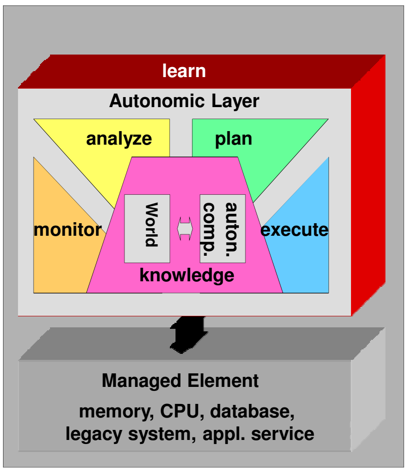
  means

International Research efforts (Selection)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

- Software Engineering Institute (CMU)
  Ultra-Large-Scale Systems
- European Union, IST Programme:
  Dynamically evolving, large-scale information systems (DELIS)
- DGF (German Science Foundation):
  Organic Computing
- IBM et al.:
  Autonomic Computing
- UC Berkeley et al.:
  Cyber Physical Systems



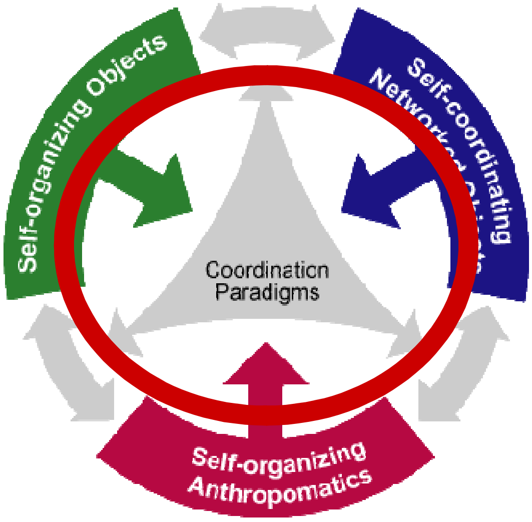
Source: IBM

Structuring Reseach

- We need to understand the virtual organism to design, ie. we have to create a new kind of „System Theory for Self-coordinating Systems“
  - Models and modelling
  - New system structures
  - New algorithmic foundations
- We need to understand self-coordinating Objects
- We need to understand self-coordinating networking of self-coordinating objects
- We have to serve human beings, i.e. we have to understand self-coordinating *anthropomatics*

Structuring Research

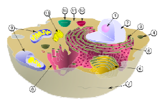
Structuring Research



Central Aspect of Coordination Paradigms: Models and Modeling Techniques

New challenges in modelling:

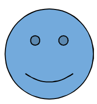
- Communication-centric instead of object-centric
- Highly concurrent systems
- Loosely coupled systems
- Self-adaptation of objects
- Self-adaptation of communication infrastructure



Source: Wikipedia

Approach:

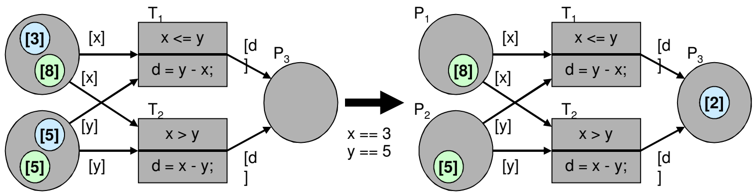
- Basic principle: Petri nets
  - Communication-centric
  - Highly concurrent
  - Loosely coupled



- **Extensions:**
  - Information processing: Pr/T-nets
  - Timing: Own extensions
  - Self-adaptation: Own extensions
  - Standardization: UML Activity Diagrams

Predicate/Transition Nets

- **Basic model for behaviour: Pr/T nets**
  - Property "distributed system" is central one
  - Good modelling means for self-modification
  - Completely distributed control
  - Locality

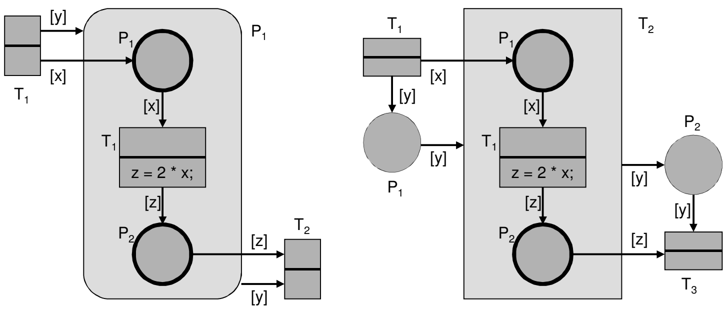


HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Hierarchy in Pr/T-nets

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

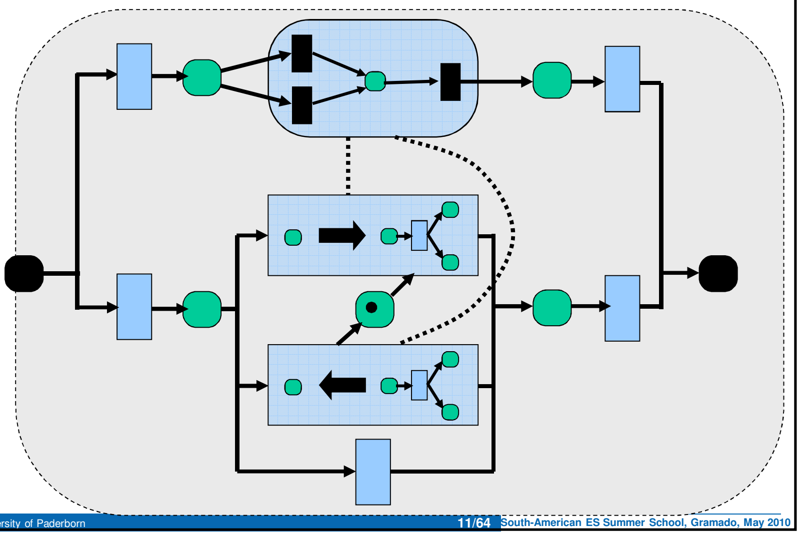
- Hierarchical places
- Hierarchical transitions
- Semantics considers activity of sub-nets



Dynamic Modification
(with Carsten Rust)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Idea: Dynamic Modifications bound to Transitions

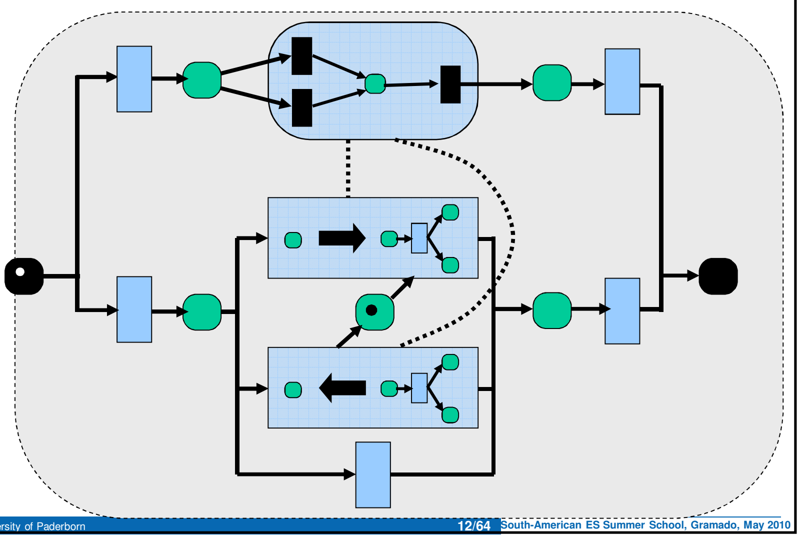


Dynamic Modification

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

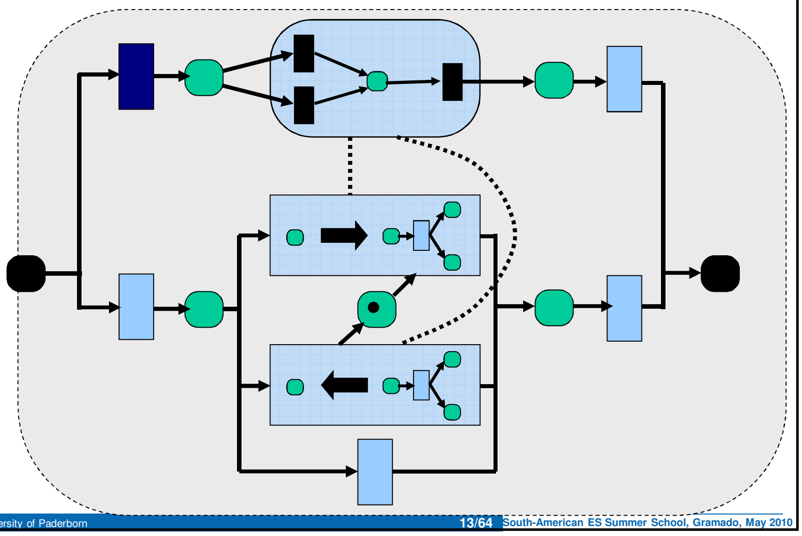
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

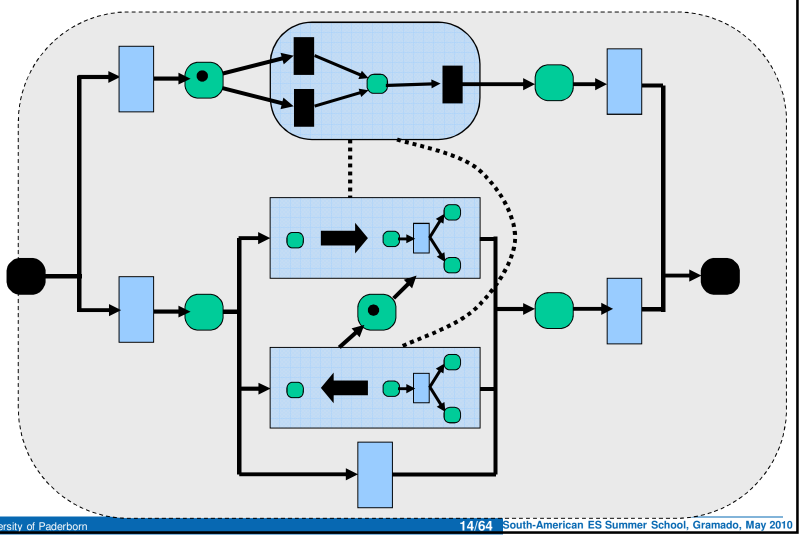
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

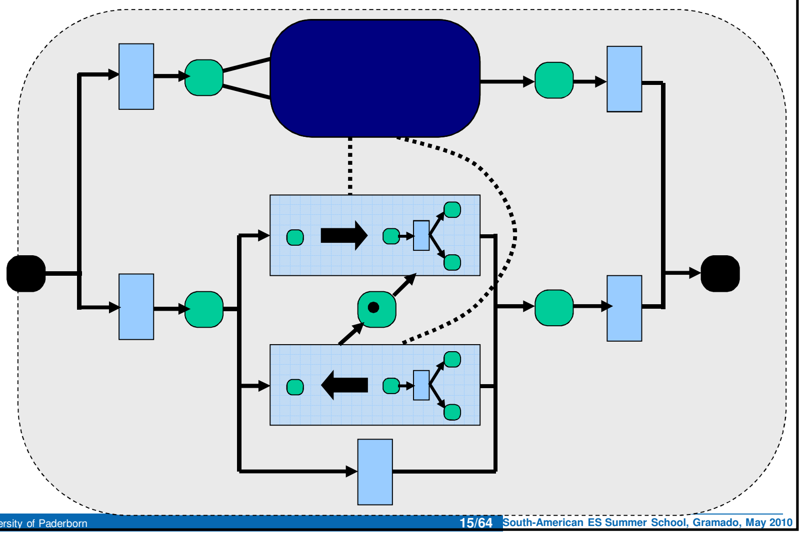
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

- Idea: Dynamic Modifications bound to Transitions

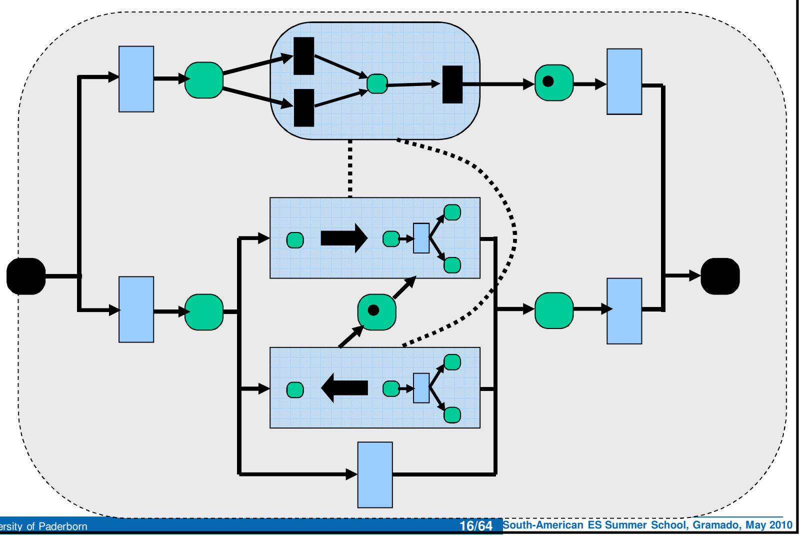


HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Dynamic Modification

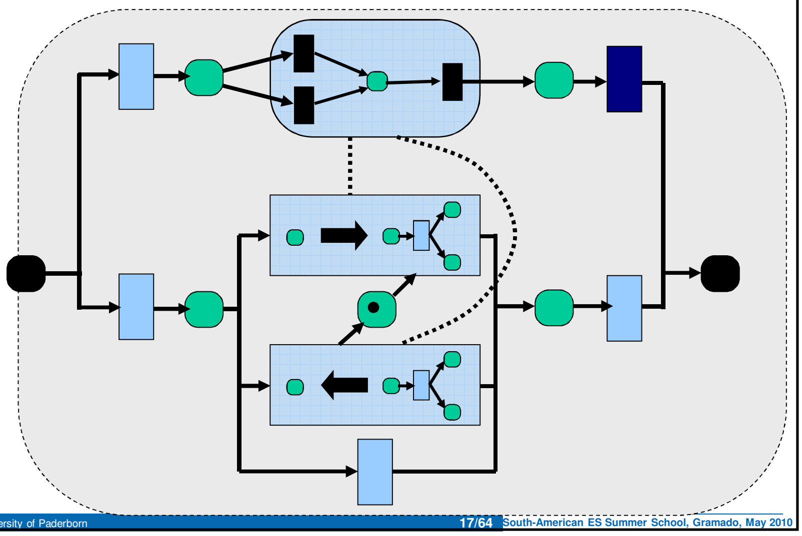
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

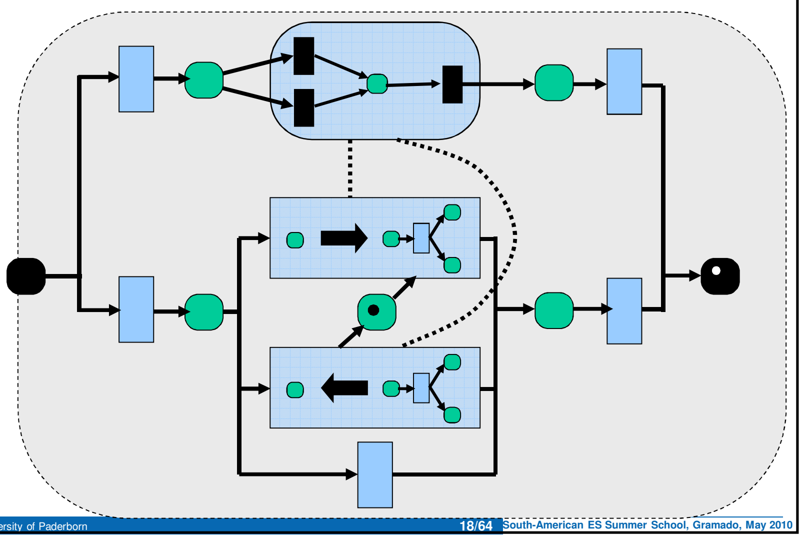
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

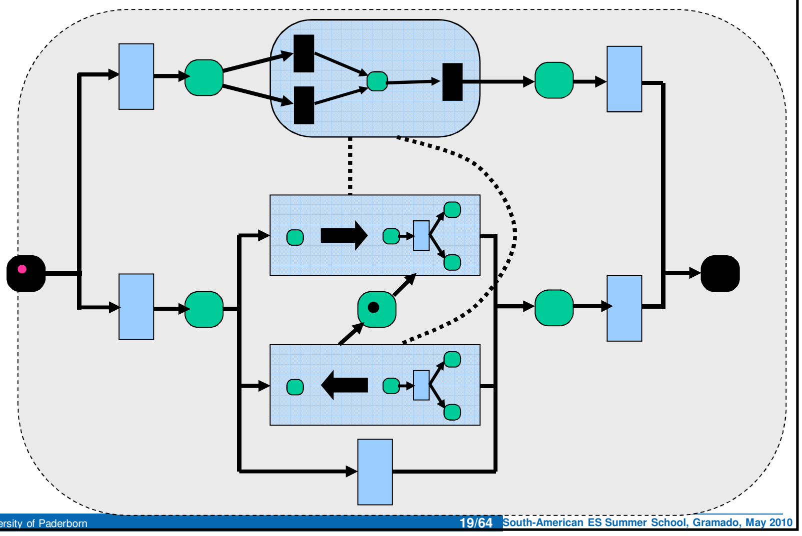
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Idea: Dynamic Modifications bound to Transitions



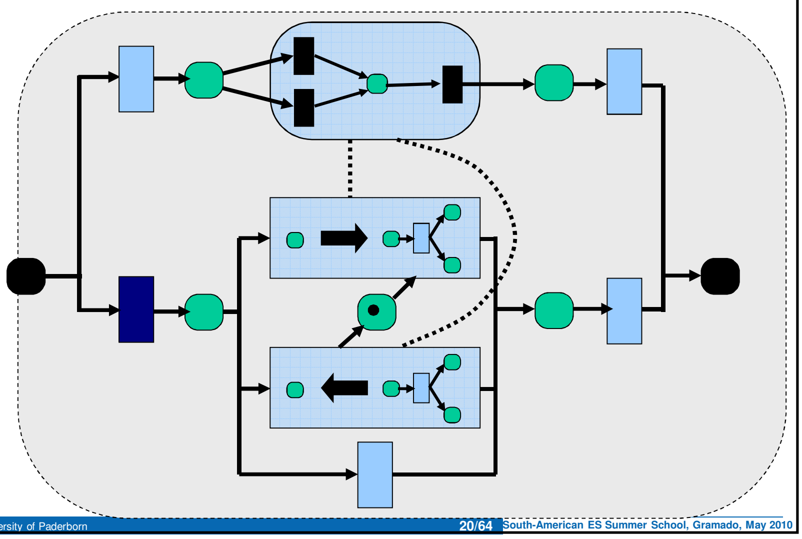
Dynamic Modification

- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

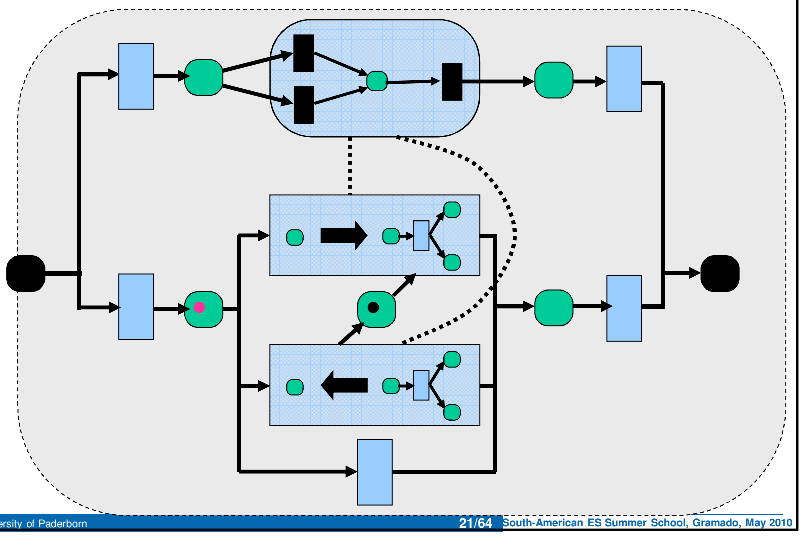
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Idea: Dynamic Modifications bound to Transitions

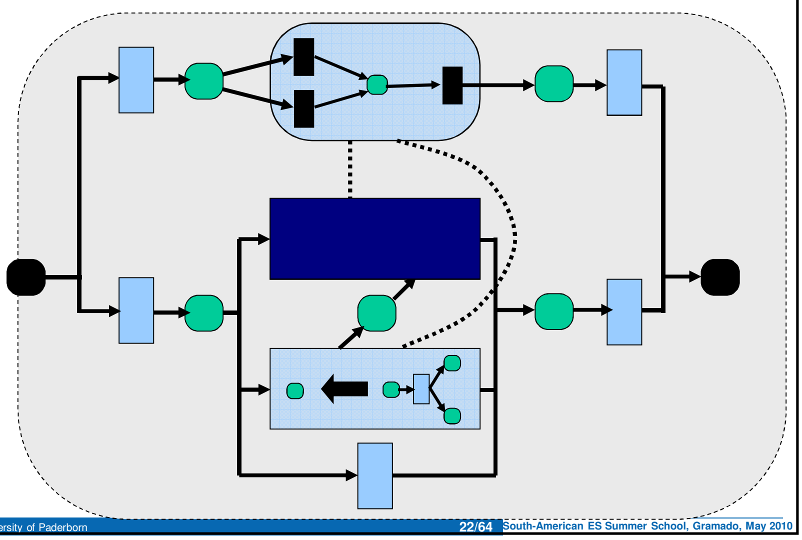


Dynamic Modification

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

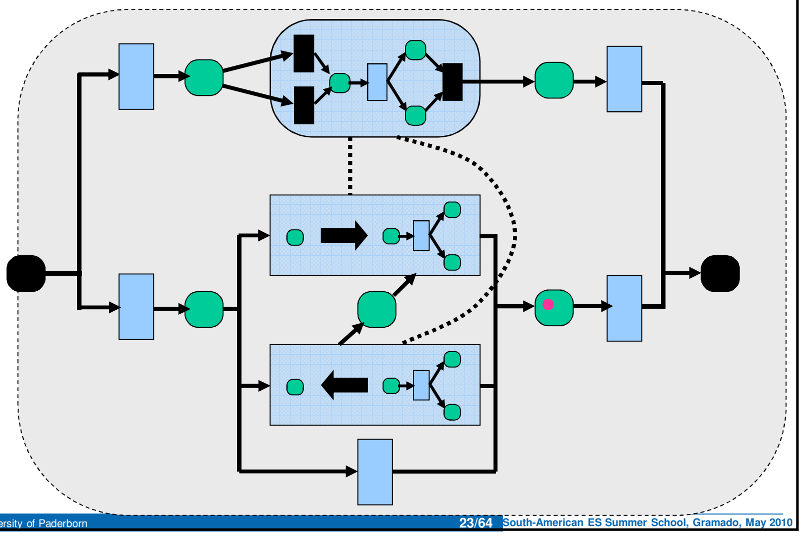
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

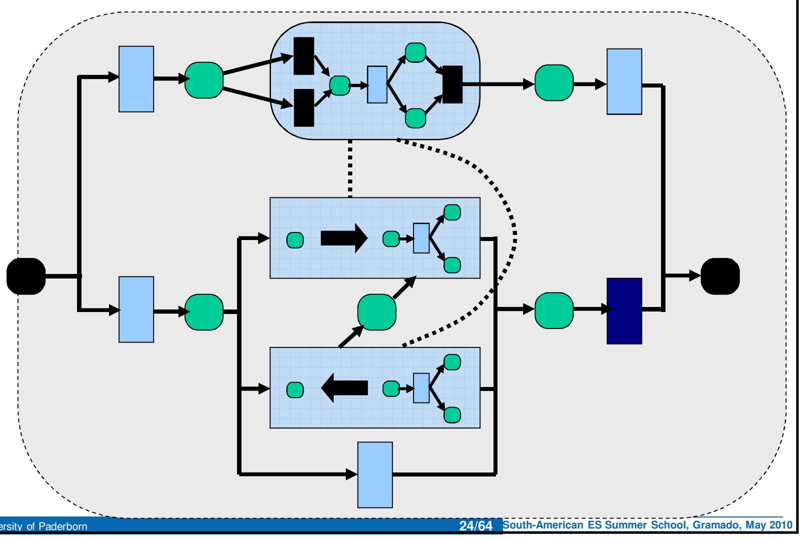
- Idea: Dynamic Modifications bound to Transitions



Dynamic Modification

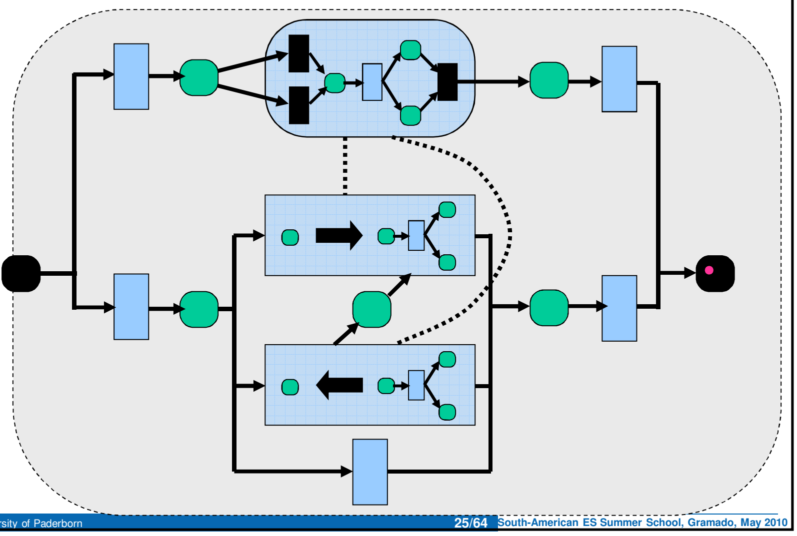
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Idea: Dynamic Modifications bound to Transitions

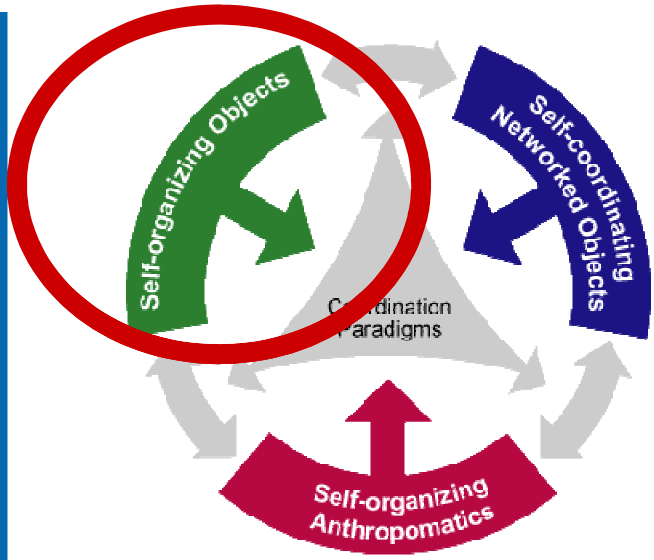


Dynamic Modification

- Idea: Dynamic Modifications bound to Transitions



Structuring Research



Self-Coordinating Objects

Example: Evolution of self-adapting virtual creatures

Reflective knowledge:

- information from the environment
- previous environmental reaction on activities of the system
- global knowledge the system

⇒System is able to learn and hence to self-optimize its behavior.

Research Directions:

- Self-learning strategies for motion.
- Cognitive actuators with integrated smart sensors and inherent intelligence.
- Ability to supervise itself and to take care of its health
- "Self-x" paradigms of Autonomic Computing

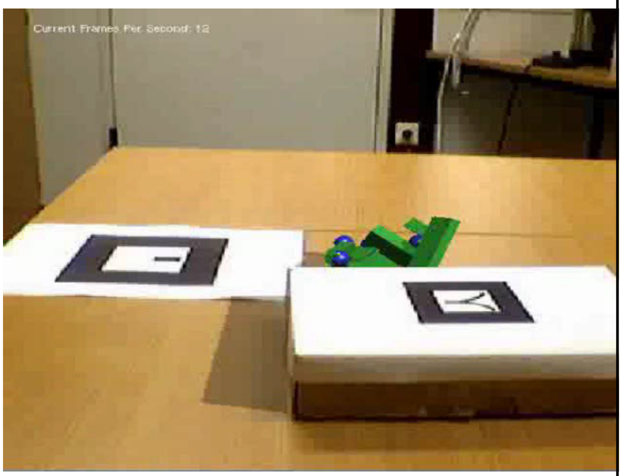
Self-Coordinating Objects: Virtual Creatures (with Chr. Geiger, T. Schmidt)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Create, simulate and represent virtual Creatures

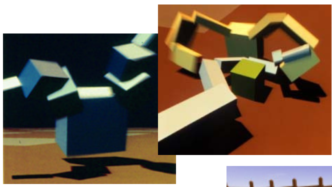
Fully self-contained approach
- Specify target behaviour
- Generate appropriate body structure and control system using self-optimization



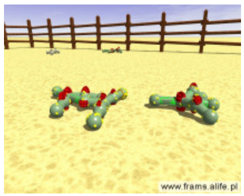
Virtual Creatures: Similar Approaches

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

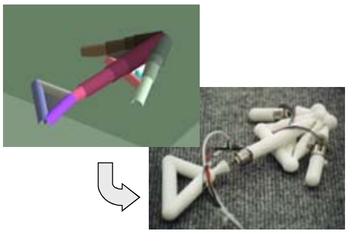
- **Karl Sims**
  - Swim, jump, follow light point, competition



- **Framsticks**
  - Nice visualisation
  - Comparison of different kinds of coding genes in genetic algorithm

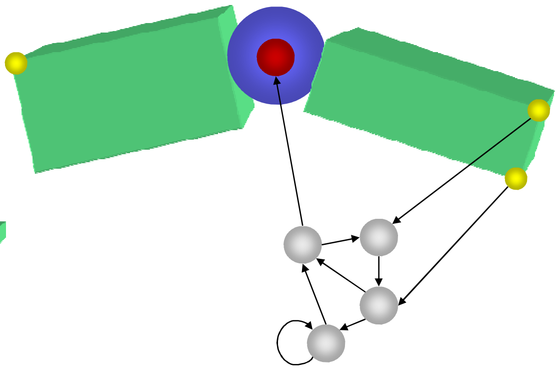


- **Golem**
  - Automatic fabrication of robots by robots
  - Design process
    - Simulation & virtual evolution
  - Fabrication
    - Plan 3D-Plastic printer
    - Insertion of electric motors



Virtual Creatures: Body Structure

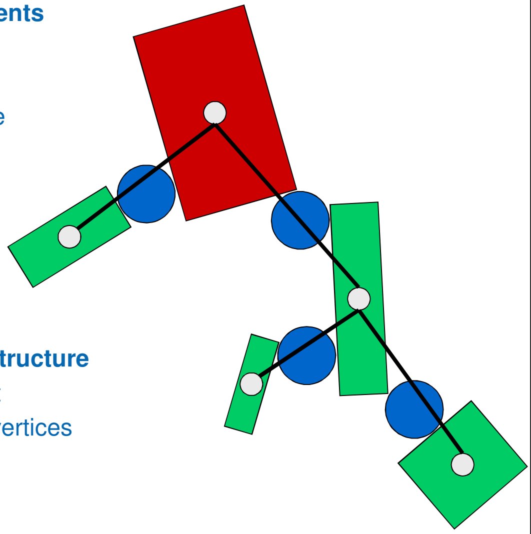
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



- **Morphology**
  - Solids
  - Joints
  - Sensors
  - Actuators
- **Control**
  - Artificial Neural Net
- **Rigid Body Dynamics**
  - Simulated physics

Virtual Creatures
Hierarchical Composition

§ **Combination of segments via joints**
- Arbitrary positions on segment surface



§ **Segmente build tree structure**
- single root segment
- joints modelled by vertices

Virtual Creatures:
Neurons

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

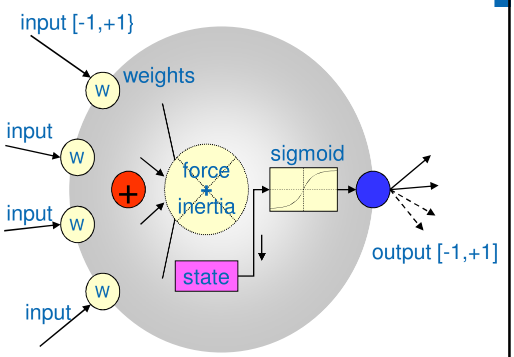
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Execution cycle:

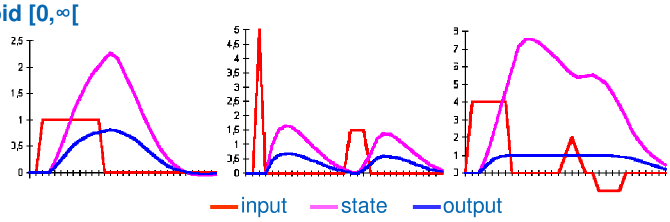
1a. Weighted sum over all inputs

1b. Calculate excitation state

2. Limit output value



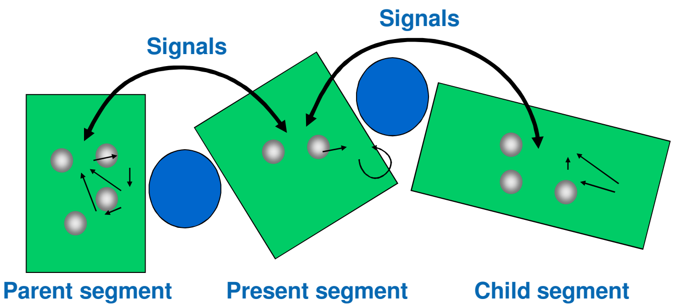
Parameter
- **force [0,1]**
- **inertia [0,1]**
- **sigmoid [0,∞[**



Virtual Creatures: Coupled ANNs

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Network made of 2-8 neurons per segment
- Coupling of ANNs
  - With parent segment
  - With child segments



Virtual Creatures: Sensors

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

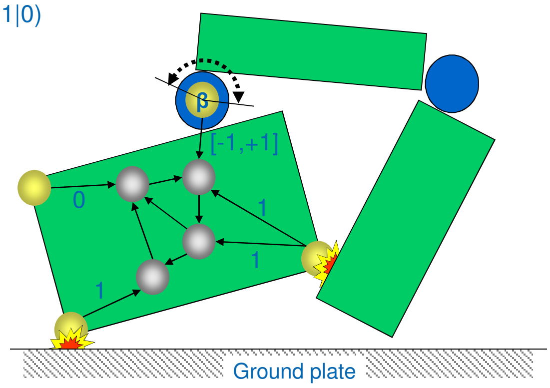
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Contact

- Segment segment
- Segment ground
- Measurement: contact yes/no
- Output signal: (1|0)

Angle of joints

- Measurement: [-PI, +PI]
- Output signal: [-1,+1]

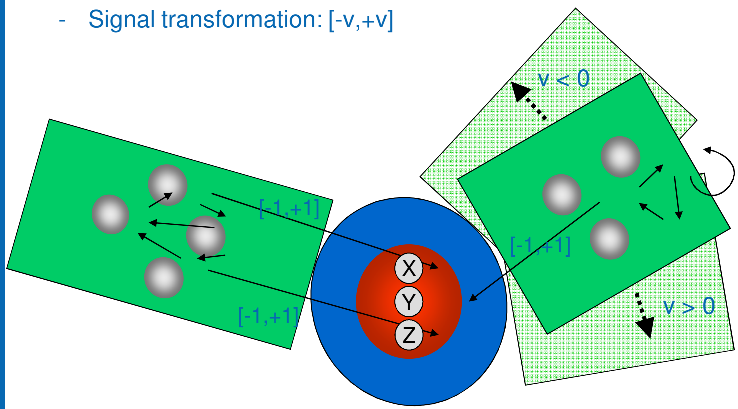


Virtual Creatures: Actuators

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Joint motor

- Turning segments around X-, Y- und Z-axis
- fixed maximal turning speed: v
- Input signal: [-1,+1]
- Signal transformation: [-v,+v]



Automatic Evolution of Creatures

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Creating specific behaviour**
  - Select adequate morphology
  - Design (distributed) control system
  - combine Morphology and control system
- **Hardly manageable by manual design**

- **Specify goals**
- **Generate and optimize automatically the creatures**
  - Without user interaction

- **Domain characterized by complex, mutually dependencies**

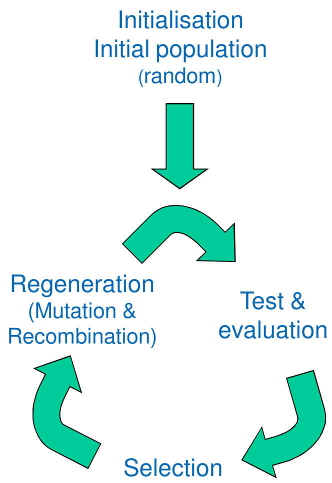
- **Adequate method Evolutionary algorithms**

Automatic Evolution of Creatures: Evolutionary Algorithm

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Heuristic Optimization**
- **Candidates for solving a problem evolve following the principle of evolution**
- **Each candidate solves the problem in more or less well manner**
- **„Survival of the fittest"**
- **Number of descendents proportional to fitness**

Good properties succeed



Coding the Genes

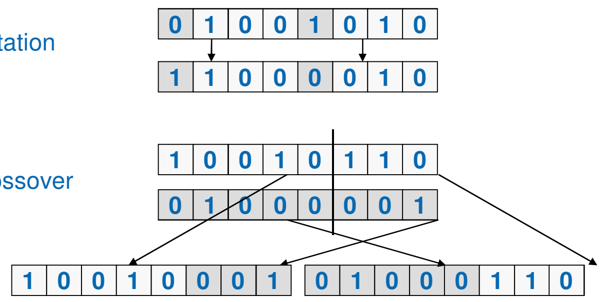
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

- **Code to represent candidate solutions**
- **Usually vectors of bits, reals, …**

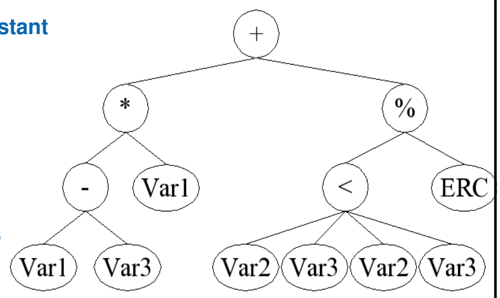
Operations

- Mutation
- Crossover

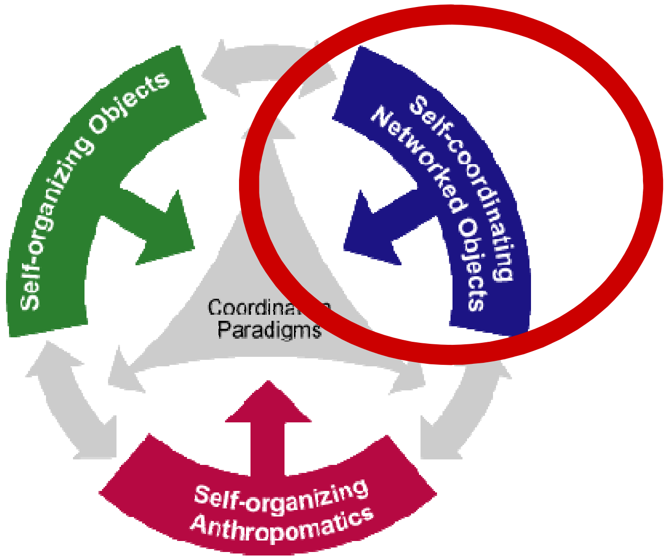


Automatic Evolution of Creatures: Genetic Programming (GP)

- **Tree structure as Gene**
- **Leaves**
  - Ephemeral random constant (ERC)
  - Variables (external)
- **Inner nodes**
  - Functions
- **Evaluation**
  - Complete traverse
  - Activation of functions using child elements as parameter
- **Mutation**
  - Select node and create new sub-tree
- **Crossover**
  - Select two nodes and exchange mutually



Structuring Research



Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Generalized view on logistics:

Services are provided by networks in the most general sense

- Networks to produce and transport physical goods
- Networks transport immaterial data

Profound commonalities between networks of different kinds that have not yet been exposed or exploited.

Some research directions

- Network-specific optimization heuristics
- Services and networks need to be composed
- Distributed, concurrent, and communicating set of processing nodes
- Operating systems themselves will become network-based

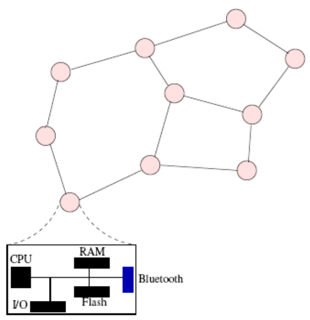
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Motivation

Ad-hoc network from embedded devices:

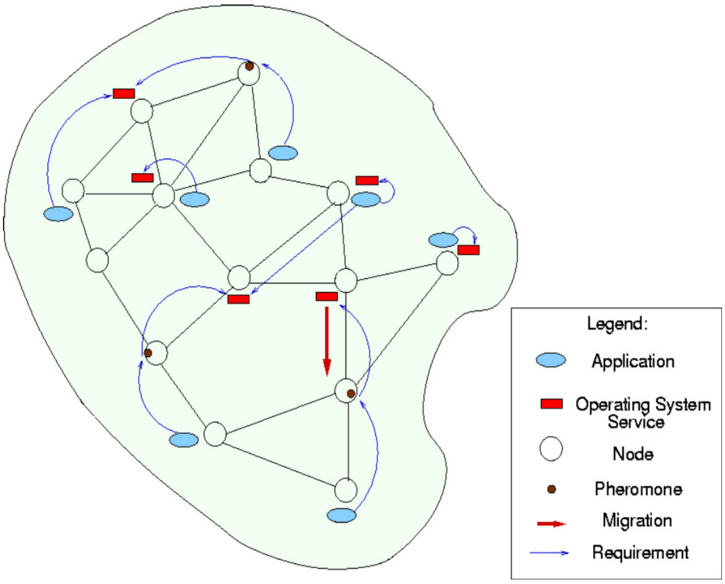
- Small processing capabilities: FPGA or microprocessor
- Small memory capacity
- Wireless interface (with ad-hoc capabilities, e.g. Bluetooth)
- Mobility

→ Applications geographically distributed need the support of a



Nano Distributed Operating System

Nano OS: Services Offered by Swarms



NanoOS

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

The basic principles of the NanoOS are:

Geographically distributed shared services Services should be shared and accessed from any processing node

Uniform environment of execution It should be provide even in the presence of dynamical change in the connection pattern (due movement)

Single Image: Each cluster contains a single consistent kernel that is the aggregation of different components. In the presence of movement, the components should stay accessible

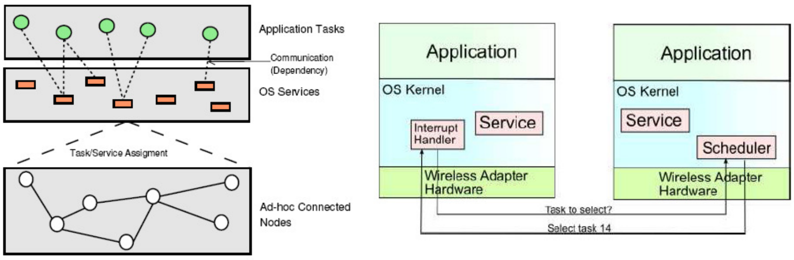
Self-adaptability The OS should be dynamic to adapt to changes in the environment and in the requirements of the applications using reconfiguration

Self-Optimization A additional target is a continuous and gradual self-optimization

Small Footprint Due the memory and processing limitation, the OS should have a small footprint

Service sharing

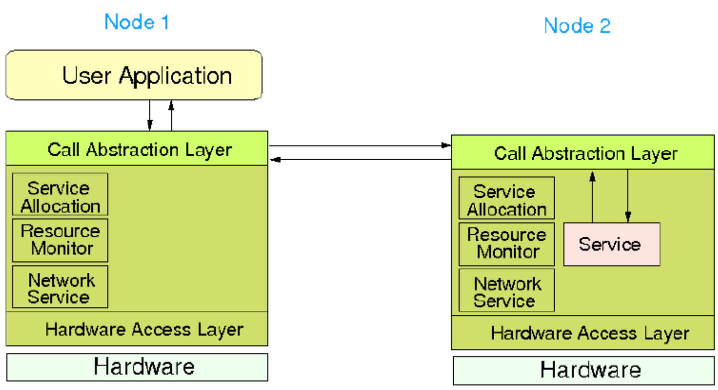
The basic principle of the operating system is the service sharing among nodes:

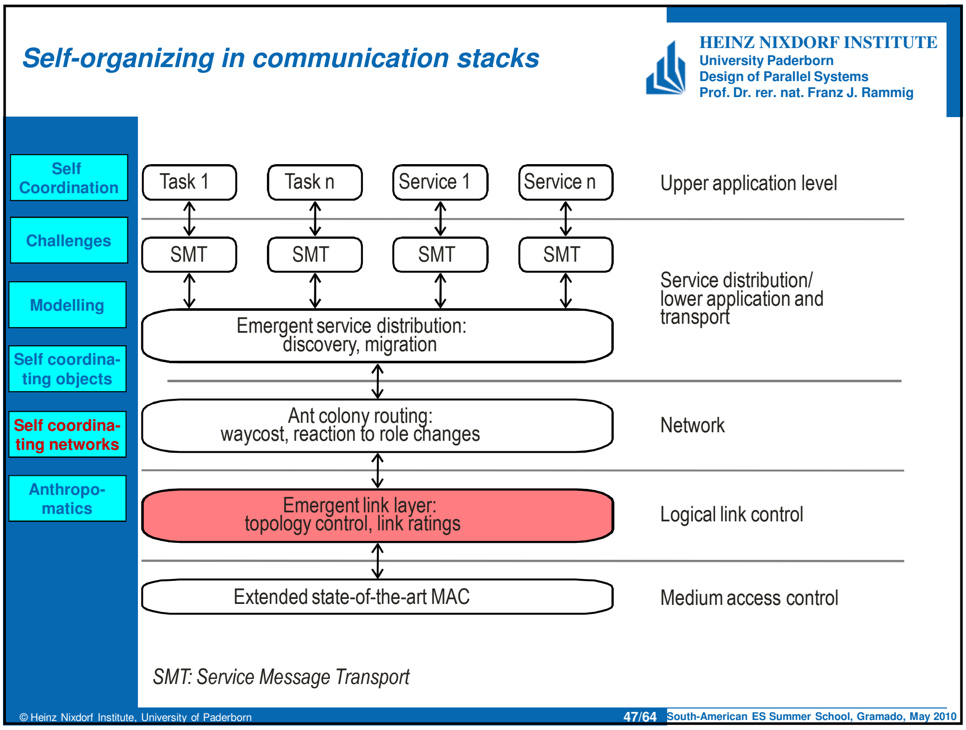


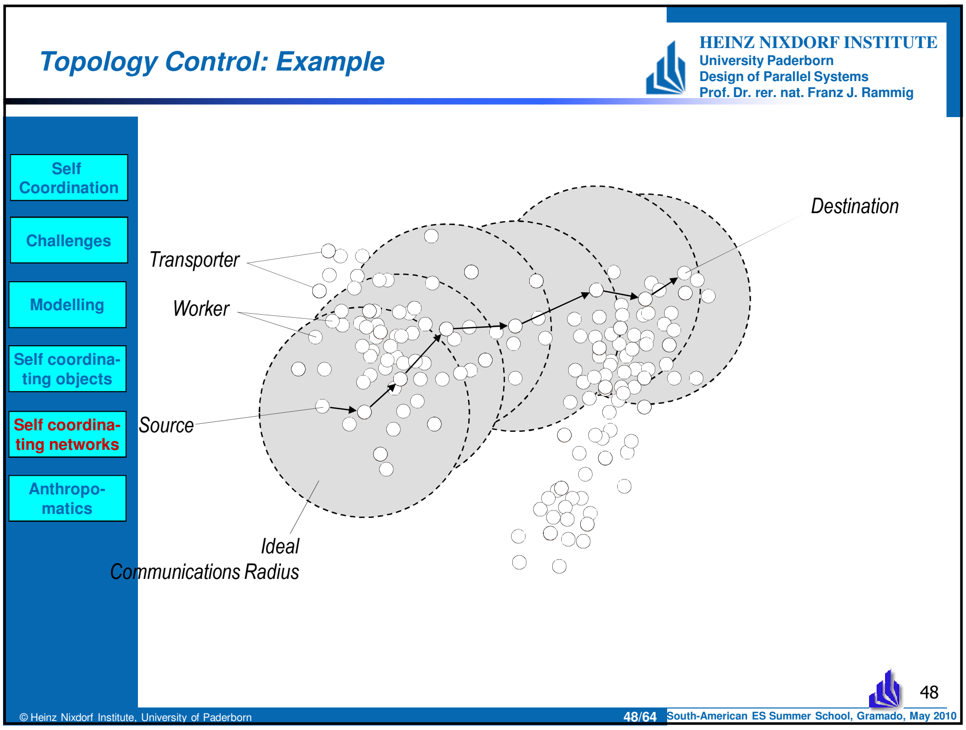
- Trade-off: Communication (execution) overhead × reduction of resource requirements

⟶ Service can change its physical position (migrate)

Node Architecture




Self-organizing in communication stacks
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics
Task 1
Task n
Service 1
Service n
Upper application level
SMT
SMT
SMT
SMT
Service distribution/ lower application and transport
Emergent service distribution: discovery, migration
Ant colony routing: waycost, reaction to role changes
Network
Emergent link layer: topology control, link ratings
Logical link control
Extended state-of-the-art MAC
Medium access control
SMT: Service Message Transport
© Heinz Nixdorf Institute, University of Paderborn
47/64
South-American ES Summer School, Gramado, May 2010

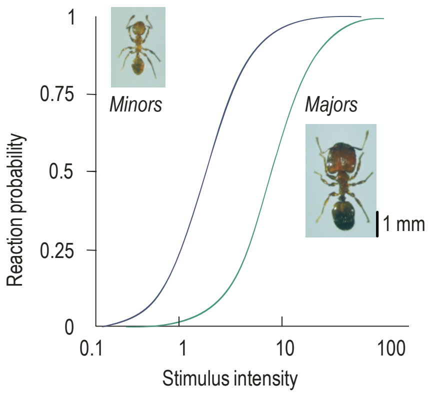

Topology Control: Example
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics
Destination
Transporter
Worker
Source
Ideal
Communications Radius
48
© Heinz Nixdorf Institute, University of Paderborn
48/64
South-American ES Summer School, Gramado, May 2010

Topology control: division of labour in ants

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

- Three phases: clusterhead emergence, members selection, maintenance
- Clusterhead emergence: based on the division of labor of natural ants
  - Minors: Quotidian tasks
  - Majors: Defence, seed milling, food storage
  - Elasticity: Ability to respond to alterations θ
  - Threshold : Tendency to react to a stimulus s
  - Stimulus : Intensity of quantitative cues sensed
  - Probability for reaction

$$T_\theta(s) = \frac{s^n}{s^n + \theta^n}$$



In terms of minor-specific tasks

Topology control: division of labour between nodes

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

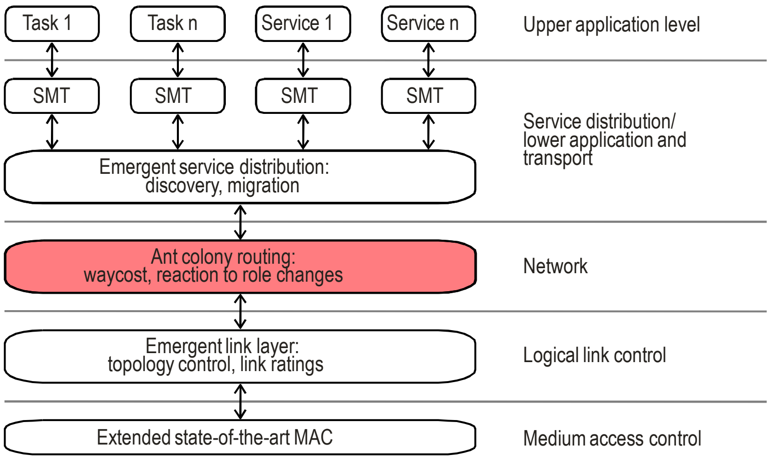
Transporter

- Primary: Routing
- Threshold:
  + Backbone less dense
- Stimulus:
  + More energy consumed since becoming transporter

Worker

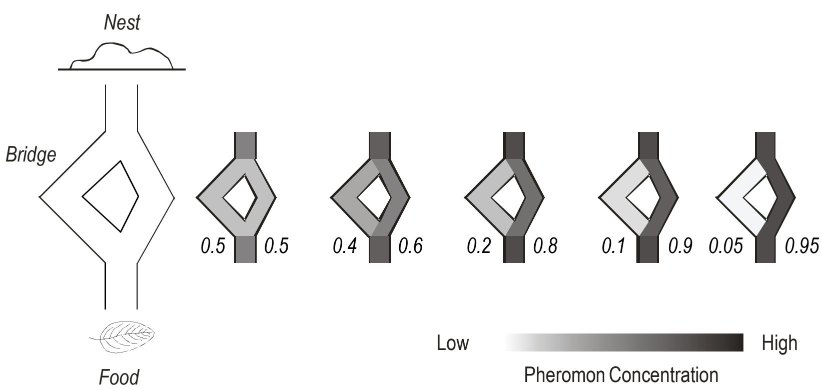
- Primary: Computing
- Threshold:
  + More energy consumed
  + Fewer neighbouring worker nodes
- Stimulus:
  + Longer disconnection from backbone

Self-organization in communication stacks



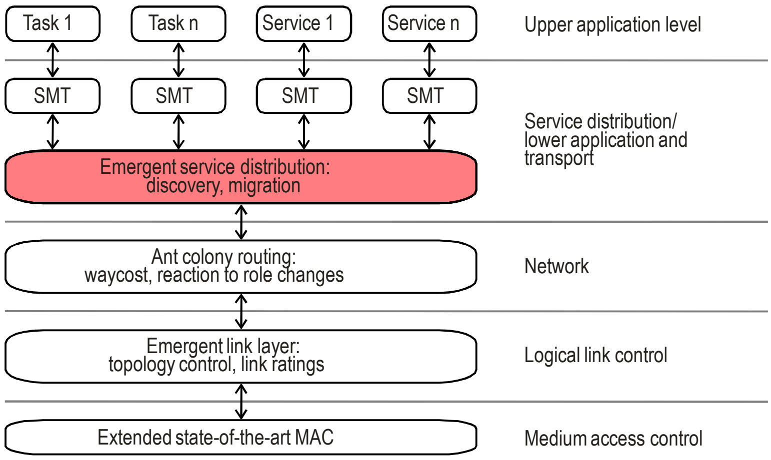
SMT: Service Message Transport

Ant Colony Routing



- Pheromon estimation und adaptation based on transport cost function based on connection ratings

Self-organization in communication stacks



SMT: Service Message Transport

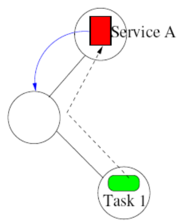
Service Migration

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

The connectivity pattern changes dynamically, self-optimization is desired $\longrightarrow$ migration of OS services necessary

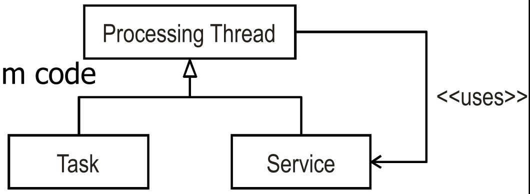
- Dynamic reconfiguration of the OS necessary
- Global consistence should be assured, i.e., the services should be accessible even when nodes change position
- System calls from tasks should be redirected
- Dependency constraints must be observed



Model for Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- *Processing Thread*: Execution of program code together with state



  - *Task*:
    - Activated for specific mission
    - Stays on node
  - *Service*:
    - Activated to react on requests
    - May be migrated to other nodes

Service Distribution Heuristic

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

Objective: Minimization of the communication overhead

For that, the system tries to optimize the service position through *migration*

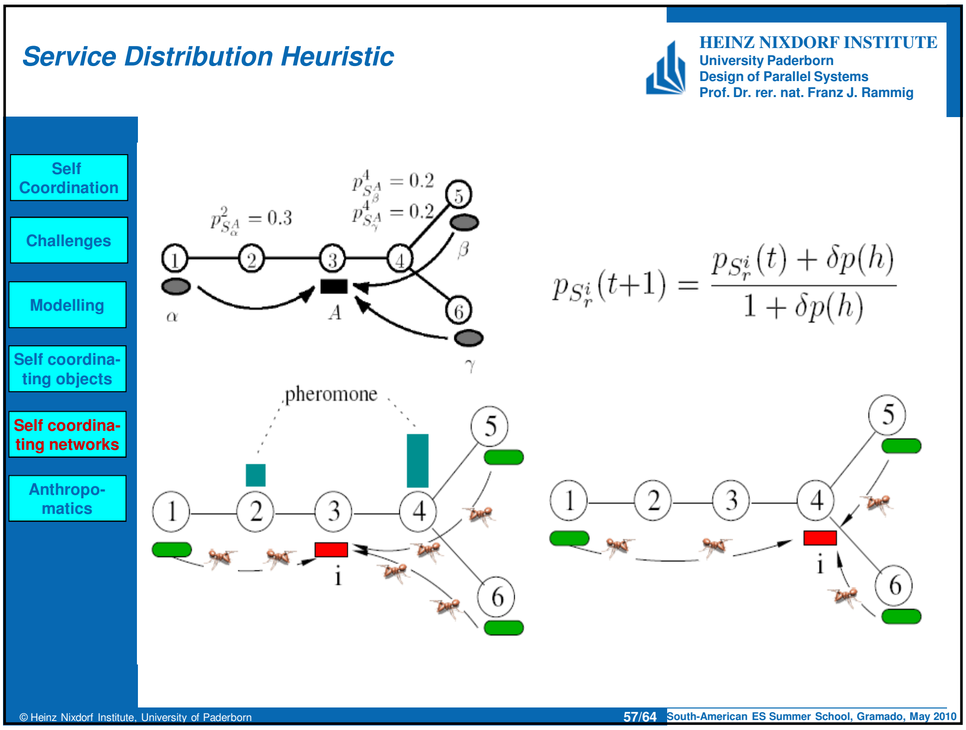
Important parameters: resource utilization in each node, link utilization.

Ant Swarm Optimization

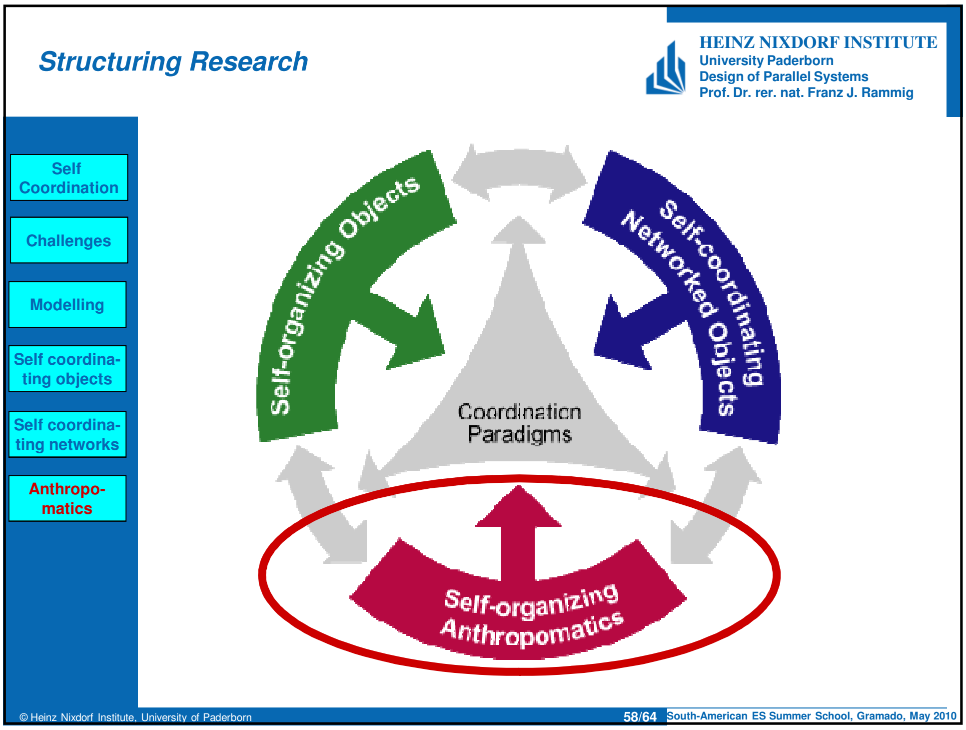
Analogy:

- Services ⟶ foot source
- Requests ⟶ Ants
- Requesters ⟶ Nest
- Wireless Links ⟶ the path used by the ants

- Agents leave pheromone through the path to the target service
- A optimization step happens when the service migrates to the neighbor node with higher pheromone concentration

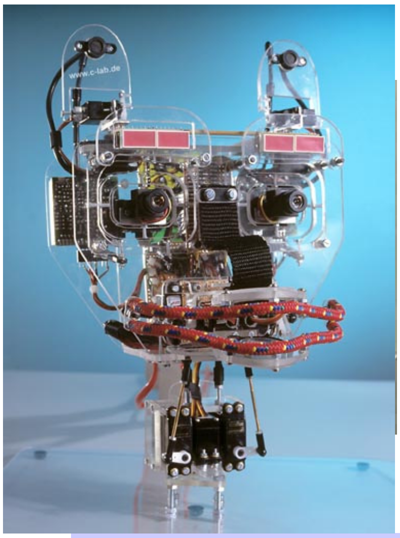


Service Distribution Heuristic
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics
pheromone
i
$p_{S_r^i}(t+1) = \frac{p_{S_r^i}(t) + \delta p(h)}{1 + \delta p(h)}$
© Heinz Nixdorf Institute, University of Paderborn
57/64
South-American ES Summer School, Gramado, May 2010



Structuring Research
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig
Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics
Self-organizing Objects
Self-coordinating Networked Objects
Coordination Paradigms
Self-organizing Anthropomatics
© Heinz Nixdorf Institute, University of Paderborn
58/64
South-American ES Summer School, Gramado, May 2010

mexi - machine with emotionally extended intelligence



- robotic head
- 15 degrees of freedom
- *expresses* emotions
- does *not* really have emotions
- behavior-based robotic system

mexi: A robot head able to show emotions

Emotions

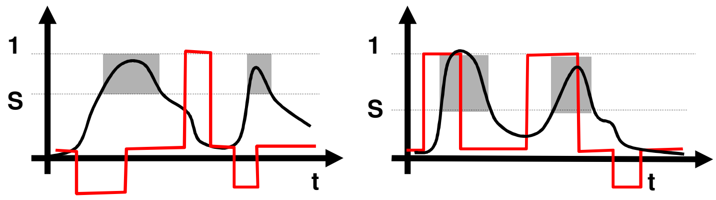
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Self Coordination
Challenges
Modelling
Self coordinating objects
Self coordinating networks
Anthropomatics

- **Temporal Excitation Functions**
  - **Amplification**
  - **Decay**
- **Evaluation Functions (Threshholds)**

Negative emotions
Temporal decay
Example: Anger

Positive emotions
Temporal decay
Example: joy

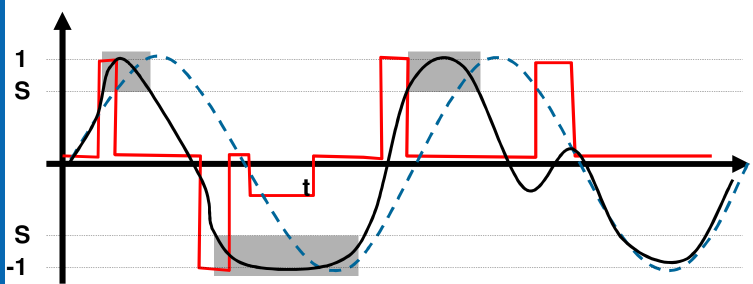


Exitation by perception

Evaluation function

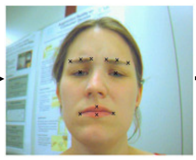
Drives

- **Modelling of drives via values within [-1,1]**
- **Periodic functions (sine or approximation)**
- **Temporal exitation functions**
- **Resulting behaviour by superposition**

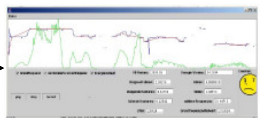


Automatic Emotion Recognition

- System for recognizing emotions from the facial expressions

Camera →  → Sad

- System for recognizing emotions from the speech input

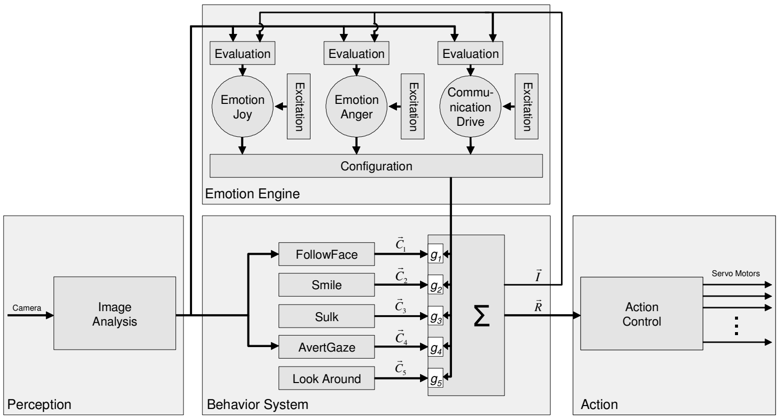
Microphones →  → Sad

- Both approaches based on the Fuzzy Logic

Emotions and Drives: Architecture of Mexi

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Self Coordination
- Challenges
- Modelling
- Self coordinating objects
- Self coordinating networks
- **Anthropomatics**



HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Thank you for your patience.

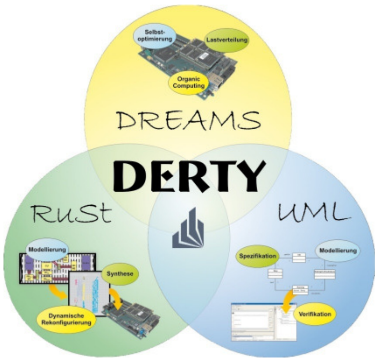
Heinz Nixdorf Institute
University of Paderborn
Fuerstenallee 11
33102 Paderborn
Germany

Phone: +49 (0) 52 51/60 65 01
Fax: +49 (0) 52 51/62 65 02
Email: franz@upb.de
http://www.heinz-nixdorf-institut.de

Self-Coordinating Objects

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Artificial Immune Systems and Applications



Norma Montealegre, Franz Rammig
Heinz Nixdorf Institute

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Immune System

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Layers of protection

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

1. **Physical barriers**

   Mechanical, chemical and biological barriers

2. **Innate immune system**

   Response is **non-specific**

   No immunological memory

3. **Adaptive immune system**

   Antigen **specific response**

   Expose leads to **immunological memory**

Immunity

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

a. **Cell-mediated immunity**

   Involves the **activation of other cells**

b. **Humoral immunity**

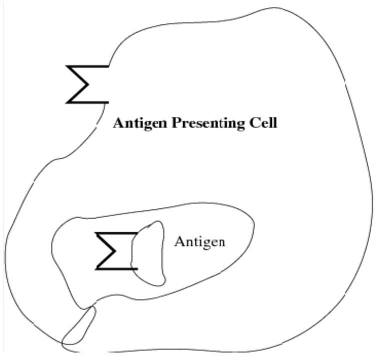
   Involves the **secretion of antibodies**

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Antigen Presenting Cells

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

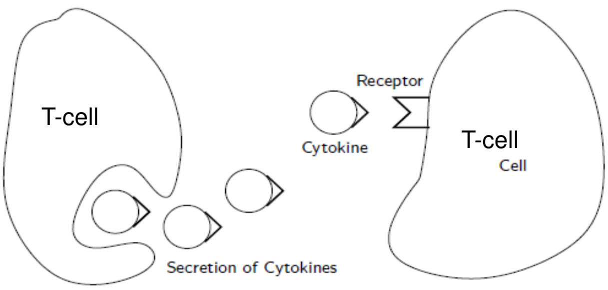
- **Digest an antigen** and present it in their extern membrane to **T-Cells**
- APCs are:
  - B-Cells
  - Dendritic Cells
  - Macrophages



T-Cells

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Maturation in the **thymus** by:
  - Negative selection
  - Positive selection
- They have **T-Cell Receptors** (TCR)
- **Activated by Antigen Presenting Cells** (APC)
- Secrete **cytokines** when activated

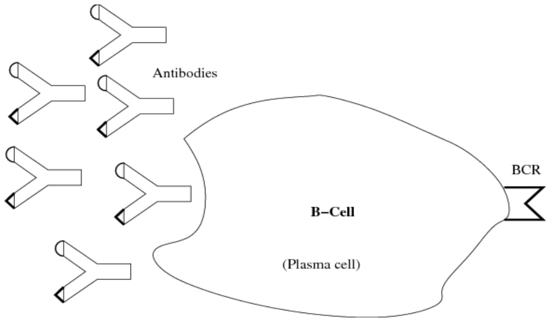


© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

B-Cells

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

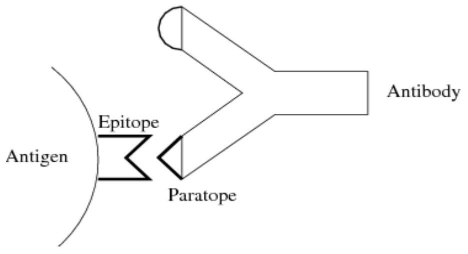
- Immature B-cells are produced in the **bone marrow** and they mature in the **spleen**
- They have **B-Cell Receptors** (BCR)
- Perform the role of **Antigen Presenting Cells** (APC)
- **Activated by T-Cells**
- Develop into **memory cells** or **plasma cells**
- **Make antibodies** against antigens when activated



Antibodies

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Produced by B-Cells**
- They have **paratopes** which bind to the **epitopes of antigens**
- They **bind to antigens** and flags them for destruction



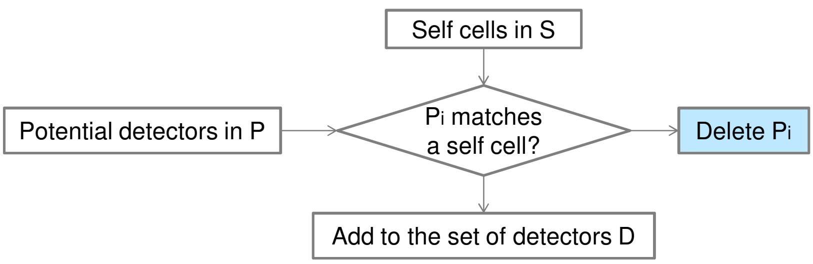
© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Selected algorithms from AIS

Self / Non-Self

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **T-Cell maturation** in the thymus
  - Positive selection - **Non-Self** is presented
  - **Negative selection** - **Self** is presented



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Negative Selection
Algorithm for Pattern Recognition

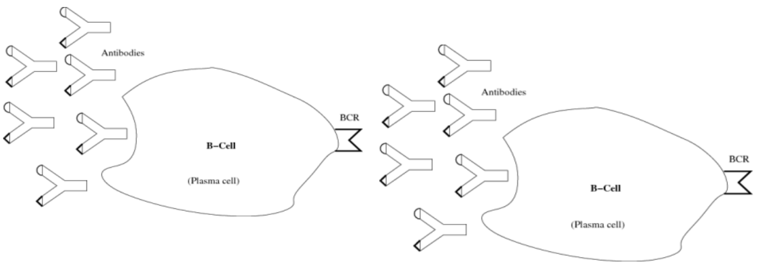
input : S_{seen} = set of seen known self elements
output : D = set of generated detectors

begin
    repeat
    Randomly generate potential detectors and place them in a set *P*
    Determine the affinity of each member of *P* to each member of the self set S_{seen}
    if at least one element in *S* recognizes a detector in *P* within a ***threshold***
        then the detector is **rejected**
        otherwise it is **added to the set** of available detectors *D*
    until stopping criteria has been met
end

Clonal Selection

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **B-Cell activation**
  - **Affinity maturation**
    - **Proliferation** of B-cells is **proportional to the affinity** of the antigen that binds it, thus the higher the affinity, the more clones are produced
  - **Somatic hypermutation**
    - The **mutations** suffered by the antibody of a B-cell are **inversely proportional to the affinity** of the antigen it binds



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Clonal Selection
Algorithm for Pattern Recognition

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

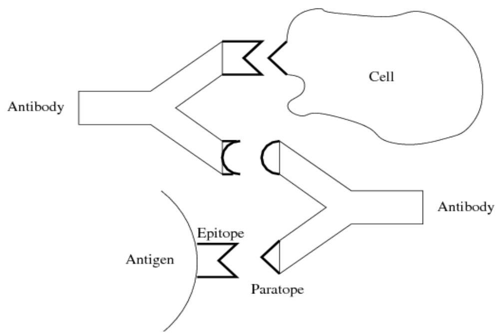
input : S = set of patterns to be recognized
n the number of worst elements to select for removal
output : M = set of memory detectors capable of classifying unseen patterns

begin
Create an initial random set of antibodies, *A*
for all patterns in *S* **do**
Determine the affinity with each antibody in *A*
Generate clones of a subset of the antibodies in *A* with the highest affinity
The number of clones for an antibody is proportional to its affinity
Mutate attributes of these clones to the set *A*
Place a copy of the highest affinity antibodies in *A* into the **memory set, *M***
Replace the ***n*** **lowest affinity antibodies** in *A* with **new randomly generated antibodies**
end
end

Immune Networks

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Epitop-Paratop interactions**
  - The immune system is an interacting **network of lymphocytes** and molecules that have variable V regions.
  - These V regions bind not only to things that are foreign to the vertebrate, but **also bind to other V regions within the system**.



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Immune Networks - Algorithm

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

input : S = set of patterns to be recognized, *nt* network affinity threshold, *ct* clonal pool threshold, *h* number of highest affinity clones, *a* number of new antibodies to introduce
output : N = set of memory detectors capable of classifying unseen patterns

begin
Create an initial random set of network antibodies, *N*
repeat
for all patterns in *S* do
Determine the affinity with each antibody in *N*
Generate clones of a subset of the antibodies in *N* with the highest affinity
The number of clones for an antibody is proportional to its affinity
Mutate attributes of these clones to the set *A* , *a*
Place *h* number of the highest affinity clones into a **clonal memory set, *C***
Eliminate all elements of *C* whose affinity with the antigen is less than a predefined **threshold *ct***
Determine the affinity amongst all the antibodies in *C*
Eliminate those antibodies whose affinity with each other is less than the **threshold *ct***
Incorporate the remaining clones of *C* into *N*
end
Determine the affinity between each pair of antibodies in *N*
Eliminate all antibodies whose affinity is less than the **threshold *nt***
Introduce a random number of **randomly generated antibodies** and place into *N*
until a stopping criteria has been met
end

Dendritic Cells

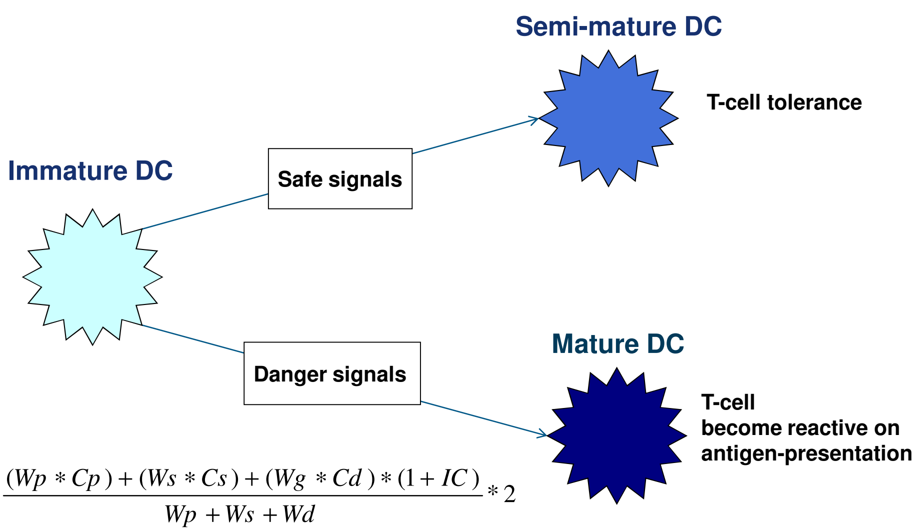
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Activation of T-Cells**
  - T-Cells **require a co-stimulatory signal** from non-antigen-specific APCs in order to initiate an effective adaptive immune response
- **Danger Theory**
  - Explanation of the **tolerance** to agents outside of the host
  - APCs are activated via an alarm: **danger signals**
  - These **activated APCs** will then be **able to provide the necessary co-stimulatory signal to the T helper cells**
  - The **danger signals** are emitted by ordinary **cells of the body that have been injured due to attack by pathogen**

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Dendritic Cells

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



$$C = \frac{(Wp * Cp) + (Ws * Cs) + (Wg * Cd) * (1 + IC)}{Wp + Ws + Wd} * 2$$

Dendritic Cells – Algorithm

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

input : S = set of data items to be labeled safe or dangerous
output : D = set of data items labeled safe or dangerous

begin
Create an initial population of dendritic cells (DCs), *D*
Create a set to contain migrated DCs, *M*
for all data items in *S* **do**
Create a set of DCs randomly selected from *D* , *P*
for all DCs in *P* **do**
Add data item to DCs collected list
Update **danger, PAMP and safe signal concentrations**
Update **concentrations of output cytokines**
if concentration of co-stimulatory molecules is above a **threshold**
Migrate the DC from *D* to *M* and create a new DC in *D*
end
end

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Dendritic Cells – Algorithm (part 2)

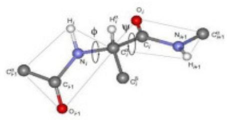
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

for all DCs in *M* **do**
if output concentration of non-inflammatory cytokines is greater than inflamatory cytokines
Set DC to be **semi-mature**
otherwise set as **mature**
end
for all data items in *S* **do**
Calculate number of times data item is presented by a mature DC and a semi-mature DC
if presented by more than semi-mature DCs than mature DC's
Label data item a **safe**
otherwise label as **dangerous**
Add data item to labeled set *M*
end
end

Formal Immune Networks

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Network of proteins**
  - Epitopes and paratopes are kind of surface **proteins**
  - A protein is a molecule composed of aminoacids arranged in a chain with a 3D recognized
  - One epitope matches with a paratope in **molecular recognition**



- **Antigen presentation**
  - Only certain epitopes of a pathogen are presented by the APC because they are immunodominant
  - Therefore only the **dominant epitopes** of antigens are used – **Compression by SVD**
- **B-Cell => (cytokine,proteins) maturation**
  - **Apoptosis** is the mechanism by which the body removes both the **ineffective** and the potentially damaging immature cells
  - **Autoimmunity** is the failure of an organism to recognize its own constituent parts as dangerous, which allows an immune response against its own cells and tissue
  - Ineffective cells are deleted and erroneously deleted cells are reinserted

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

cFIN - Algorithm

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Input: Data Matrix *[c,A]*, threshold, *p* number coordinates
Output: Compressed Matrix *[cpau,Pau]*

Calculate de **Singular Value Decomposition** of *A*
foreach *Ai* vector **do**
    foreach coordinate in [1,…,p] **do** Calculate coordinate of the *Ai* vector in *Pi* **end**
end
foreach *Pi* **do Apoptosis**
    foreach *Pj* **do**
    Calculate the distance between *Pi* and *Pj*
        if distance < threshold **and** same cytokine **then** eliminate *Pj*
    end
end
foreach *eliminated Pi* **do Autoimunization**
    foreach saved *Pj* **do**
    Calculate the distance between the *eliminated Pi* and the *saved Pj*
    end
    Find the *saved Pj* with minimum distance to the *eliminated Pi*
    if same cytokine **then** reinsert *eliminated Pi*
end

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Artificial Intelligence

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Artificial Intelligence

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

„The science and engineering of doing intelligent machines"

- Intelligent machines are built around **automated inference engines**
- Based on certain conditions, the system infers consequences (**if-then**)
- AI applications in term of **consequences**
  - **Classifiers: „if" adjective „then" noun**
  - **Controllers: „if" adjective „then" verb**
  - Classifiers and controllers make use **pattern recognition** for **„if" adjective**

Learning

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Supervised learning**
  - Is a technique for **deducing a function from training data**
  - The training data consist of **pairs of inputs and desired outputs**
  - The output of the function can be a continuous value (**regression**) or can predict a class label of the input object (**classification**)

Learning

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Unsupervised learning**
  - Is a class of problems in which one seeks to determine **how the data are organized**
  - The learner is given **only unlabeled examples**
  - One form of unsupervised learning is **clustering**

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

cytokine – Formal Immune Networks

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

cFIN

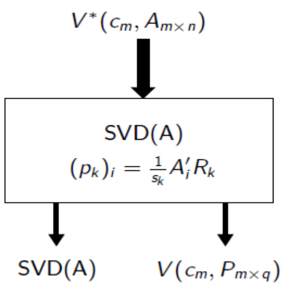
- For **pattern recognition**
- Uses **supervised learning**
- Given **training data** in a matrix *A* with respective classes in *c*, we have a pair ***(c,A)***
- (*ci,Ai*) is the expresion of a B-Cell with a cytokine and proteins which will go into maturation

Learning Compression

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

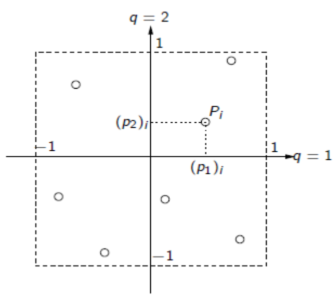
Singular Value Decomposition

$$A = s_1 L_1 R_1' + s_2 L_2 R_2' + s_3 L_3 R_3' + \ldots + s_k L_k R_k' + \ldots + s_r L_r R_r'$$



$k = 1, \ldots, q$

$i = 1, \ldots, m$

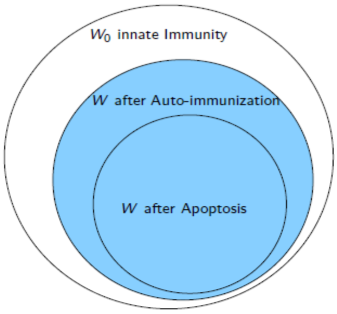


Two-dimensional FIN's space

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Learning
Apoptosis and Autoimunization

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Apoptosis

When $V_i \in W$ recognizes $V_k \in W$, then eliminate V_k of cFIN

Auto-immunization

Removed cells V_i nearest to a cell V_k belonging the set W will be inserted again if $c_i \neq c_k$

Recognition

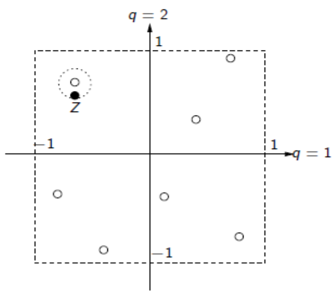
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Distance between two points in the cFIN space

$$d_{ij} = max\{| (p_1)_i - (p_1)_j |, \dots, | (p_q)_i - (p_q)_j |\}$$

Recognition with threshold

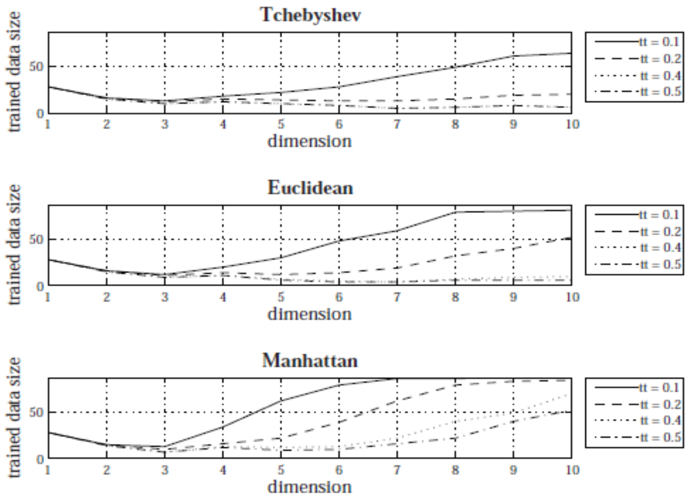
$$c_i = c_k$$
$$d_{ik} \leq h$$



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Distance calculation

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Applications

Wire Bonding

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Faulty wire bonding recognition and classification

Setup 2			
	Training	Test	
Set name	**H**	**T**	*Class c_i*
Good bond	34	27	*0*
Improperly mounted wedge	5	6	*1*
Plastic particles	3	3	*2*
Surface contaminated by a finger print	2	5	*3*
Double Bond	0	1	*4*
Total vectors	44	42	
Training matrix **A** size	44×1726		
Test matrix **T** size	42×1726		

Recognized vectors per class for the set of the second setup

Setup 2							
Class		*0*	*1*	*2*	*3*	*4*	Total
opt method	$\mathbf{T}_I$	27	6	3	5	1	113
Set H							
min faults	Recognized	25 | 0 0 2 | 0	0 | 6 0 | 0 | 0	0 0 3 | 0 0	4 0 0 | 1 0	1 0 0 | 0 0	35
	Not rec. in %	7.41	0	0	80	100	14.29
	$(\mathbf{c}^{reduced}, \mathbf{P})$ size			$(\mathbf{c}^{reduced}_{12\times1}, \mathbf{P}_{12\times7})$			
min product	Recognized	24 | 3 0 0 | 0	0 | 6 0 | 0 | 0	1 2 0 | 0 0	5 0 0 | 0 0	1 0 0 | 0 0	30
	Not rec. in %	11.11	0	100	100	100	28.56
	$(\mathbf{c}^{reduced}, \mathbf{P})$ size			$(\mathbf{c}^{reduced}_{10\times1}, \mathbf{P}_{10\times1})$			
$(\mathbf{c}_{44\times1}, \mathbf{A}_{44\times1726})$	$\mathbf{A}_I$	34	5	3	2	0	44

Hardware layers of protection

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

1. **Physical barriers** - EM protection
2. **Innate immune system** - N-modular redundancy
3. **Acquired immune system** – Use of AIS methods

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

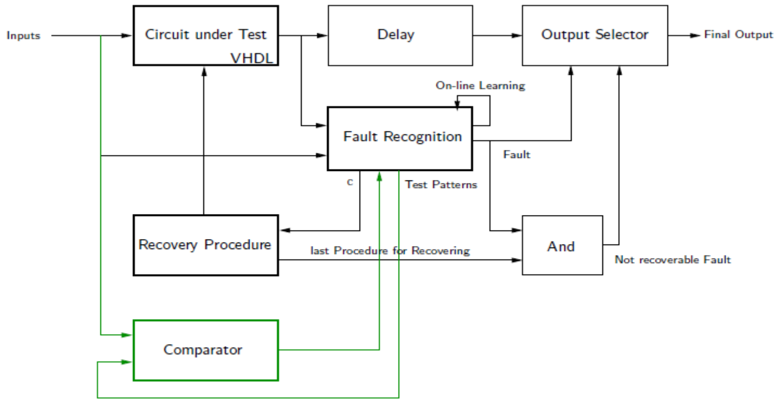
Self-Repairing

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- FPGAs for fault recovery through **dynamic reconfiguration**
- **Versatile** fault recovery
- Design **independent** of the implementation hardware
- **Concurrent** embedded test

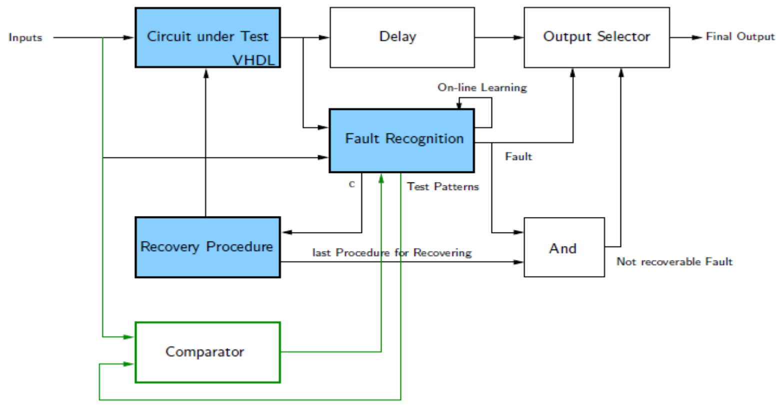
Self-Repairing Architecture

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Self-Repairing Architecture

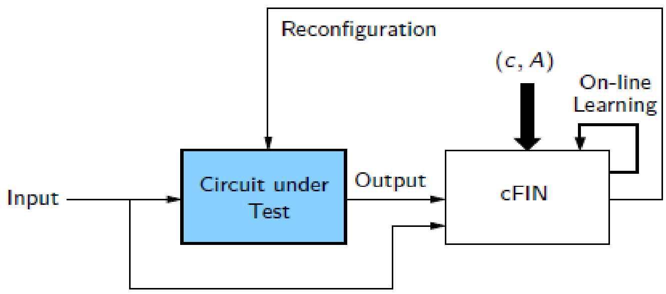
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



cFIN for Self-Repairing

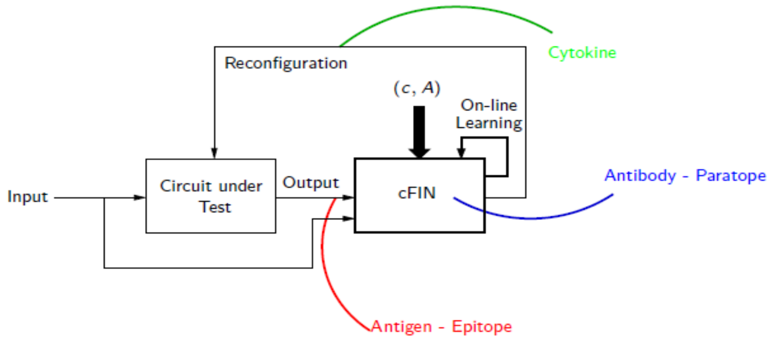
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- **Testing data** matrix **A** for compression
- **Recovery techniques** atached to the class vector **c**
- Whenever a **new test case** is desired to be added, it is not necessary to apply learning again



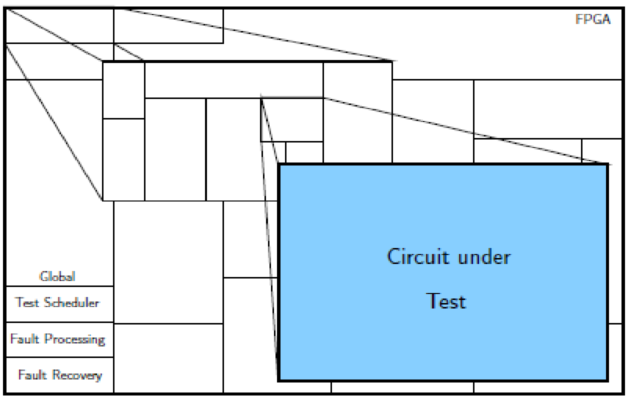
Biological Analogy

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



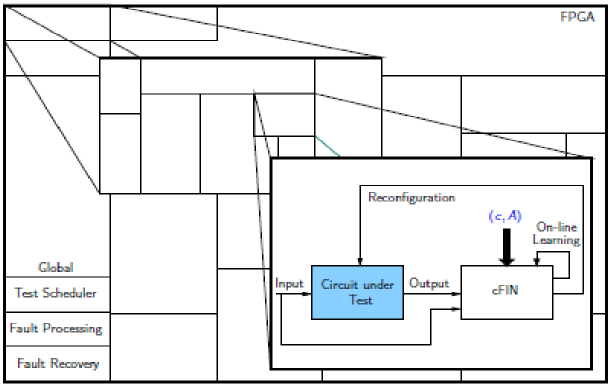
Partitioning of the system

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



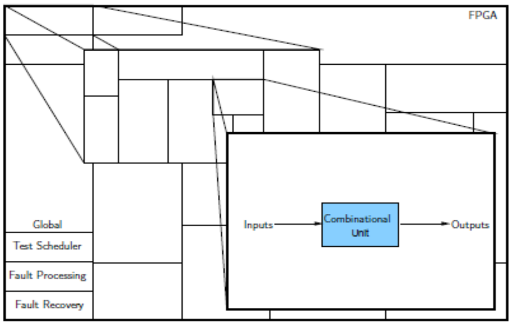
cFIN for each Partition

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Combinational Circuits

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



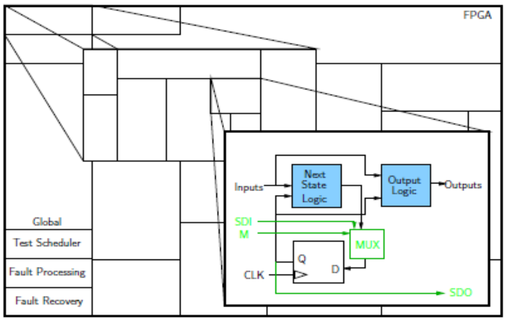
Inputs		Output
0	0	1
0	1	0
1	0	0
1	1	0

$$A_i = [Inputs|Outputs]$$

© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Sequential Circuits

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

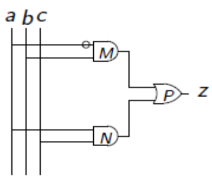


Inputs		Present States		Next States		Output
0	0	0	0	0	0	1
0	1	0	0	0	1	1
1	0	0	0	0	1	1
1	1	0	0	1	1	1

$$A_i = [Inputs|PresentStates|NextStates|Outputs]$$

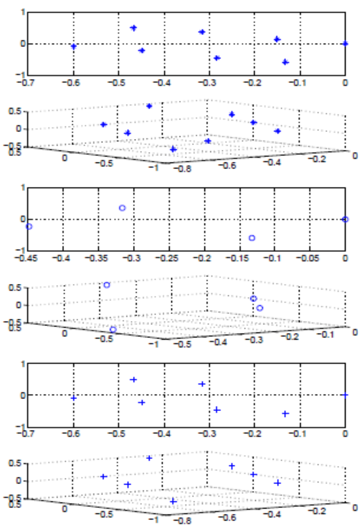
Example: Combinational Circuits

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



$$(c, A) = \left(\begin{array}{cccc|cccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right)$$

$$(ci, A_i) = (Self\,|\,Reconf\,\{MNP\}\,|\,Inputs\{abc\}]\,|\,Output(z))$$



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

Example: Sequential Circuits

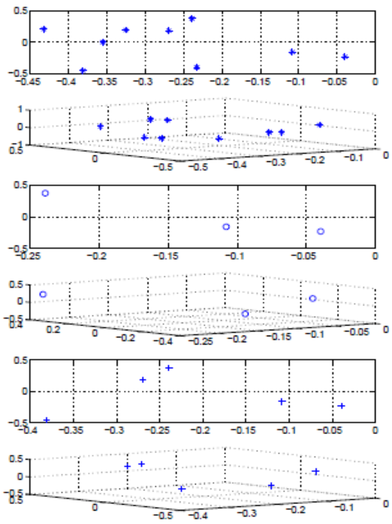
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

A sequential system has two inputs, X, Y and one output, Z. When the sum of the inputs is a multiple of 3, the output is true, otherwise it is false.

$$(c, A) = \left(\begin{array}{c|ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{array}\right)$$

$$(ci, A_i) =$$

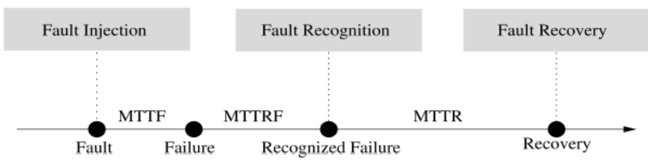
$$(SystemReset|Inputs\{XY\}]|PresentStates|NextStates|Output(Z$$



Performance Measurement

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

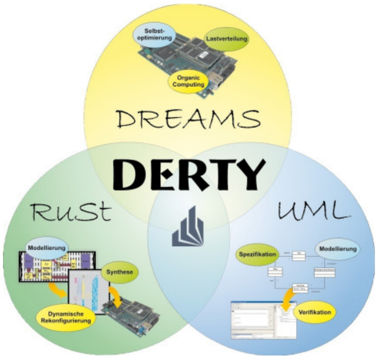
- **MTTF** - Mean Time To Fail
- **MTTRF** - Mean Time To Recognition of the Failure
- **MTTR** - Mean Time To Recovery
- A **compressed testing data set** leads to **small failure recognition times and soon fault recovery**



© Prof. Dr. rer. nat. F. J. Rammig, Heinz Nixdorf Institute, University of Paderborn

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Thank you for your attention



Heinz Nixdorf Institute
University of Paderborn
Design of Distributed Embedded Systems
Fuerstenallee 11
33102 Paderborn

Phone: ***+49 5251 606501***
Fax.: ***+49 5251 606502***
E-mail: ***franz@hni.upb.de, norma@upb.de***
http://www.hni.uni-paderborn.de/en/eps

Self-Organizing Construction of Connected k-Hop Dominating Sets in Wireless Sensor Networks

Franz Rammig and Peter Janacik
May 12, 2010

franz@uni-paderborn.de
Heinz Nixdorf Institute
University of Paderborn
Germany

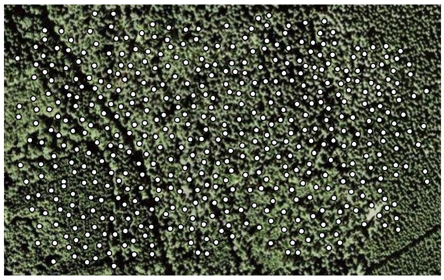
Outline

- **Introduction**
  - Wireless Sensor Networks
  - Definition
  - Motivation
- State of the Art
- Proposed Approach
  - Inspiration & Design Considerations
  - Behavior Block I: Initial Dominating Set Construction
  - Behavior Block II: Transformation to a Connected k-Hop Dominating Set
  - Properties
- Evaluation
  - Setup
  - Results
- Conclusion

Wireless sensor networks (WSNs)

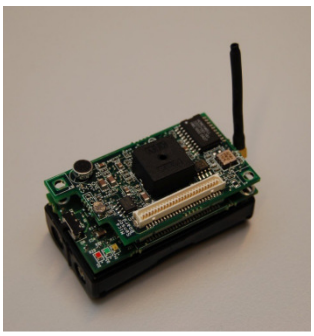
- Monitoring of large areas in which the network is deployed
- Wide range of applications: habitat monitoring, machine surveillance, precision agriculture, etc.
- Example: ExScal* (by Berkeley et al.)
  - published experiment with 1200 nodes in 1,300 x 300 m area; experiment with 10,000 nodes classified

ExScal Node



* Arora et al. ExScal: Elements of an Extreme Scale Wireless Sensor Network. In: *Proceedings of 11th IEEE Int. Conf. on Embedded and Real-Time Computing Systems and Applications*, 2005

The challenge (1)

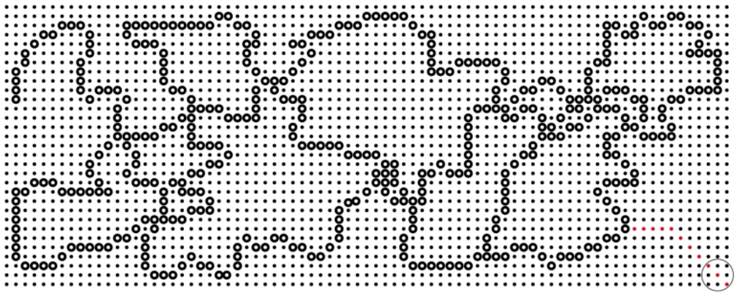


- Example hardware: Crossbow MicaZ
  - Processor
    - **16 MHz**
    - 4 Kb internal SRAM
    - Current Draw: **8 mA** active, < 15 µA sleep
  - Radio
    - **38.4 Kbaud**
    - **27 mA** tx, 10 mA rx, < 1 µA sleep

The challenge (2)

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Wireless radio consumes relative high amounts of energy compared to other node components
  - Thus: need for effective approaches, such as **connected k-hop dominating sets (CkDS)** to cope with challenge
    - Saving energy by e.g. reducing effects of broadcast storm problem and allowing non-CkDS nodes to lower duty cycles

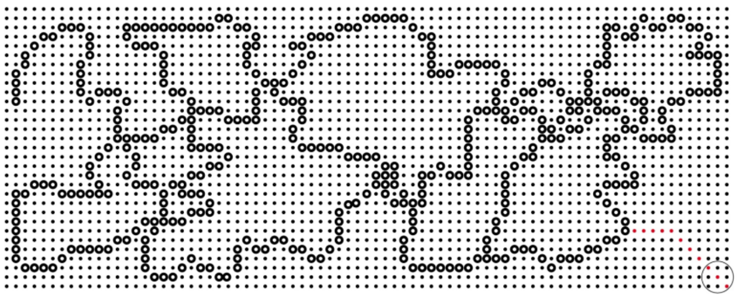


5

CkDS problem definition

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

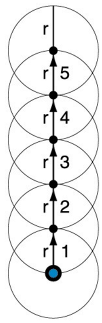
- A set of vertices S in an undirected graph G=(V,E) is a connected k-hop dominating set (CkDS), if two conditions are satisfied:
  - each vertex v from the set of all vertices in the graph, V, is either in S or there exists a path of length m, m<=k, between v and a vertex in S
  - between each pair of vertices in S, there exists a path which consists only of vertices from S



6

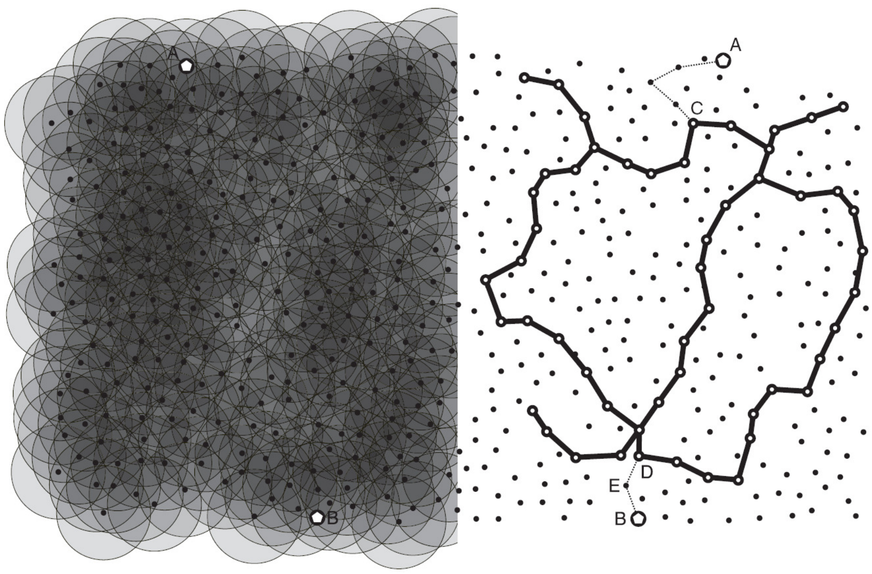
CkDS motivation: overview

- CkDS are used in wireless sensor networks for/to achieve
  - reduction of overhearing, idle listening, and collisions
  - lower duty cycles and delays
  - reduction of broadcast storm problem
  - *approximation of area coverage*
  - creation of rendez-vous areas
  - providing an adjustable amount of dominating nodes/coverage compared to CDS (i.e. CkDS, k=1)
  - etc.



Area coverage example with r (e.g. =100m): maximum possible communication/sensing radius

CkDS motivation: broadcast storm

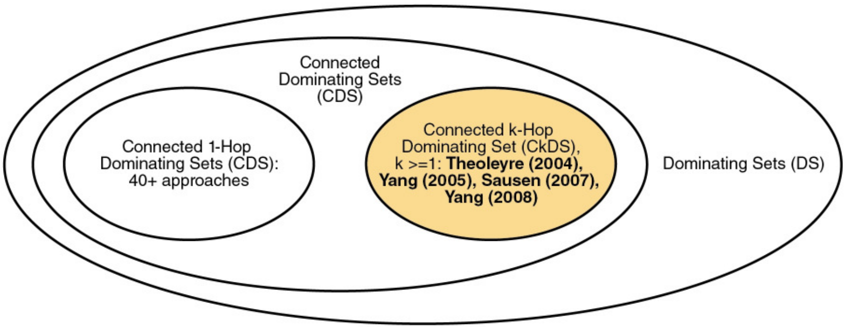


Outline

- Introduction
  - Wireless Sensor Networks
  - Definition
  - Motivation
- **State of the Art**
- Proposed Approach
  - Inspiration & Design Considerations
  - Behavior Block I: Initial Dominating Set Construction
  - Behavior Block II: Transformation to a Connected k-Hop Dominating Set
  - Properties
- Evaluation
  - Setup
  - Results
- Conclusion

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Overview



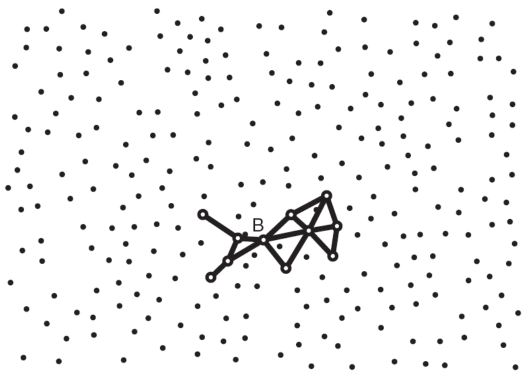
Theoleyre (connected clustering-based, 2004): Theoleyre, F. & Valois, F. A virtual structure for hybrid networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1040-1045, 2004
Yang (connected clustering-based, 2005): Yang, S.; Wu, J. & Cao, J. Connected k-Hop Clustering in Ad Hoc Networks. In *Proceedings of International Conference on Parallel Processing (ICPP)*, pages 373-380, 2005
Sausen (greedy, 2007): Sausen, P.; Spohn, M. A.; de Lima, A. M. N. & Perkusich, A. Bounded-distance multi-coverage backbones in wireless sensor networks. In *Proceedings of the 2007 ACM Symposium on Applied Computing* (SAC), pages 203-208, 2007
Yang (pruning-based, 2008): Yang, H.; Lin, C. & Tsai, M. Distributed Algorithm for Efficient Construction and Maintenance of Connected k-Hop Dominating Sets. In *IEEE Transactions on Mobile Computing*, pages 444-457, 2008

Greedy approach by Sausen et al. (2007)(1)

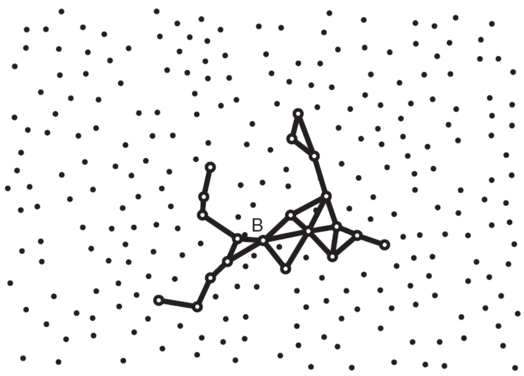
- Phase 1: acquisition of distance to base station (dbs) and knowledge of k-hop neighborhood at each node
  a) The base station broadcasts an information message (IM) including a hop counter. Nodes receiving an IM record the hop count in the dbs field and broadcast it: if it is the first one encountered or has a lower hop count than already recorded
  b) In each of the k rounds, each node broadcasts the (above) information already collected about its k-hop neighborhood to its neighbors
- Phase 2: local election at each node, performed first by base station, then by a node after the first reception of an election message (EM)
  - Node z is elected by node x (z is within k hops of x; z is dominating or 1-hop neighbor of a dominating node), if it has
    - the minimum distance to base station,
    - ties broken by largest degree, then largest ID
  - Upon election, x floods an EM (including its choice) within k hops

Greedy approach by Sausen et al. (2007)(2)

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

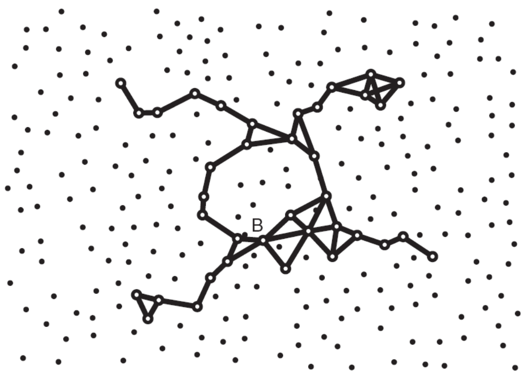


Greedy approach by Sausen et al. (2007)(2)



Greedy approach by Sausen et al. (2007)(2)

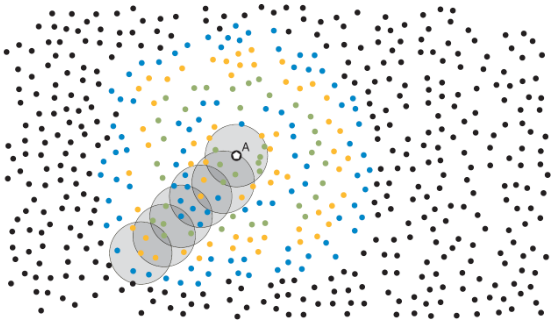
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Greedy approach by Sausen et al. (2007)(3)

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Communication cost
  - depends quadratically on k, since being dominated by k-hop flooding (flooding spreads in a circular manner, so that the area covered grows quadratically)
  - increases linearly with the average node degree d, since flooding consists of broadcasts, received by 1-hop neighbors
- Construction time
  - grows linearly with k, being dominated by k-hop flooding
  - depends linearly on the number of nodes in the network



Outline

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Random walks in natural systems

- Self-organizing approach based on random walks encountered in many natural systems
  - Crustaceans: copecod
  - Fish: haddocks, sprats, halibuts
  - Insects: apple maggot fly
  - Mammals: caribous, wolves
  - Also e.g. physics: molecule trajectories

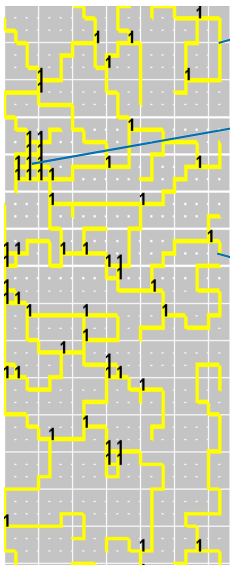
Copecod (swimming and feeding process)

Caribou (different objectives, e.g. search for calving grounds)

Very first versions of algorithm

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Based *entirely* on the *abstract* concept of random walks
- Interesting properties of random walks
  - Distributed, parallel process
  - Randomization
  - Solely local knowledge and interaction
- ... helping to achieve
  - More robustness and scalability
  - Faster execution
  - Lower information requirements (i.e. **# of broadcasts = # of dominating nodes)**
- First versions had many issues



Discon-nections

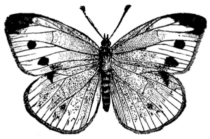
Too large clusters of CkDS nodes

Too many dead ends

Result of one of the first versions with 1000 nodes

Redesigned algorithm: considerations

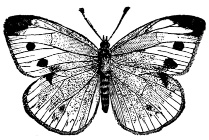
- To cope with issues, following directions were taken
  - *More imitation:* adopt solution from a **concrete** natural system which solved coverage problem successfully
    - Use elements from the solution in *Pieris rapae*
  - *More adaptation:* Add artificial elements to make up for/to cope with the differences between artificial and natural systems
    - For example add a second behavior block for interconnection

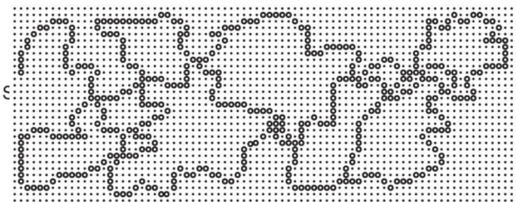


Subsequent redesign decisions

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Combine behavioral elements from flight behavior of ovipositing Pieris rapae (Root (1984)) with artificial elements
- Each fecund female maximizes payoff of random walk visiting hosts (collards) for oviposition by balancing
  - risk of too concentrated egg placement, given the possibility of localized catastrophic events (e.g. drowning in rain water, being washed off/stuck in sites)
  - the amount of energy spent for relocation





Root (1984): Root, R. B. & Kareiva, P. M. The Search for Resources by Cabbage Butterflies (Pieris Rapae). *Ecology*, 1984, 65, 147-165

Why random walks?

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Interesting properties associated with random walks
  - Distributed, parallel formation process
  - Randomization
  - Redundancy
  - Solely local knowledge and interaction
- Potential for achieving
  - Robustness
  - Scalability
  - Faster execution
  - Low communication overhead
  - Self-organization

22

General design decisions

- No superior entity that controls or influences agents or nodes
- All acting entities are agents situated in their habitat, the WSN
- The swarm of agents realizes exploration and construction in **concurrent** behavior blocks
  1. Dominating set construction
  2. Interconnection
- Agents
  - are solely aware of their own state and the state of the visited node (host) at current point of time
  - cannot perceive each other or even interact directly
  - communicate and interact solely indirectly via stigmergy

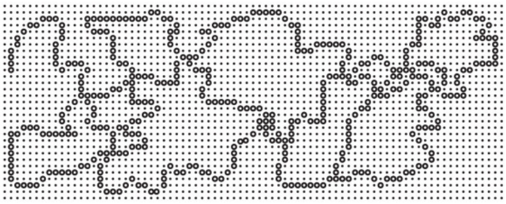
Outline

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Behavior block I: exploration

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Basic idea: agents explore the network using random walks, thereby defining paths, which serve as candidates for the addition to the dominating set
- A CkDS can be intuitively interpreted as a structure consisting of numerous intersecting paths
  - Thus the proposed approach decides after the exploration phase whether to add entire paths instead of single nodes
  - Since a path consists of numerous nodes, there is only one decision needed for many nodes, saving communication cost

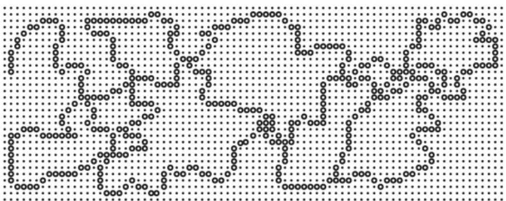


25

Behavior block I: exploration

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

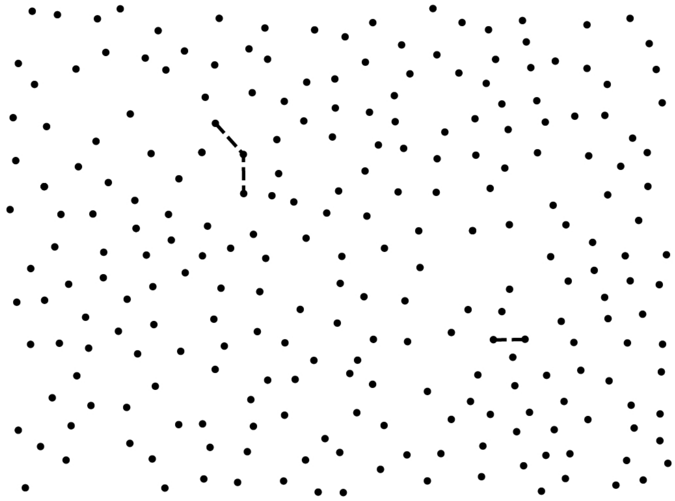
- Basic idea: agents explore the network using random walks, thereby defining paths, which serve as candidates for the addition to the dominating set
- There must be rules to select from candidate nodes after exploration
  - For protocols operating on a per-node basis, the k-hop vicinity (*growing quadratically!*) needs to be known
  - When paths resulting from random walks are used as a basis for finding decisions, path length already implies distance information
    - Communication cost is saved, since a path is the result of a series of unicasts and the *only* criterion for the decision



26

Process overview

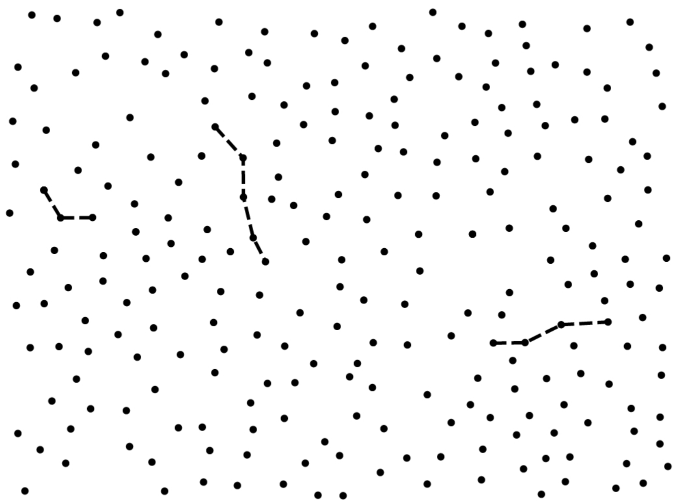
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



27

Process overview

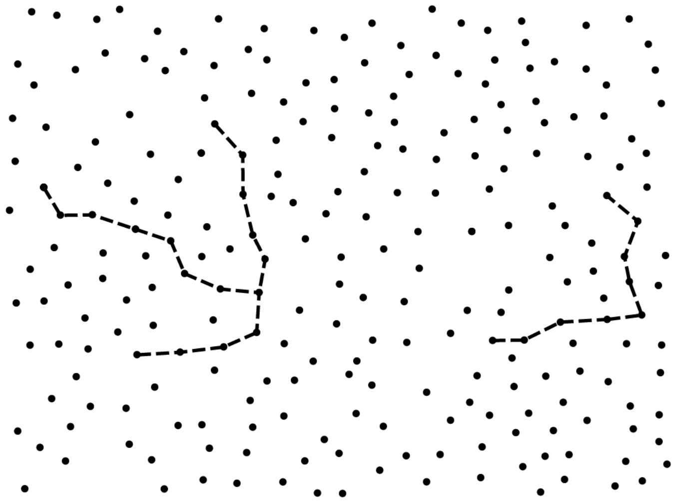
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



28

Process overview

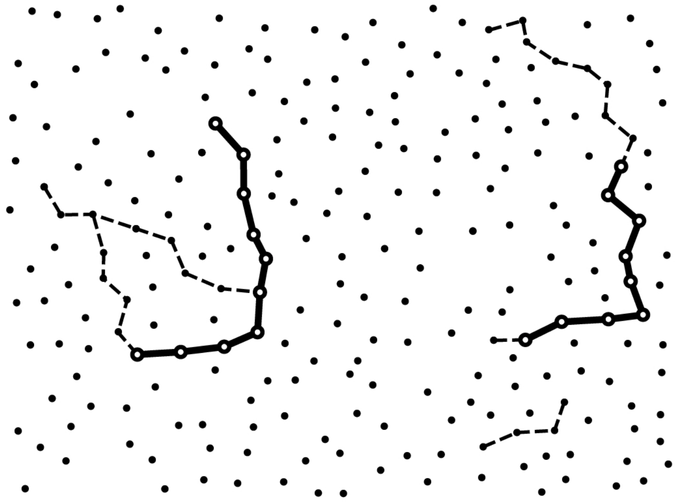
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Process overview

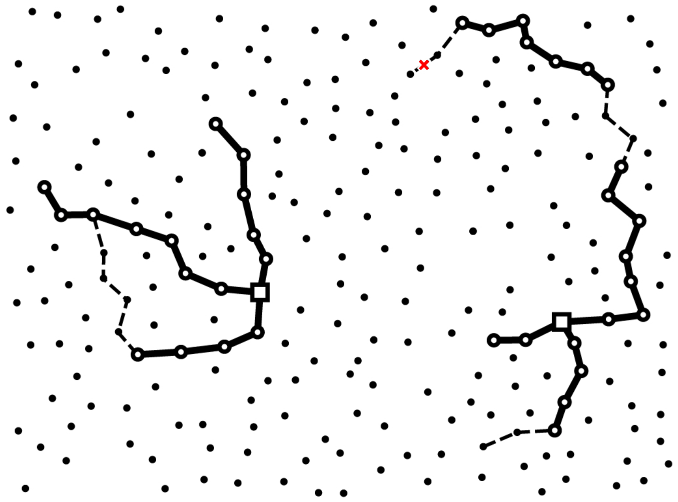
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Process overview

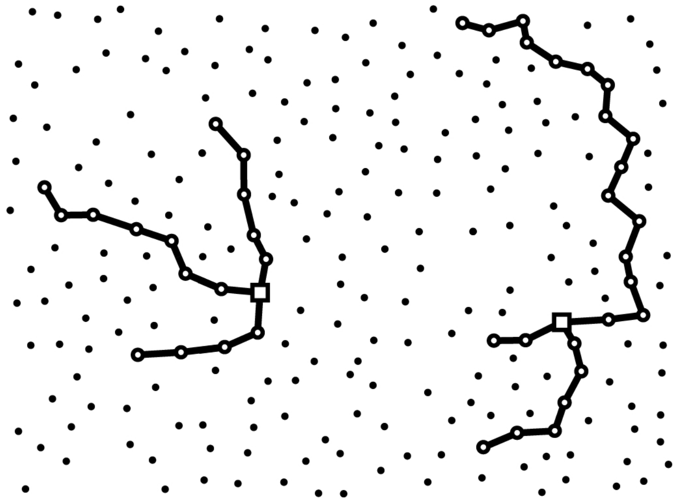


Process overview

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

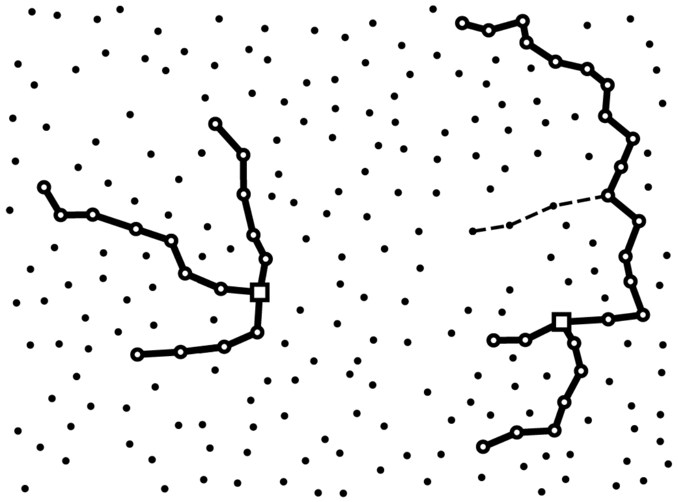


Process overview



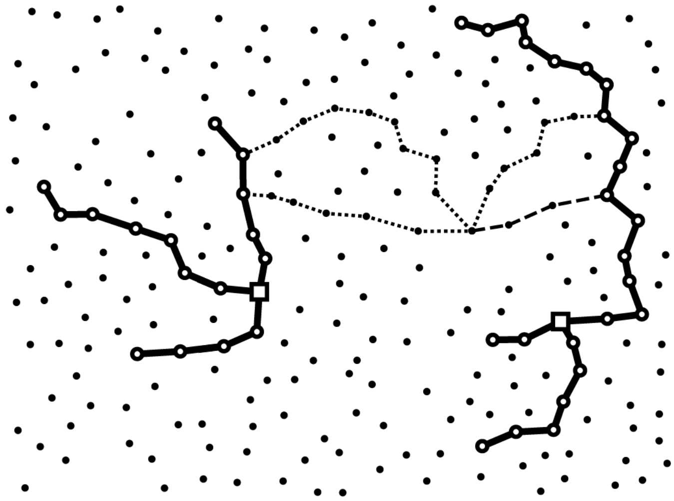
Process overview

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Process overview

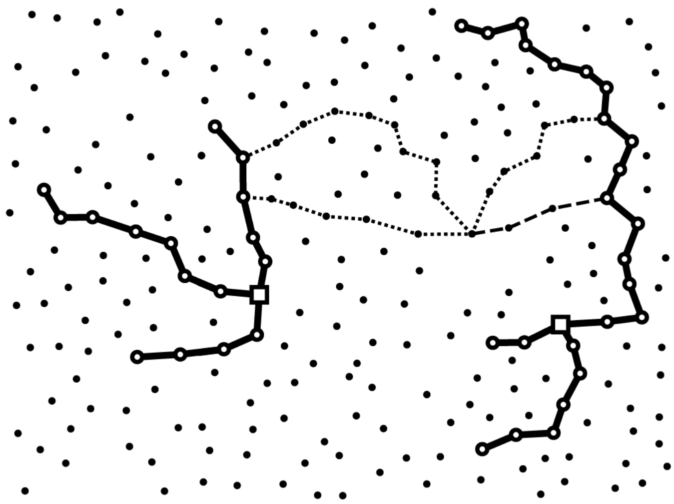
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



35

Process overview

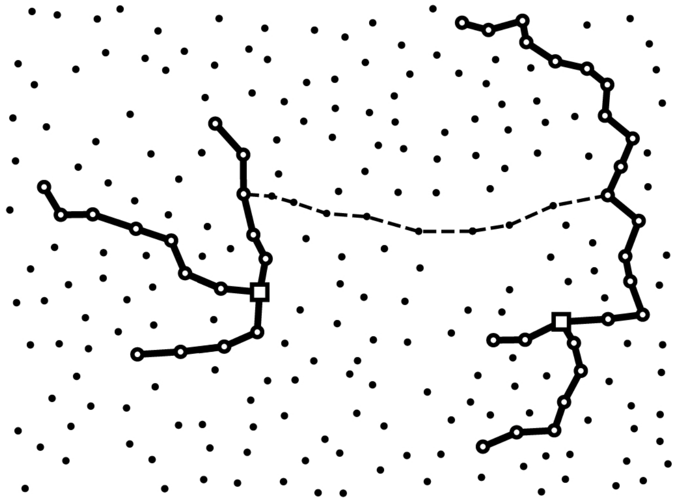
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



36

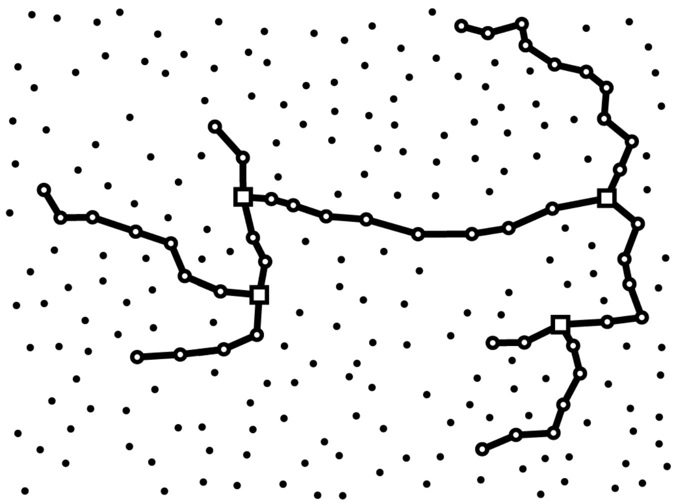
Process overview

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Process overview

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Local data structures and link ratings

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Tables to maintain state information
  - Neighborhood table (nTab)
  - Interconnection table (iTab)
  - Next-hop utilization table (nhuTab)
  - Center distance table (cdTab)
- Link ratings considering link quality and energy

Behavior block 1: exploration

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- On a non-dominating node, initial exploration agent (IEA) starts after

Current time
Random component in [0,1]

$$t_{IEA} = \begin{cases} t_c + random() \cdot t_{IEAmd} & \text{if} \quad random() \leq p_{gIEA} \\ \infty & \text{else} \end{cases}$$

Generation probability

- Sticky behavior
- Probabilistic next-hop decision of IEA, path straightening

Quality of
Neighborhood
Tabu set

$$p(v_x) = \frac{qr(v_x)}{\sum_{v_i \in S_c} qr(v_i)}$$

$$S_c = \begin{cases} (N_1(v) \setminus S_t) \setminus v_p & \text{if} \quad |(N_1(v) \setminus S_t) \setminus v_p| \geq \eta \\ N_1(v) \setminus v_p & \text{if} \quad |N_1(v) \setminus v_p| \geq \eta \\ N_1(v) & \text{else} \end{cases}$$

Next-hop candidates
Current node
Previous hop

Behavior block 1: construction

- Two important parameters
  - IEA walk segment length IEA_{wsl}
  - IEA minimum dominating contact build distance IEA_{mdcbd}
- Path used by IEA is added to dominating set if
  1. Path length is greater or equal IEA_{wsl} or
  2. Visited node is dominating and path length greater or equal IEA_{mdcbd}
- Nodes are added to dominating set by initial construction agent (ICA) using backtracking
- Each dominating node broadcasts its new status to enable sticky behavior (*# of broadcasts = # of dominating nodes*)
- Center information propagation agents (CIPA) spread center information

Behavior block 1: construction, rule 1

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

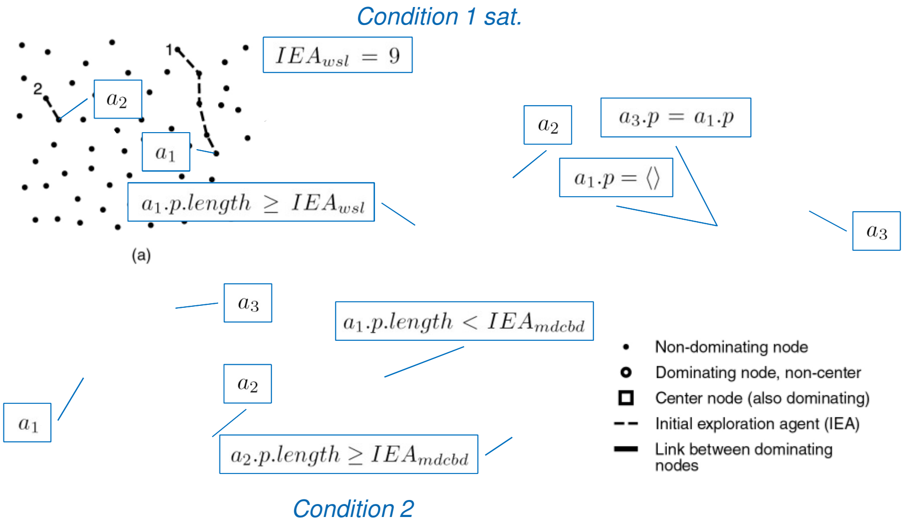
- IEA walk segment length: IEA_{wsl}
- Path used by IEA is added to dominating set if
  1. Path length is greater or equal IEA_{wsl}
- Underlying ideas:
  - Paths need to be divided into sections: arbitrarily long paths cannot be added reliably due to the unreliable medium, since the addition is performed by using backtracking (most efficient method)
  - Since walks must wait for a certain amount of time, when a node has been visited by another walk, the length of IEA_{wsl} has implications on the dominating distance (i.e. it is used to adjust the dominating distance)

Behavior block 1: construction, rule 2

- IEA minimum dominating contact build distance IEA_{mdcbd}
- Path used by IEA is added to dominating set if
  2. Visited node is dominating and path length greater or equal IEA_{mdcbd}
- Underlying idea:
  - No dominating segment should be built across another dominating segment in order to avoid too high densities of dominating nodes, which represent redundancies
  - The choice of IEA_{mdcbd} influences the dominating distance, since it represents the number of hops an IEA walks before being allowed to transform its path to a dominating segment that connects to an already existing dominating segment

Behavior block 1: example

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Outline

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Introduction
  - Wireless Sensor Networks
  - Definition
  - Motivation
- State of the Art
- Proposed Approach
  - Inspiration & Design Considerations
  - Behavior Block I: Initial Dominating Set Construction
  - **Behavior Block II: Transformation to a Connected k-Hop Dominating Set**
  - Properties
- Evaluation
  - Setup
  - Results
- Conclusion

Block 2: exploration and construction

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Preparatory work
  - CEAs keep track of used path, centers known at their generation node
  - This information is deposited at all visited nodes
  - Next-hop utilization information maintained at nodes
- Disjoint fragment *not* locally recognized based on center information
  - Random walk like IEA
  - But: spread next-hop choices more evenly
  - Idea: when feasible connectivity information is not available, use random exploration coupled with a tendency to select new choices
- Disjoint fragment locally recognized based on center information
  - Follow greedily links to closest disconnected fragment
  - Idea: when feasible connectivity information is available, use it to lower cost and create a connection

Block 2: exploration, CEA generation

- Only dominating nodes generate *connectivity exploration agent* (CEA), after and every

$$t_{CEA} = \begin{cases} t_{CEA} \cdot (\beta_1 + \beta_2 \cdot cr(S_c')^{\sigma}) & \text{if} \quad t_{CEA} < t_{CEAmg} \\ \infty & \text{else} \end{cases}$$

Previous value

Max. generation time

Centers known to current node

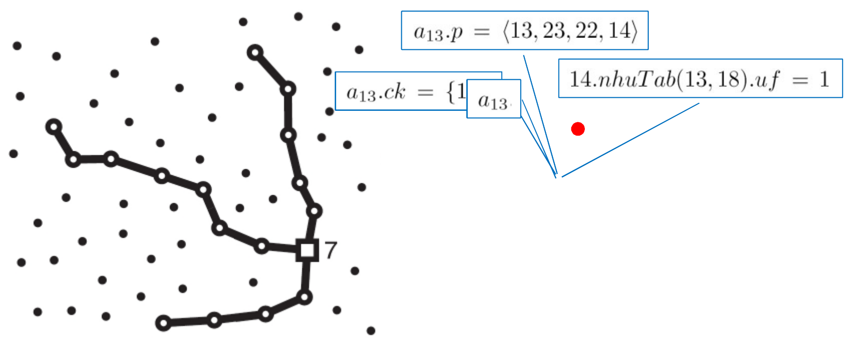
- Connectivity rating

$$cr(S_c') = \sum_{c_i \in S_c'} \max\left(CIPA_{mh} - v_d.cdTab(c_i).d, 0\right)$$

Max. hops of CIPA

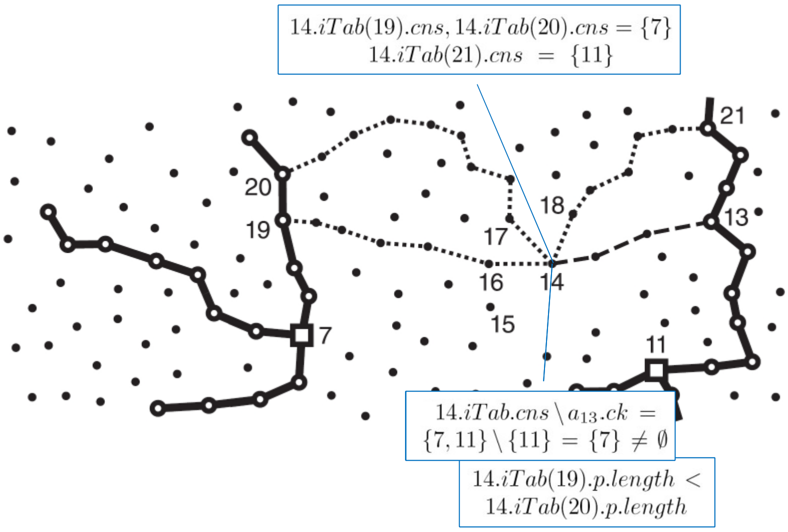
Distance

Block 2: exploration



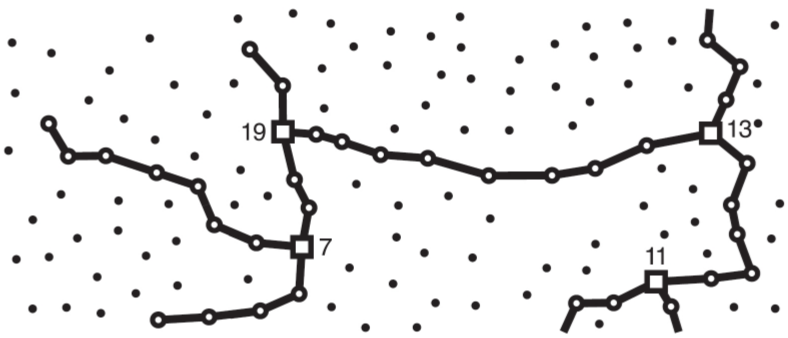
If
$$\begin{aligned} 14.nhuTab(13,15).uf &= 2 \\ 14.nhuTab(13,16).uf &= 3 \\ 14.nhuTab(13,17).uf &= 3 \\ 14.nhuTab(13,18).uf &= 2 \end{aligned}$$
then $ur(13,16) = 1 - \frac{3}{2+3+3+2} = 0.7$

Block 2: exploration (2)



Block 2: construction

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Outline

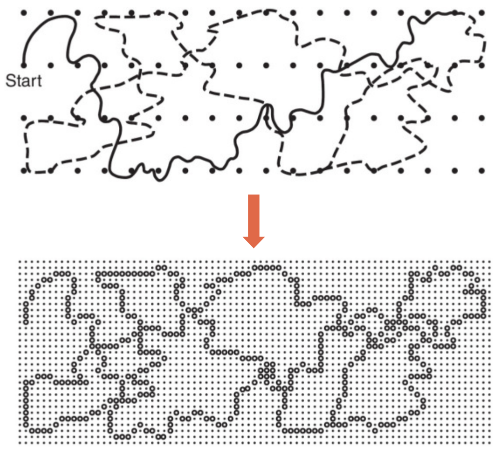
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Introduction
  - Wireless Sensor Networks
  - Definition
  - Motivation
- State of the Art
- **Proposed Approach**
  - Inspiration & Design Considerations
  - Behavior Block I: Initial Dominating Set Construction
  - Behavior Block II: Transformation to a Connected k-Hop Dominating Set
  - **Properties**
- Evaluation
  - Setup
  - Results
- Conclusion

Proposed approach properties (1)

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Similarities to Pieris rapae
  - Operating in a bounded area
  - Agents visit hosts along random walk
  - Random walks are divided into segments
  - No bifurcations along walks
  - No interaction between agents (except via stigmergy)



Proposed approach properties (2)

- Process is self-organizing

> ***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 the system. Moreover, the rules specifying interactions among the system's components are executed using only local information, without reference to the global pattern.*
> *– from "Self-Organization in Biological Systems", by Camazine, Deneubourg, et al.*

- Implications
  1. Self-organization yields a global-level result, the CkDS
  2. The utilization of only lower-level interactions and local information translates to low requirements in terms of information exchange
  3. There is no reference between the execution based on local information at a lower level and the global-level result (e.g. no parameter directly corresponding to k).
- Self-organization is often employed in nature: for example in the Argentine ant to find a structure consisting of short paths between points of interest

Outline

- Introduction
  - Wireless Sensor Networks
  - Definition
  - Motivation
- State of the Art
- Proposed Approach
  - Inspiration & Design Considerations
  - Behavior Block I: Initial Dominating Set Construction
  - Behavior Block II: Transformation to a Connected k-Hop Dominating Set
  - Properties
- **Evaluation**
  - Setup
  - Results
- Conclusion

Evaluation method

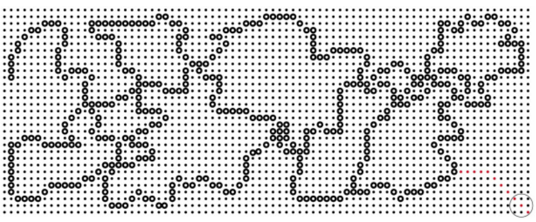
- ShoX network simulator (shox.sourceforge.net)
- Wireless medium, including factors like packet loss and collisions
- Topologies
  - Random placement
  - Rectangular topologies with aspect ratio 1:1.5
  - Number of nodes n=500,1000,1500,2000,2500,3000
  - Average node degree d=7,9,11,13,15,17
- Reference approach: Sausen (2007)
- Series of 50 runs for each parameter combination

Sausen (2007): Sausen, P.; Spohn, M. A.; de Lima, A. M. N. & Perkusich, A. Bounded-distance multi-coverage backbones in wireless sensor networks. *Proceedings of the 2007 ACM Symposium on Applied Computing* (SAC), 2007, 203-208

Important parameters

- Parameter settings
  - Reference approach: $par = r = 2, 3, \ldots, 9$
  - Proposed approach: $par = IEA_{wsl} = 2, 3, \ldots, 37,\ CIPA_{mh} = \lfloor 0.5 \cdot IEA_{wsl} \rfloor$
    $IEA_{mdcbd} = \lfloor 0.7 \cdot IEA_{wsl} \rfloor$
- Basis of comparison: $k' = \max_{v_i \in V}(length(cdsp(v_i)))$

Closest dominating shortest path: Shortest path between v_a and $v_d \in D$ closest to v_a

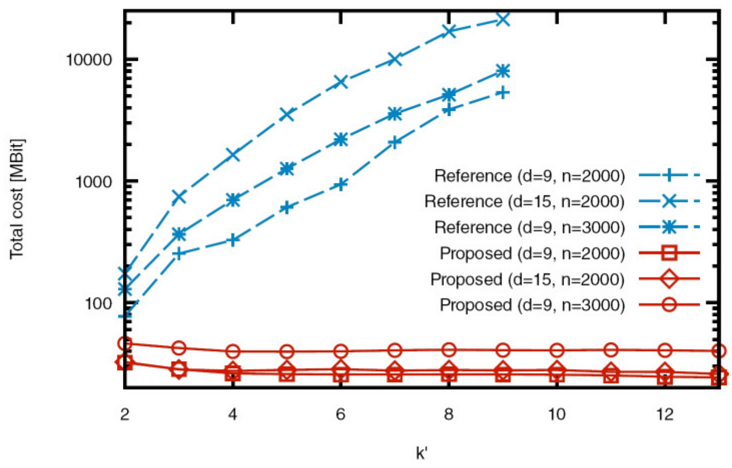


CkDS with k=11 and also k=1000

... but k'=11 and not k'=10 or k'=12

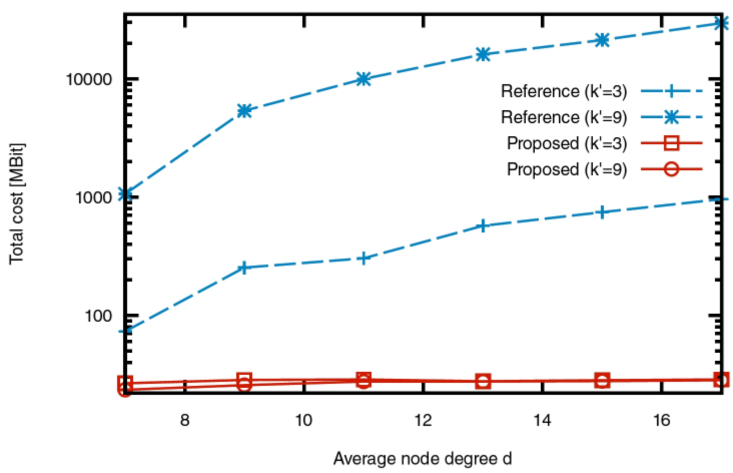
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Cost with varying k'



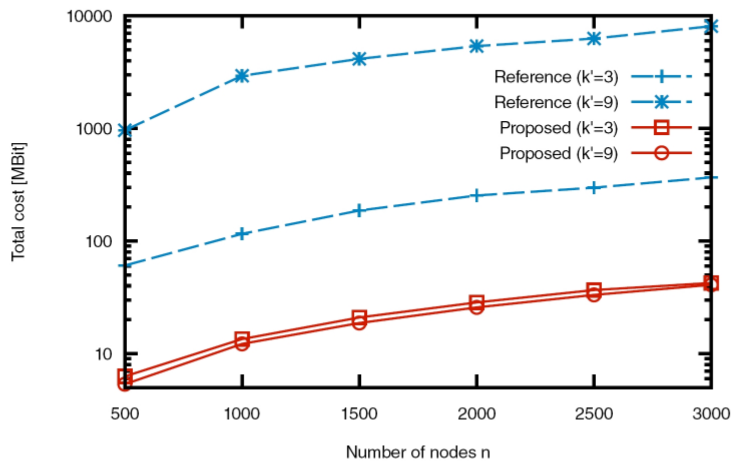
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

Cost with varying d, n=2000



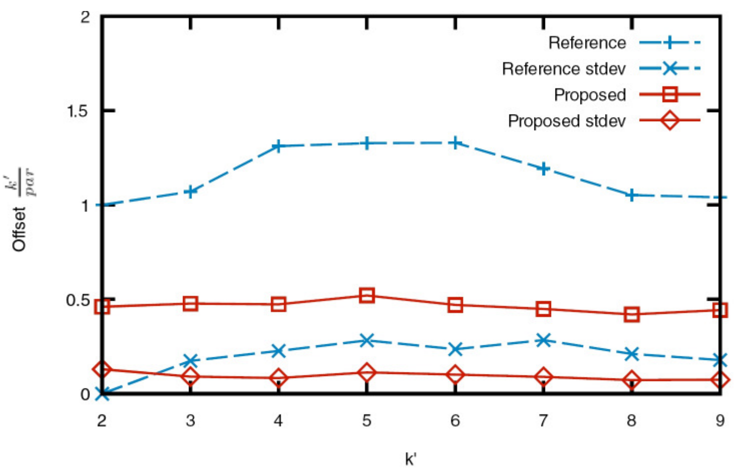
Cost with varying node number n, d=9

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

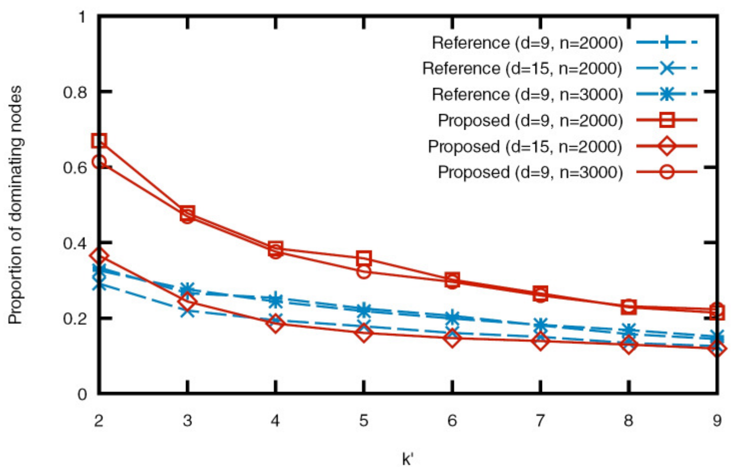


Offset, d=9, n=2000

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

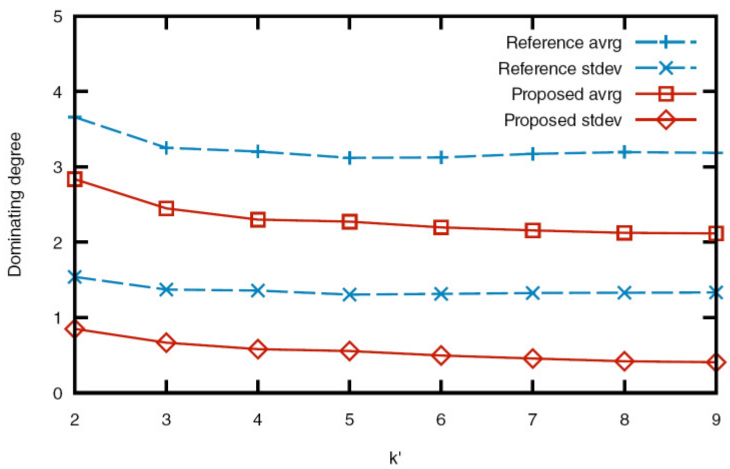


Solution size



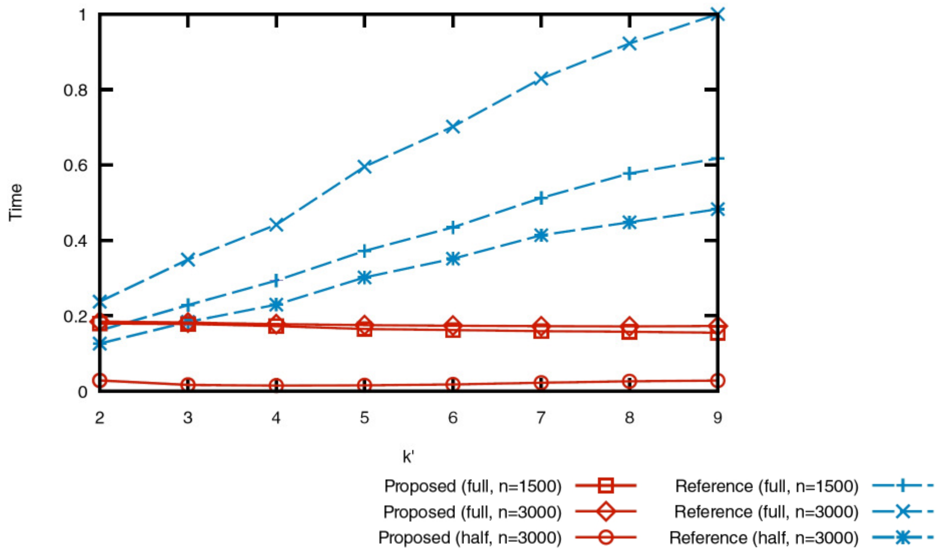
Dominating degree, d=9, n=2000

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



Construction time, d=9

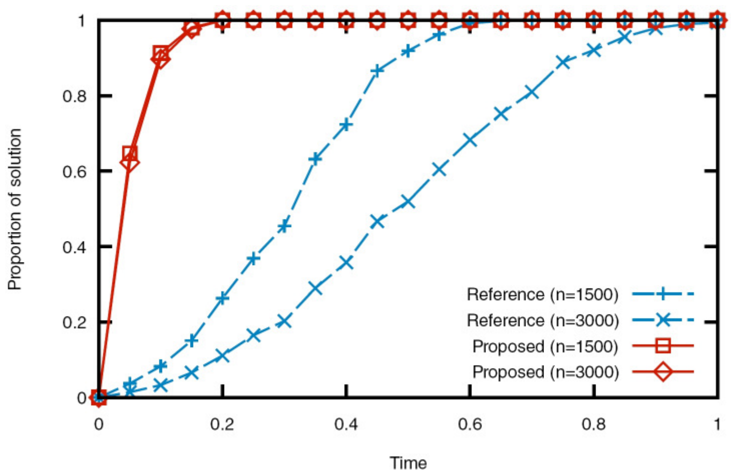
HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



63

Construction progress, k'=9, d=9

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig



64

Outline

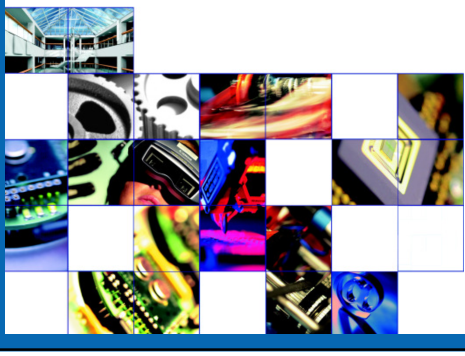
Conclusion

HEINZ NIXDORF INSTITUTE
University of Paderborn
Design of Distributed Embedded Systems
Prof. Dr. Franz J. Rammig

- Self-organizing protocol drawing inspiration from flight behavior of ovipositing Pieris rapae
- Local information used by locally-interacting distributed agents constructing the global solution in a parallel manner
- Main method: random walks
- More efficient, high amount of predictability, low dominating degree
- In contrast to state of the art
  - Cost nearly independent of k' and growing slowly linearly with d
  - Construction time nearly independent of k' and number of nodes

	Communication cost		Construction time	
Parameter	k,k'	d	k,k'	n
Sausen et al.	quadratic	**linear**	linear	linear
Proposed	**independent**	**linear**	**independent**	**independent**

Thank you for your attention



Franz Rammig and Peter Janacik
May 12, 2010

franz@uni-paderborn.de
Heinz Nixdorf Institute
University of Paderborn
Germany

Biologically Inspired Methods for Organizing Distributed Services on Sensor Networks

Tales Heimfarth*, Franz J. Rammig

Heinz Nixdorf Institute
University of Paderborn, Germany

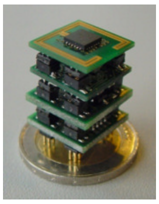
** : Now Univ. Lavras, Lavras (MG), Brazil*

Outline

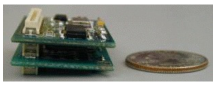
- **Motivation**
  - Wireless sensor Networks
  - System software
- **NanoOS**
  - Requirements
  - Overview of the approach
- **Service Distribution**
  - Problem definition
  - Ant based service distribution
- **Self-organizing cluster construction**
  - Problem definition
  - Clustering “quasi” static networks
  - Clustering dynamic networks
- **Conclusion**

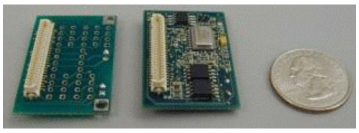
Motivation

- **Wireless Sensor networks enable numerous novel applications**
  - Human-embedded sensing
  - Ocean data monitoring
  - Environmental monitoring
- **Challenges**
  - Self-organization
  - Energy-efficient operation
  - Collaboration
  - In-network processing
  - Ability to deal hardware constrained nodes



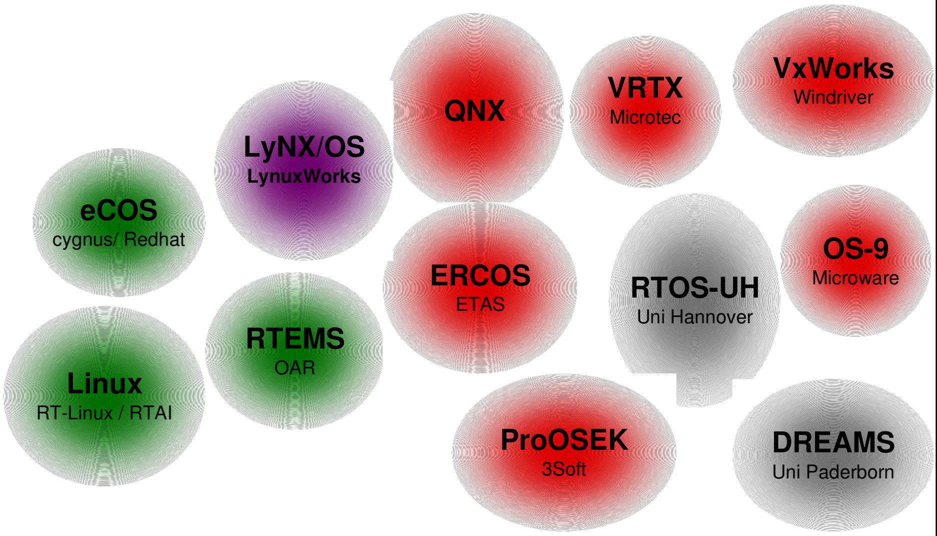
IMEC Sensor Module



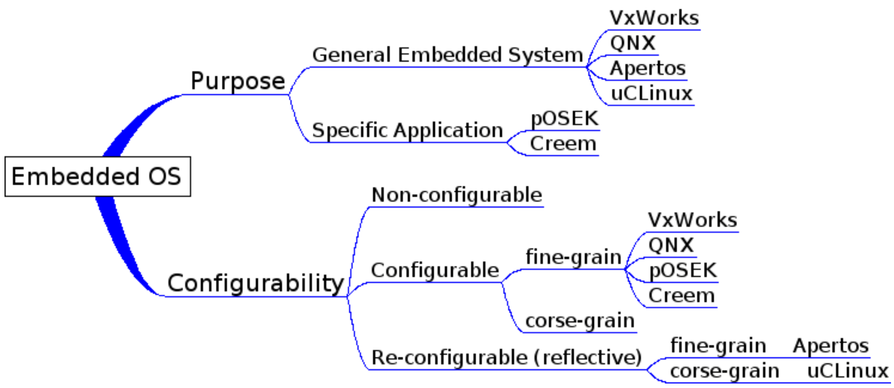


Berkeley Rene and Basic Sensor Board

Numerous Real-Time Operating Systems



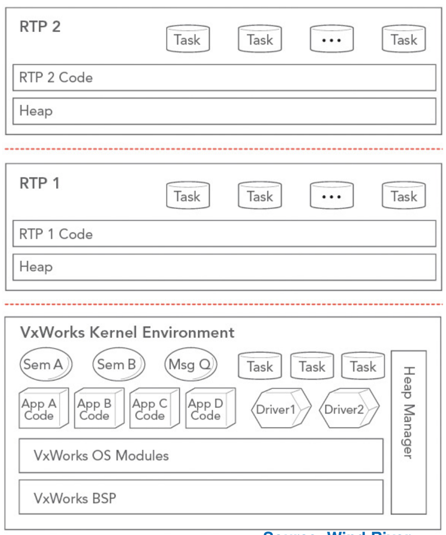
Taxonomy



VxWorks

WIND RIVER

- VxWorks available as different editions for different fields of application
  - General purpose platform
  - Automotive devices
  - Consumer devices
  - Industrial devices
  - Network equipment
  - Safety critical devices
- POSIX
- Support for both
  - MMU-based and
  - MMU-less processors
- Processor Abstraction Layer
- Highly scalable Network Stack
  - IPv4/IPv6 Dual Stack
  - High-Performance routing engine
- Support for shared libraries



Source: Wind River

VxWorks Real-Time

WIND RIVER

- **Priority-based preemptive scheduling**
  - Priorities from 0 to 255
  - RR or Time Slicing per priority level
- **Priority inheritance protocol**
- **Nested Interrupts with interrupt priorities**
- **High speed Interrupt handling**
  - No context switch necessary for ISRs
- **Information are hard to get from WindRiver about internals**

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

QNX

QNX SOFTWARE SYSTEMS

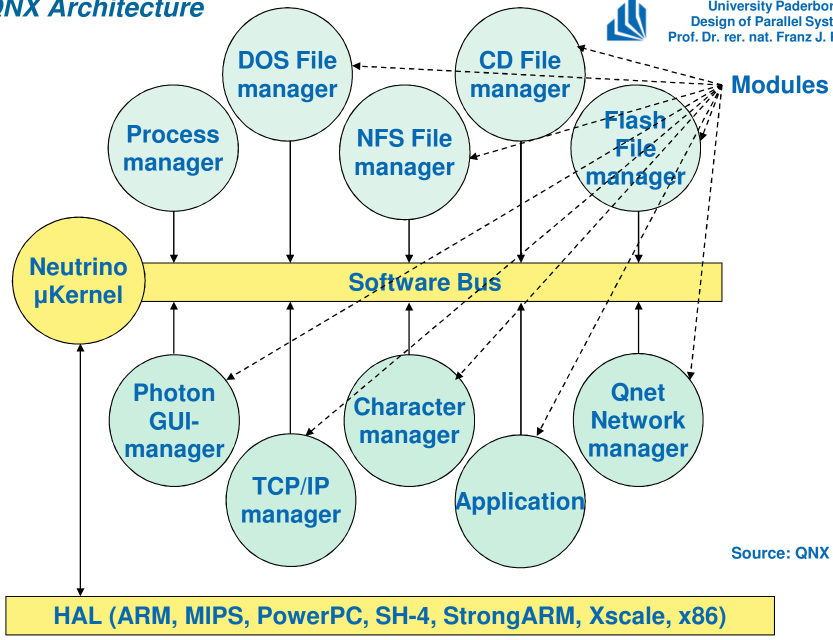
- **Multipurpose RTOS**
  - Footprint 270 KB + Modules
  - Not applicable for deeply embedded systems
  - POSIX, POSIX RT and POSIX Embedded Interface
- **Process isolation by virtual memory**
  - MMU required
- **µKernel services**
  - Threads
  - Signals
  - Synchronization
  - Scheduling
  - Software Bus
    - Message passing interface
    - Inter-node communication possible
    - Distribution of OS Services transparently possible

Source: QNX

© Heinz Nixdorf Institute, University of Paderborn

QNX Architecture

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



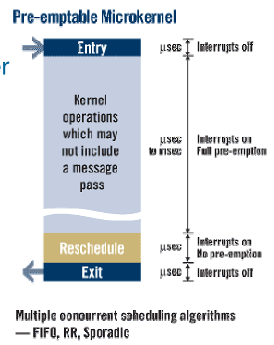
Source: QNX

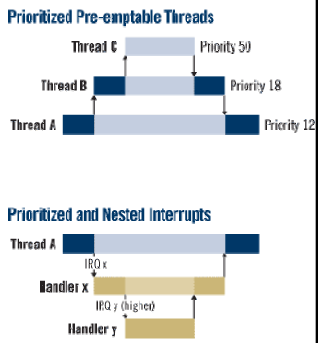
QNX Real-Time Capabilities

QNX SOFTWARE SYSTEMS

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Priority-based preemptive scheduling**
  - Priorities from 0 to 255
  - Every priority class has its own Queue
  - On tie: user-defined per-thread scheduling mechanism (RR, FIFO)
- **Aperiodic scheduling**
  - with sporadic server
- **Nested interrupts**
  - fixed upper bound on interrupt latency
- **Distributed priority inheritance protocol**



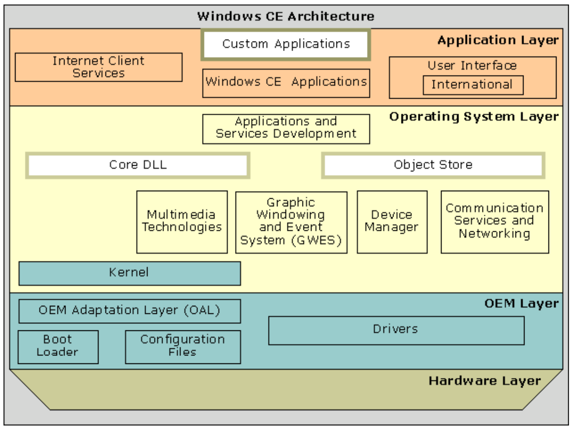


Source: QNX

© Heinz Nixdorf Institute, University of Paderborn

Windows CE

- **Developed especially for small sized portable PCs and PDAs**
- **OEM Layer**
  - OAL abstracts Hardware platform
- **Drivers included in kernel mode**
- **Process isolation by virtual memory**
- **Object store as abstraction of different memory types**
- **API subset of Win32-API**

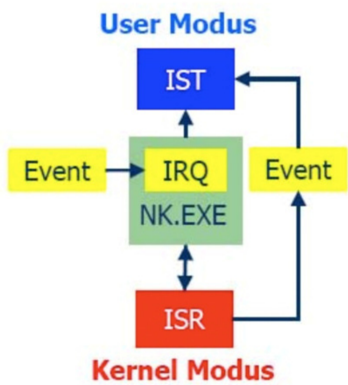


Source: Microsoft

Windows CE Real-Time Capabilities

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

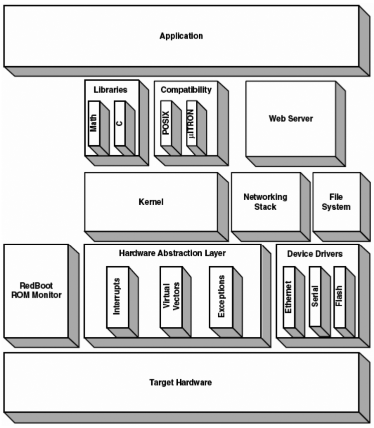
- **Priority-based preemptive scheduling**
  - 24 User-tasks
  - 8 System-tasks
  - 256 Priorities (CE 5.0)
    - RR per priority level
  - Arbitrary number of threads
- **Nested-Interrupts**
  - Interrupt Service Threads (ISTs)
    - Will be notified at end of ISR
    - Useful for drivers that wait for specific events
    - Run at highest priority
- **No Priority Inversion handling**
    - Unbounded waiting time possible



Source: Microsoft

eCOS

- **Open-source RTOS published by RedHat under GPL**
- **Especially designed for deeply embedded systems**
  - 64/128 KB SRAM
  - 512KB Flash
- **Component-based**
  - Fine grained offline configuration
  - Configuration Definition Language (CDL)
  - eCOS Configuration Tool
  - Cross compilation
- **No process isolation**
  - No MMU required



Source: RedHat

eCOS Real-time capabilities

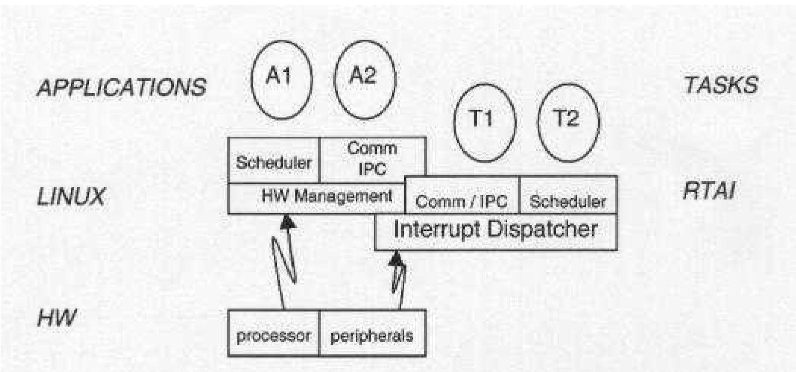
- **Priority-based preemptive scheduling**
  - Bitmap scheduler
    - 32 priorities
    - Only one thread per priority level
  - Multilevel queue scheduler
    - 32 priorities
    - Arbitrary number of threads per priority level
    - RR or LIFO per priority level
- **Priority inheritance protocol**
- **Static priority ceiling protocol**
- **Nested interrupts**

RTAI

- The RTAI plug-in should help Linux to fulfil some real time constraints (few milliseconds deadline, no event loss).
- It is based on a RTHAL: Real Time Hardware Abstraction Layer.
- The HAL exports some Linux data & functions closely related to HW.
- RTAI modifies them to get control over the HW platform. That allows RTAI real time tasks to run concurrently with Linux processes.
- The HAL defines a clear interface between RTAI & Linux.

RTAI Architecture

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Source: Politechnico di Milano

RTAI Architecture

- **RTAI real time entities are simple mono-thread tasks whereas Linux applications are full features (mono or multi threads) processes.**
- **RTAI tasks can be kernel-mode (created inside a module) or user-mode (created by Linux and afterwards stolen by RTAI)**
- **RTAI is basically an interrupt dispatcher. The Intel processor interrupts (0..31) are still managed by Linux. RTAI mainly** traps the peripherals ISA interrupts and if necessary re-routes them to Linux **(e.g.: disk interrupts). It is also able to manage other kind of interrupts.**
- **It supports, like Linux, both SMP (Symmetric Multi Processor) and UP (Uni-Processor) Architecture. From a real time point of view, it is quite similar to RTLinux.**

LynxOS

- **RTOS targeted at the deployment in safety-critical applications**
  - Aircrafts
  - Military facilities
  - Communication networks
- **POSIX, POSIX-RT and POSIX Thread Extensions OS Interface**
- **API compatibility with Linux**
  - Same object file format as Linux (Linux v2.6, glibc v2.3.3)
  - API in integral parts compatible with Linux
  - Commercial Linux products without source code availability can run under LynxOS
- **Process isolation by virtual memory**
  - MMU required
  - Kernel mapped into running process

LynxOS Real-time capabilities

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

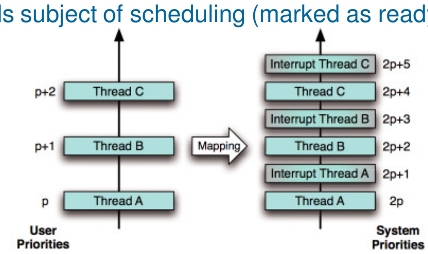
- **Deterministic behavior assured througout all parts of OS**
  - Response time of any service known in advance
- **Priority-based preemptive scheduling**
  - 256 priorities
  - Every priority class has its own Queue
  - On tie: user-defined per-thread scheduling mechanism
    - FIFO
    - HRRN (Aging)
    - RR
    - Priority quantum (user-defined time slices per priority level)
- **Priority inheritance protocol**

Lynx OS Real-time capabilities

Take the LynxOS tour

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Priority tracking**
  - Idea: Interrupts serviced by driver-threads
  - Driver-threads run in kernel context
    - No context switch necessary
  - 512 priorities at all
    - 256 scheduling priorities (Even numbers)
    - 256 interrupt priorities (Odd numbers)
  - Driver priority slightly higher (p+1) than associated application priority
  - Driver-threads subject of scheduling (marked as ready by ISR)

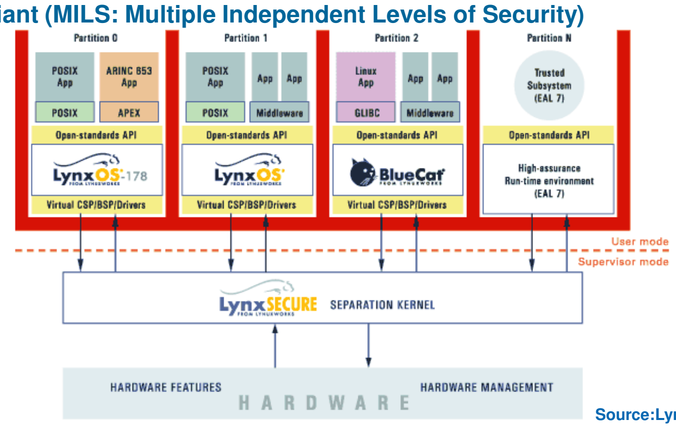


Source:LynxWORKS

© Heinz Nixdorf Institute, University of Paderborn

LynxSecure

- Based on separation µKernel
- EAL-7 security certification
- MILS compliant (MILS: Multiple Independent Levels of Security)

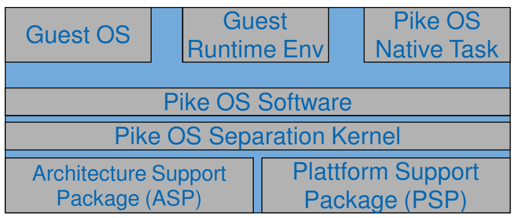


Source:LynxWORKS

PikeOS

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- MILS compliant
- Strict time and resource partitioning
- Combines para-virtualization and hard real-time
- Linux, OSEK, POSIX, Java, Ada, ITRON, CoDeSys. ARINC653 and VxWorks subset available as partition type
- Time- and priority-driven scheduling can be changed on the fly



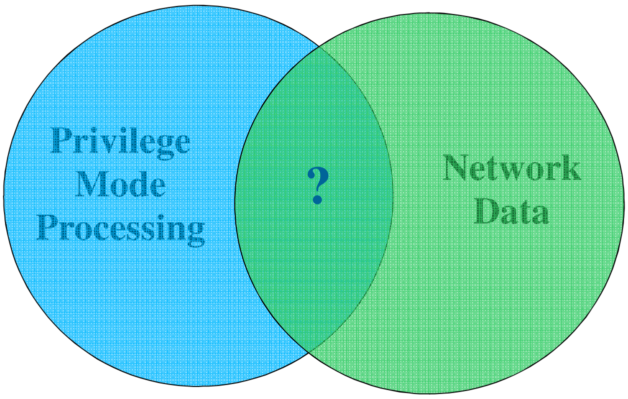
Source: SYSGO

MILS

- MILS: Multiple Independent Levels of Security
- Security Kernel is the only privileged code
- Security Kernel enforces only four very simple security policies
- All other security policy enforcement is divided among middleware and the applications
- Empowers application layer to enforce its own security policies

Source: Objective Interface Systems, Inc

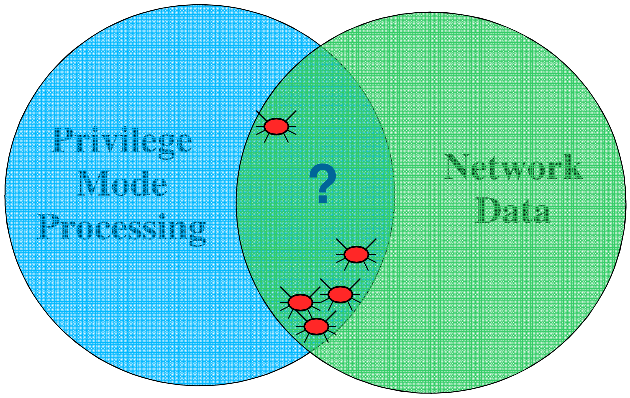
MILS Security policy



What happens when network data is processed in privilege mode?

Source: Objective Interface Systems, Inc

MILS Security policy

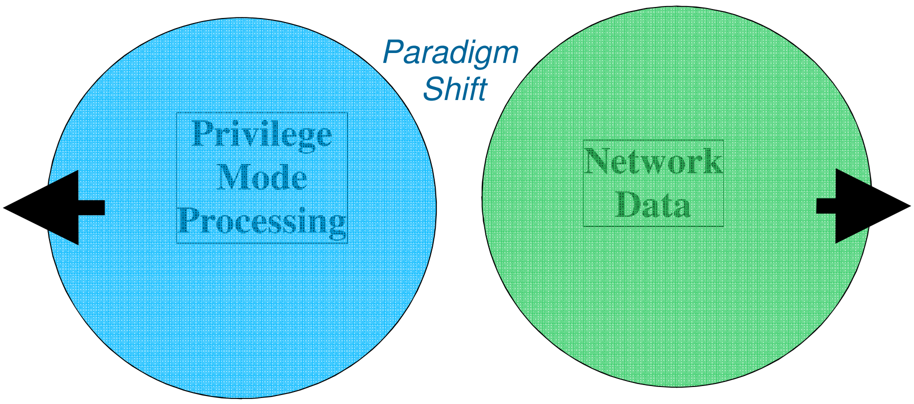


Wild Creatures of the Net: Worms, Virus, . . .

Source: Objective Interface Systems, Inc

MILS Security policy

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Under MILS Network Data and Privilege Mode Processing are Separated

Source: Objective Interface Systems, Inc

© Heinz Nixdorf Institute, University of Paderborn

The MILS Architecture

- **Three distinct layers (John Rushby)**
- **Separation Kernel**
  - Separate process spaces (partitions)
  - Secure transfer of control between partitions
  - Really small: 4K lines of code
- **Middleware**
  - Application component creation
  - Provides secure end-to-end inter-object message flow
    - Device Drivers, File Systems, Network Stacks, CORBA, DDS, Attestation, …
- **Applications**
  - Implement application-specific security functions
    - Firewalls, Cryptomod, Guards, Mapplet Engine, CDS, Multi-Nation Web Server, etc.

Source: Objective Interface Systems, Inc

The MILS Architecture

- **Separation Kernel**
- Microprocessor Based
  - Multi-core Time and Space Multi-threaded partitioning
  - Data isolation
  - Inter-partition communication
  - Periods processing
  - Minimum interrupt servicing
  - Semaphores
    - Synchronization primitives
  - Timers
- ***And nothing else!***

MILS Middleware

- Traditional RTOS services
  - Device drivers
  - File systems
  - Token and trusted path
- Traditional Middleware
  - CORBA (distributed objects)
  - Data distribution (Pub-Sub)
  - Web services
- Partitioning Communication System (PCS)
  - Extend Separation Kernel policy enforcement to distributed systems

Source: Objective Interface Systems, Inc

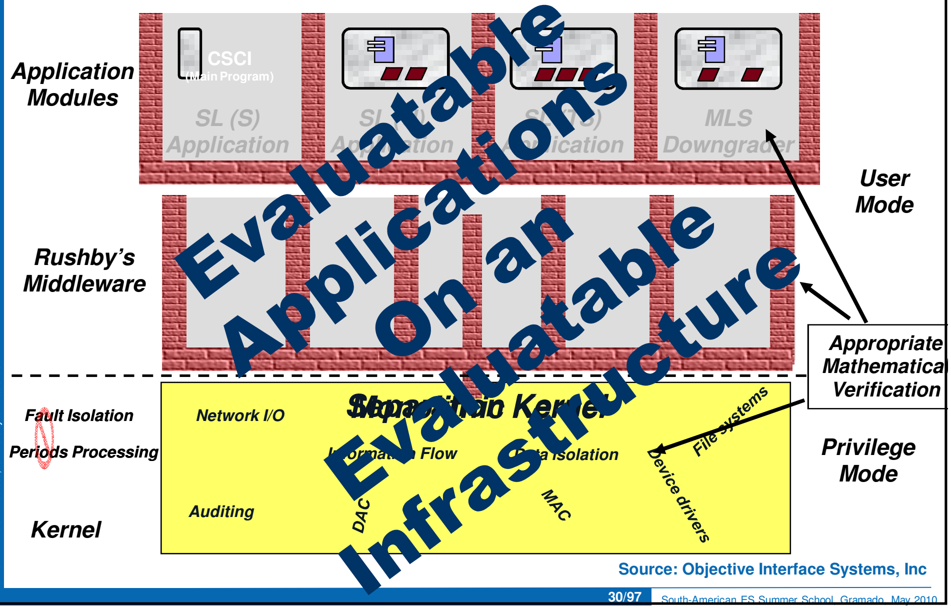
Separation Kernel

- **Where should SK reside?**
  - To be tamper-proof
    - Must be in a separate address space from *any* application code
  - To be non-bypassable
    - Must be part of every input or output service request issued by an application
- **Why keep security functions out of the kernel?**
  - Security functions are often application-specific
  - Any code co-resident with security functions could interfere with those security functions
  - Entire kernel must be analyzed for weaknesses and malicious code
- ***The SK must be the only code that runs in privileged mode***

Source: Objective Interface Systems, Inc

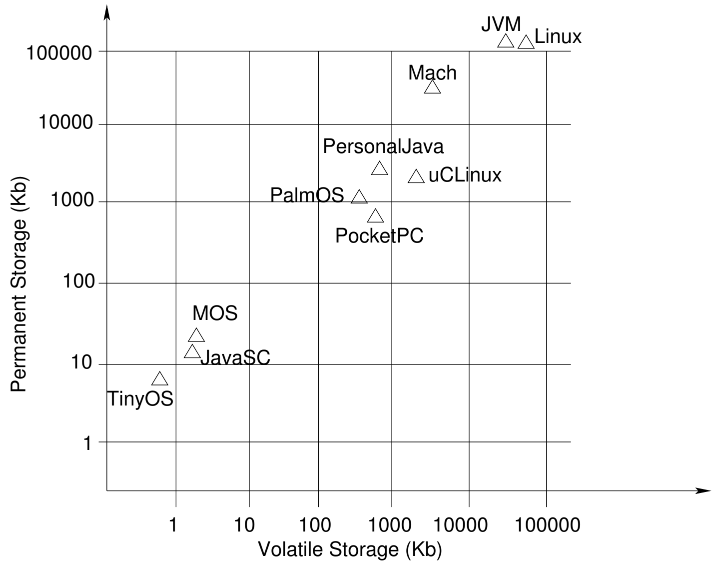
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

MILS Architecture Evolution



Source: Objective Interface Systems, Inc

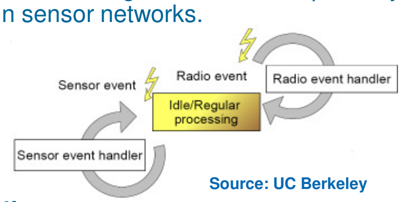
Sensor Operating Systems



TinyOS (UC Berkeley)

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **What is TinyOS?**
  - TinyOS is an open-source operating system designed for wireless embedded sensor networks. It features a component-based architecture which enables rapid innovation and implementation while minimizing code size as required by the severe memory constraints inherent in sensor networks.
- **Extremely small footprint of 178 Bytes**
- **Event-based instead of thread-based**

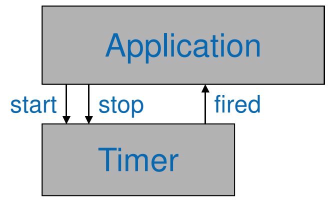


Source: UC Berkeley

- **Paradigm: Separate API from Implementation**
  - Use Interface at higher level, lower level is the implementation
- **Concurrency with Tasks**
  - Tasks are intended to do arbitrary computation, Events and Commands do state transitions
  - Tasks are queued (run until completion), on empty queue, CPU sleeps

TinyOS

- **Bidirectional interface**
  - Commands are specified at higher level and implemented at lower level
  - Events are specified at lower level and implemented at higher level
  - Events are comparable to callback functions
- **Programs are build from**
  - components, that are either
    - modules or
    - configurations

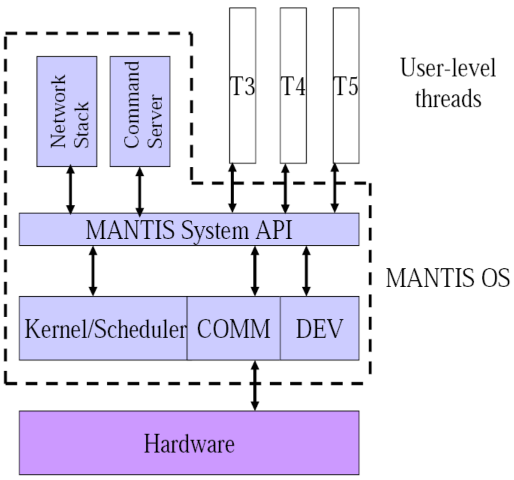


- **Modules implement interfaces with functions (command and events)**
- **Configurations connect interfaces together (wiring)**
- **No real-time components built-in**

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Mantis Operating System (MOS) (University of Colorado)

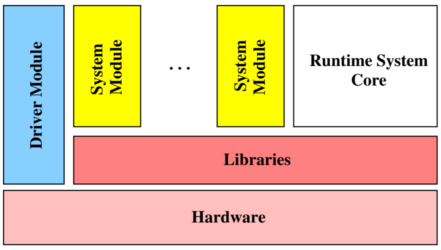
- **Behaviour similar to UNIX**
- **More functionality than TinyOS**
- **Priority-based thread scheduling**
- **Semaphores**
- **All treads in same address space**
- **More resource-intensive than TinyOS**



Source: Univ. of Colorado

Contiki and Cormos

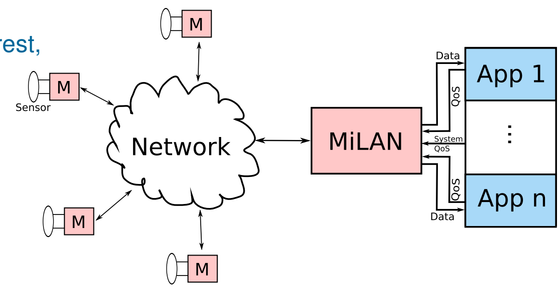
- **Contiki: (Swedisch Institute of Computer Science)**
  - For sensor nodes with limited resources
  - Dynamic loading and unloading of mudules at run-time
  - Footprint between TinyOS and Mantis OS
  - Basically event-driven, but also support for multi-threading
- **Cormos (Communication-Oriented Runtime System for Sensor Networks) (FORTH, Greece)**
  - Communication-centric approach
  - Principal abstractions:
    - Events
    - Handlers, organized in modules
  - Events trigger
    - internal and
    - external actions
  - Handler is executed
    - When event containing this handler is scheduled



Source: FORTH

MiLAN (University of Rochester)

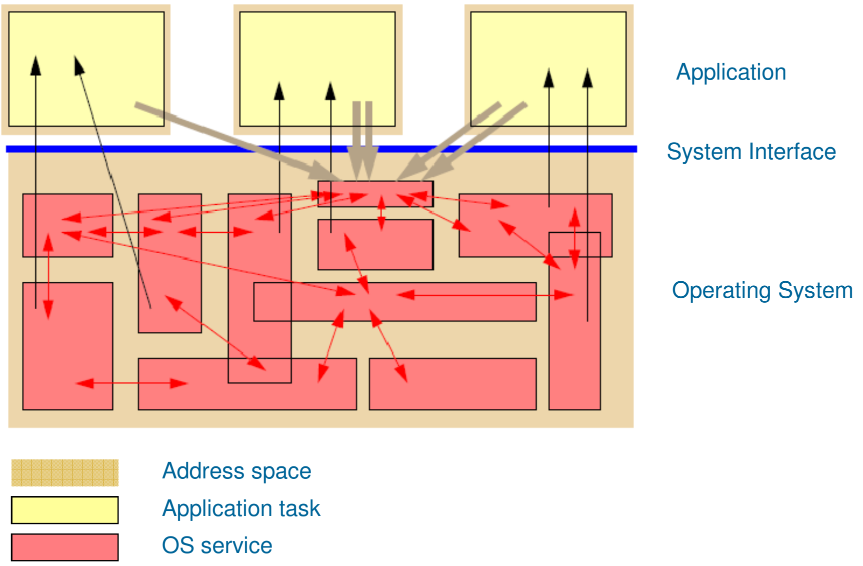
- **Middleware on top of the network stack**
- **Links application requirements**
  - described by
    - Set of variables of interest,
    - Needed QoS
  - to
    - Sensor structure
    - Network architecture
- **Applications in MiLAN are**
  - Data driven and
  - State-based
- **Service discovery protocol to learn**
  - Actual network condition
  - Accessibility of nodes
  - Energy level,…



Source: Univ. of Rochester

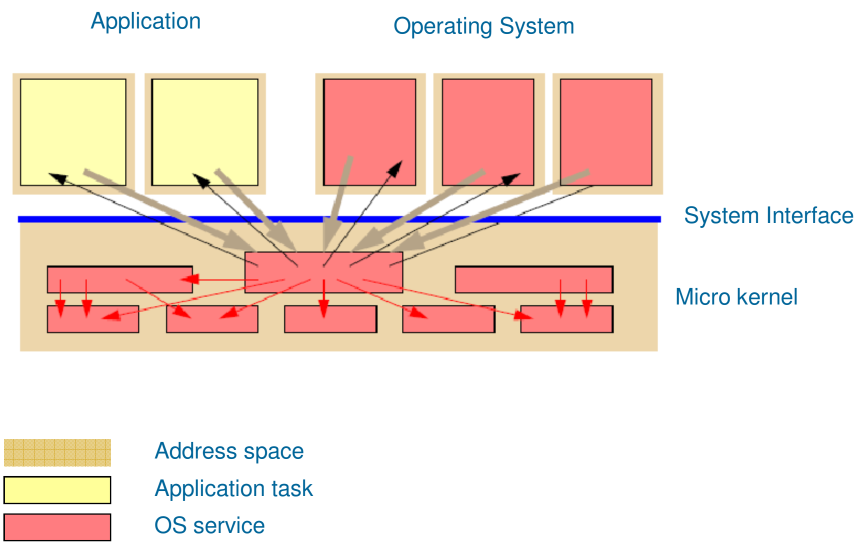
Architectures: Monolithic Approach

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Architectures: Micro Kernel

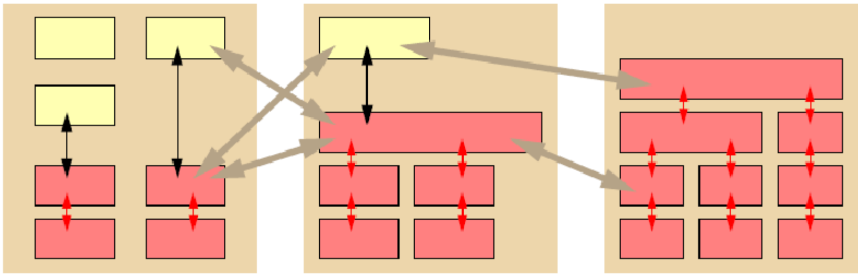
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



© Heinz Nixdorf Institute, University of Paderborn

Architectures: OO Library

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



Address space
Application task
OS service

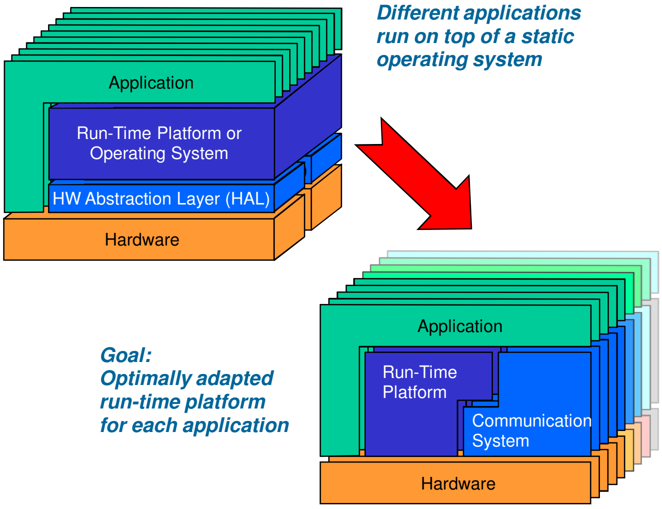
Techniques to reduce OS Resource Requirements

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Preference for library-based architecture:
  - Full flexibility
  - Can be adapted to any needs
- Fine granular customization (approach of ORCOS)
  - Off-line as far as possible
  - Potentially on-line for special cases
- Service distribution over networks of processing nodes (not considered here)
  - Careful clustering and service distribution needed

© Heinz Nixdorf Institute, University of Paderborn

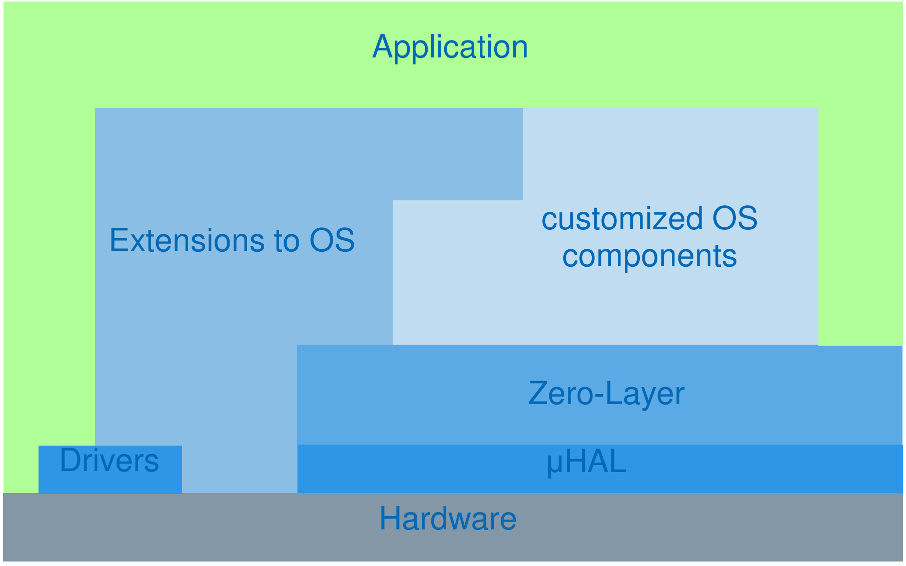
Configurable OS



ORCOS Library-Based OS: Overview

- **Features**
  - Object Oriented Design
  - GNU Development environment C/C++
  - Library Based OS
    - Only one binary for application and OS
  - Customization support library (CSL)
    - Fine grained Source Code Configurability
  - Zero Layer
    - Hardware and OS independent classes to build a framework to support software customization
    - Set of configurable classes that define a minimal OS framework (stack, thread, ...)
    - μHAL (Abstraction of minimal Architecture)
      - Board
      - Processor
      - Memory
    - Footprint (1-4 KBytes)

Architecture



ORCOS vs. TinyOS

Aspect	TinyOS	ORCOS
Architecture	component-based	component-based
Hardware Abstraction	yes	yes
Programming Model	purely event-driven only one stack	single-/multi-threading one or multiple stacks
Programming Language	NesC	ANSI C/C++
Scheduling	non-preemptive FIFO preemption by events	preemptive
Configurability	component “soldering”	offline source code
IPC	active messages	mailboxes
Synchronization	resource interface for arbitration	semaphors, barriers, condition variables
Network Protocols	dissemination and collection	to be determined
Power Management	microcontroller & non-virtualised devices	to be determined

Customization support library

- **Object Oriented Design using C/C++**
  - Simplifies Design
  - Brings overhead (polymorphism, hard-wired class-associations,...)

- **Goals of customization:**
  - Embed knowledge to the system source code which results in a application-specific definition of the concrete class structure
  - Reduce the disadvantages of C++ and fine tune the application

- **Customization:**
  - Optimize the run-time platform for a given application
  - Specialize components by exploiting application-specific knowledge

Customization support library: Levels of customization

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Customization can take place at different levels**

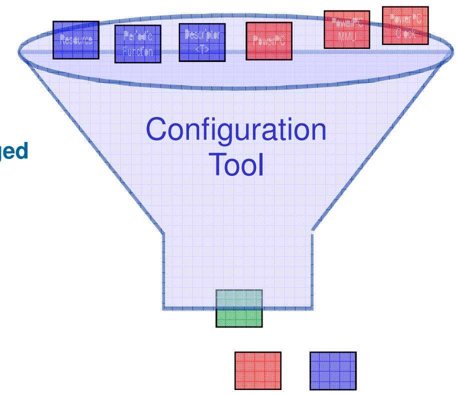
Construct. Level	Involved tool	Example subjects of optimization/customization
Source Code	Preprocessor	Define constants, addresses and data structures depending on hardware and software design
Object-code	Compiler	Eliminate unreachable code fragments, decide about function in-lining
Program	Linker	Decide which object files become part of the executable program
Application	Make tools	Customizable by environment variables or configuration files

- **Configurability of source code automatically leads to further enhancements when using**
  - Compiler with detection of unreachable and superflous code
  - Linker that does not consider unused object code
  - Example: Timerhandler installation when no timer is available

© Heinz Nixdorf Institute, University of Paderborn

Customization Decisions

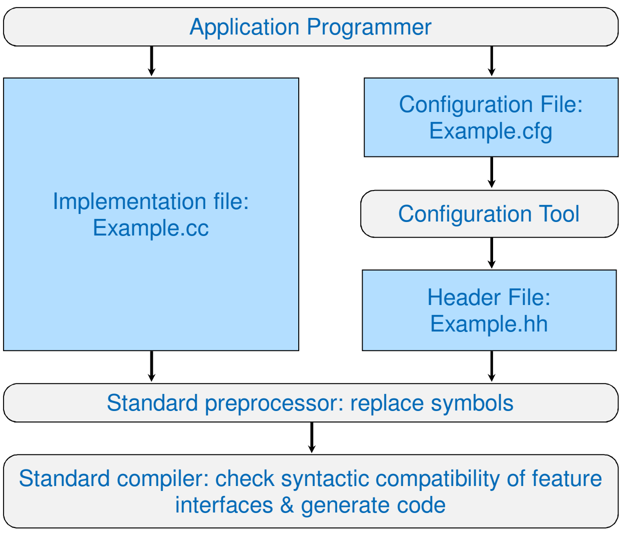
- **System Structure**
  - **Which service classes become**
  - **part of the system?**
- **Polymorphism decisions:**
  - **May the implementation be exchanged at run-time?**
- **Inheritance decisions:**
  - **What special kind of objects is it?**
- **Aggregation decisions:**
  - **Concrete type of the component**
  - **Initialization parameters**



➡ requires Customization Support Library

Customization support library: Design Flow of customizable Objects

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig



From ORCOS to NanoOS

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Ultra low resource devices mean great challenges for OS designers
- They are emerging, solutions urgently needed
- Obvious solution: build ultra slim OS (e.g. TinyOS)
  - con: limited services
- Alternative: High degree of configurability
  - e.g. ORCOS
  - provide exactly the services needed by application
  - ORCOS Open Source under GPL v3

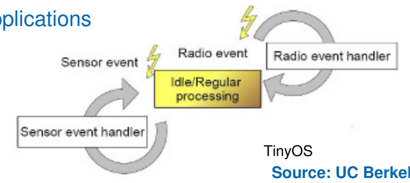
Georg Bahlon's 3D GNU head

- **Now: One step further**
  - Distribution of the OS services over a network of nodes

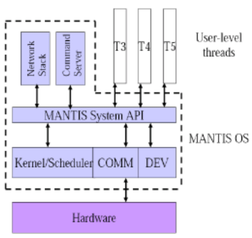
Motivation

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **System Software for Wireless Sensor Networks**
  - Embedded System OS
    - large footprint
    - does not support WSN dynamic and applications
  - Sensor Network OS
    - Support constrained nodes
    - Techniques:
      - Event-based systems
        - e.g. TinyOS, Yatos
      - Multi-threading systems
        - e.g. MantisOS, PeerOS
      - Combination
        - ContikiOS



TinyOS
Source: UC Berkeley

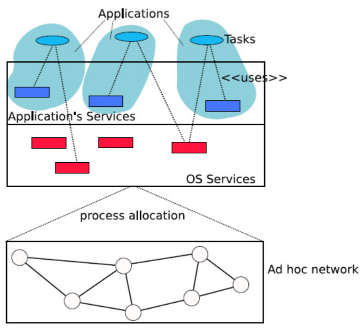


Price: complex programming models, reduced functionality

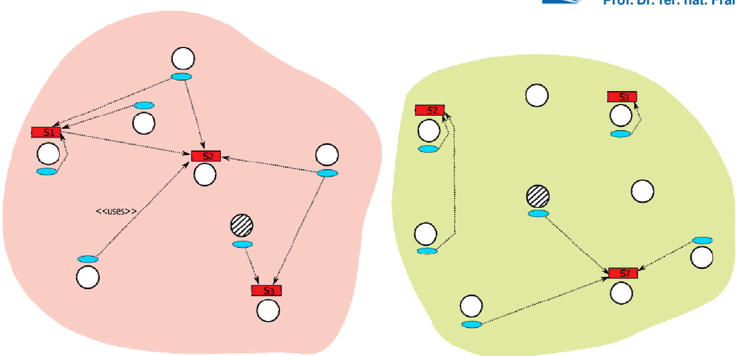
Source: Univ. of Colorado

NanoOS

- **NanoOS**
  - Distributed in-network processing
  - Generic programming model
  - Complex functionality due to distribution
  - Client-server programming model
  - Automatic service migration
  - Self-configuration and self-optimization
- **Some Requirements**
  - Geographically distributed services
  - Small resource utilization
  - Ad hoc networking
  - Self-organization, self-optimization
  - Transparency
  - Scalability
- **Main Ideas:**
  - Applications composed by a set of tasks and application services
  - Uses also OS services
  - Services are distributed among the nodes and used remotely (RMI like)



Overview of the Approach

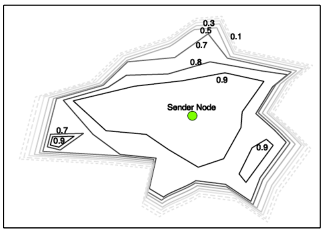


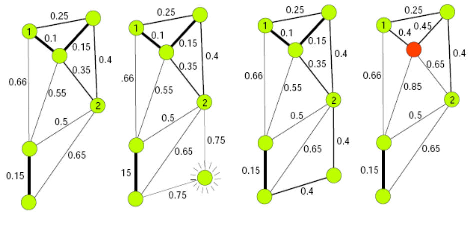
Legend
Task / Requester
Service
Clusterhead

- **Some challenges:**
  - How to measure the communication costs? How to select the best links for communication? **Link metric**
  - Reducing the communication costs: **service distribution**
  - Dealing with very large networks, simplifying service discovery: **clustering**

Combined Link Metric

- **Link metric reflects the “quality” or “usefulness” of a link**
- **Network modeled as a graph G=(V,E) with a link weigh** $w : E \rightarrow [0,1]$
- **Links of wireless network are subject of distortions caused by reflections, diffractions, scattering and doppler fading, resulting in bit errors**
- **Desirable properties of estimative: precision, agility, stability, efficiency**
- **Factors used to measure the link metric:**
  - success rate (precision+,agility-,stability+,efficiency+)
  - received signal strength (precision-,agility+,stability-,efficiency+)
  - history (agility-,stability++)
  - energy reserve





$$M_{combined} = 1 - (k_1 \cdot M_{RSSI} + k_2 \cdot M_{RSR} + k_3 \cdot M_{history} + k_4 \cdot M_{energy})$$

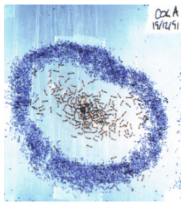
NanoOS: Challenges

- **Focused problems**
  - Dynamic service allocation
  - Clustering
- **Requirements**
  - Low computational cost
  - No global knowledge
  - Local interactions
  - Good performance
  - Robustness
  - Scalability

How to develop heuristic that copes with such requirements?

Self-Organization (in biological and natural systems)

Ripple on the superficie of sand dunes



Leptothor-ants nest behind glas

Spiral Waves (Belousov-Zhabotinski- Reaction)

Butterfly wing pattern

Fungus-Garden of Leaves-cutter ant

Self-organisation in biological systems

- **Self-organisation** in biological systems: Process in which
  - pattern at the *global* level **emerges** solely from
  - *numerous* interactions among *lower-level* components
    - using only *local* information
    - *without* reference to the global pattern.

(Camazine et al.*)

Global, desired result
Less complexity and communication cost
More robustness

**S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. Self-Organization in Biological Systems. Princeton Studies in Complexity. Princeton University Press, first edition, 2003*

Self-Organization

What are the components of self-organizing systems?

- Absence of templates, plans
- A large number of interacting subunits
- Staring point: homogeneous, randomly distributed field
- Positive feedback (local activation or attraction, autocatalysis)
- Negative feedback (long-range inhibition, depletion, decay)
- Probabilistic techniques

Self-Organization Properties:

Property	Description
No Central Control	There is no global control system or global information available. Each subsystem must perform completly autonomous
Emerging structures	The global behavior emerges in form of observable pattern or structure
Resulting complexity	Even if the individual subsystems can be simple and use basic rules, the resulting overall system becomes complex
High scalability	There is no notable performance degradation if more subsystems are added.

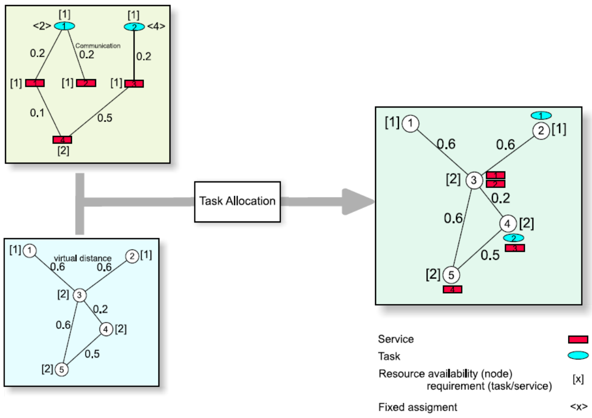
Source: Falko Dressler, Self-Organization in Sensor and Actor Networks

Service Distribution

- **Responsible to allocate a node to a given service**
- **Dynamic re-allocation possible**
- **Objective: reduce the communication overhead**
  - Used bandwidth
  - Size of the path (measured by the link metric)
- **Existing approaches for WSN:**
  - Script population specification (Sensorware)
  - Automatic object placement (MagnetOS)
  - Query Optimizer (TinyDB)
  - Attribute Matching (SINA)
  - Automatic fusion point placement (DFUSE)
- **Not able to cope with our generic service distribution architecture**

Service Distribution

Problem Definition



$$\text{cost} = \sum_{\{m_1, m_2\} \in C} b(\{m_1, m_2\}) \cdot D(q(m_1), q(m_2))$$

Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Policy of Migration Mechanism:**
  - **Transfer policy**: services are like agent and decide by themselves the moment of starting a migration. Potential targets are nodes with enough resources
  - **Selection policy**: locally made, each service decides by itself whether it should migrate. Based on the current communication overhead and the time of the last migration.
  - **Location policy**: We tackle this part with our heuristic
  - **Information policy**: Use stigmergetic communication and exploration packet.

- **Basic Heuristic**
  - **Exploration phase** – exploration packet visits and evaluates candidates
  - **Settlement phase** – information gather used to decide new service location

© Heinz Nixdorf Institute, University of Paderborn

Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Initial situation:** many tasks in the system are communicating with services with an initial assignment
- A single path routing protocol is assumed
- Possible constraints of the migration: k-hops.

- **Using an analogy with the ant foraging behaviour:**
  - In our approach services are the equivalent of food sources
  - Service locations are the equivalent of shortest paths
  - Calls made by the requesters are the ants
  - Requesters are the nests
  - Wireless links form the paths which the ants can use for movement
  - While the requests are being routed to the destination service, they leave pheromone on the nodes.

Basic Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

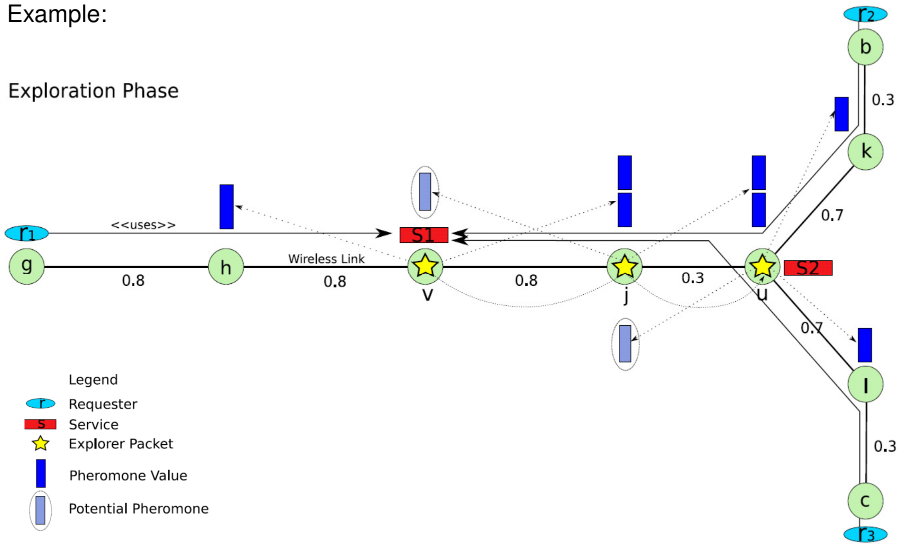
- **Exploration phase:** The exploration packet follows the pheromone trail with highest value and collects a list of possible candidates to host the service s.

- **Settlement phase:** Responsible to proof whether there is enough resource in the candidate node to host service s.
  - The collected list of nodes is analysed in reverse order.
  - Let u be the currently analysed node. If it has enough resources, it is selected to host s.
  - Otherwise, the best connected neighbour with enough resource is selected.
  - In the case that it has a worse link metric than the next node in the list, the process starts again with the next node.

© Heinz Nixdorf Institute, University of Paderborn

Example:

Exploration Phase

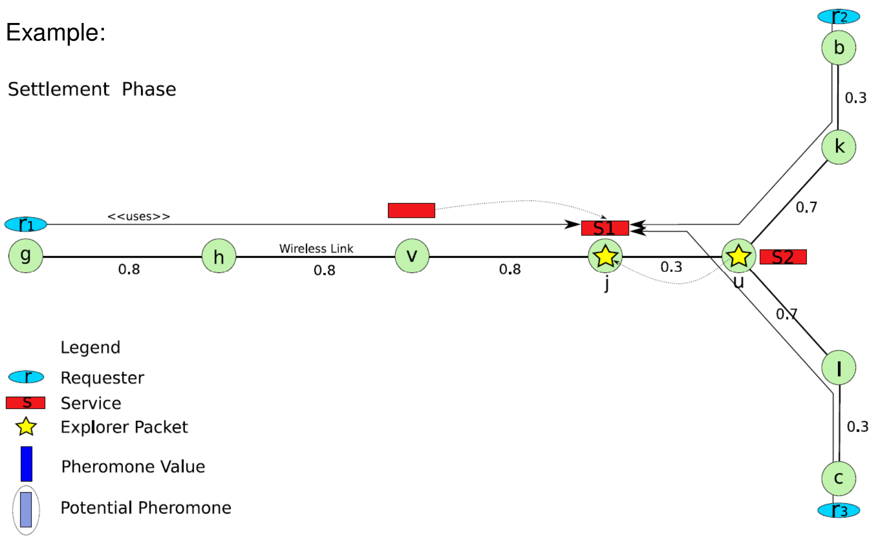


Basic Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Example:

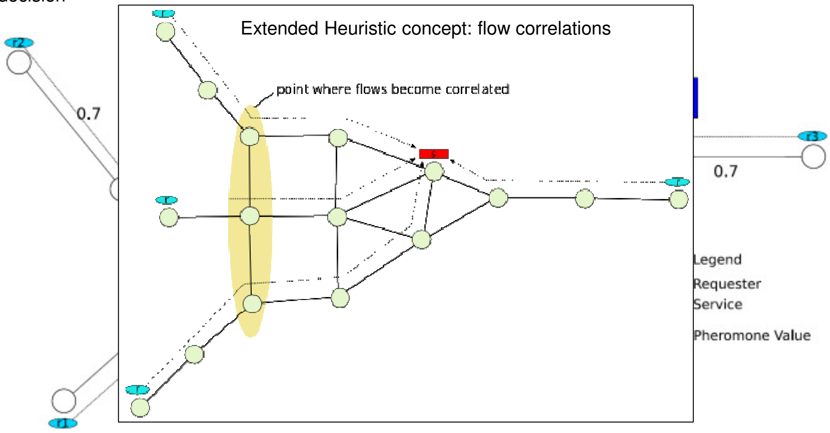
Settlement Phase



Extended Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Different flows coming from nearby regions of the network travel using different paths: wrong greedy decision



Extended Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

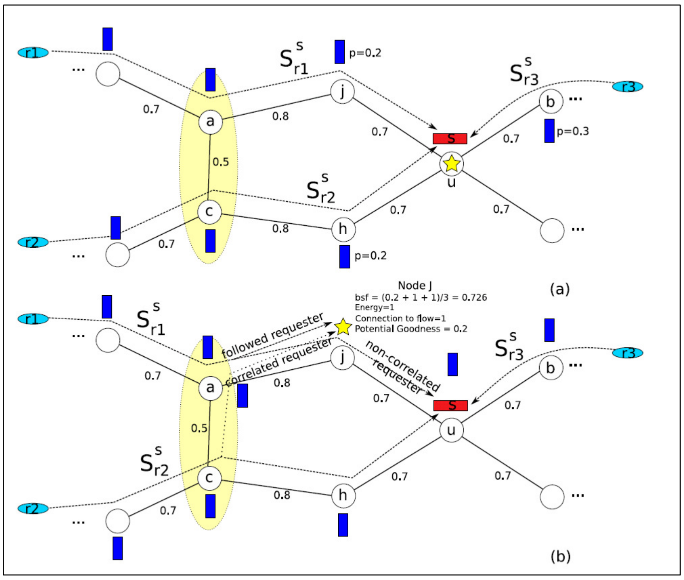
- **Exploration Phase**
  - Similar to the basic heuristic
  - Difference: correlated flows
    - Uses the information that the graph is embedded in a Euclidean plan
    - Neighbouring flows recognized as correlated
    - Service migrates to the “direction” of the network from where the highest amount of requests is coming
    - Correlating flows necessary because nodes does not have any information about their position
- **Settlement Phase**
  - Similar to the basic heuristic
  - Differences:
    - Potential goodness used to measure how central is the candidate node and how intensively the service is used.
    - Energy used to calculate the settlement fitness

© Heinz Nixdorf Institute, University of Paderborn

Extended Ant Based Service Distribution

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

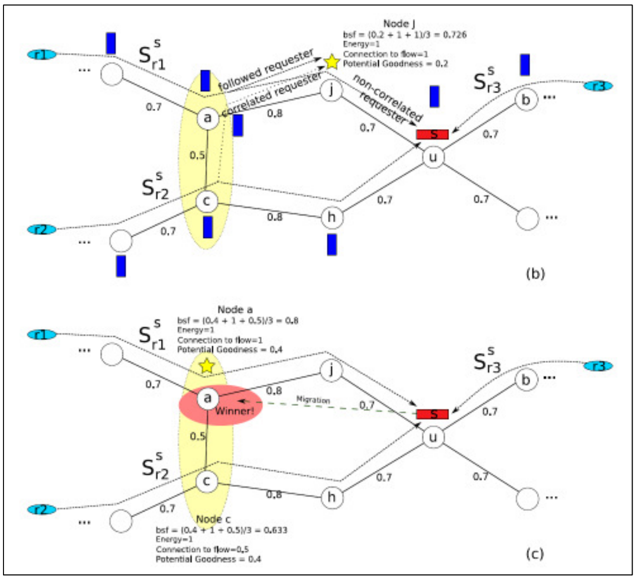
Example:



Extended Ant Based Service Distribution

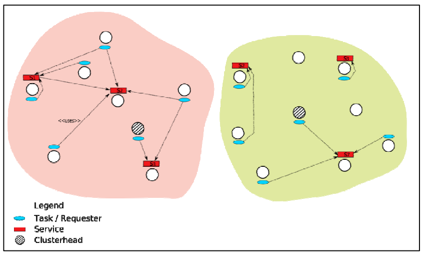
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Example:



Self-Organizing Cluster Construction

- **Clustering = identify a subset of nodes and vest them with extra responsibility of being a local leader (Clusterhead)**
- **In our approach:**
  - A complete instance of OS and application services is found inside each cluster
  - Reduction of the organization overhead
  - Algorithms based on the created hierarchy scales
  - Dependencies among different modules are constrained within the cluster



Clustering: partition the nodes of a graph in subsets $\cup_{i=1,\ldots,n} V_i = V$

Minimum intra communication-cost clustering

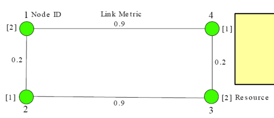
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Objective: multi-hop clusters with enough resource for OS and application**
- **Additional objective: Minimization of the intra-cluster communication cost**
  - Optimization Problem
  - Input: graph with weighted nodes and links (G,w,r) and resource requirement q
  - Ouput: minimization of the intra communication cost given by:

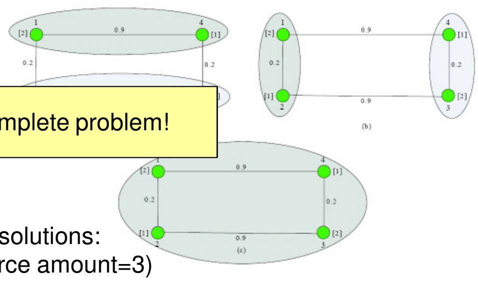
$$cost(C_k,(G,w,r,q)) = \sum_{i=1}^{nk} \sum_{u,v\in c_{ki}} \frac{1}{2} \cdot D_{c_{ki}}(u,v) \cdot (\alpha \cdot r(u) + (1-\alpha))$$

  - Constraints: no overlapping, connectivity, minimum amount of resource per cluster, complete partitioning

Example:



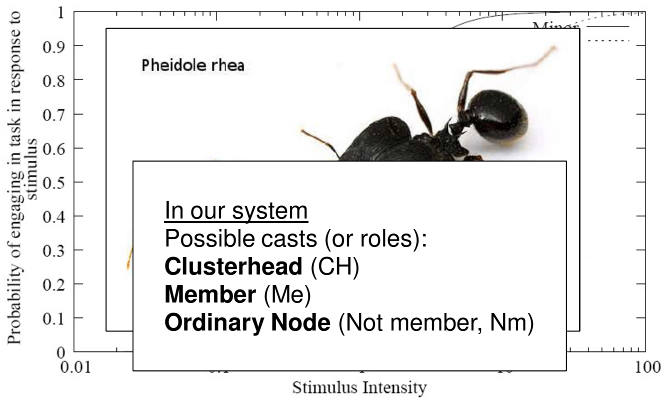
NP-complete problem!



Set of feasible solutions:
(minimal resource amount=3)

© Heinz Nixdorf Institute, University of Paderborn

Division of Labor and Task Allocation in Social Insects



In our system
Possible casts (or roles):
Clusterhead (CH)
Member (Me)
Ordinary Node (Not member, Nm)

Response function: $T_{\theta_a} = \dfrac{s_a^\beta}{s_a^\beta + \theta_a^\beta}$

First heuristic

Clustering “quasi-static” networks

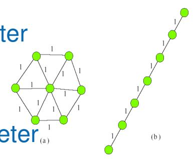
Clusterhead Selection

- **Preferable nodes with good connections and plenty energy**
  - Clusterhead has extra burden (administrative tasks, cluster organization, etc.)
  - Good connectivity: greedy assumption for small cluster cost
- **Threshold in the response function:**
  - A combined metric with: number of nonmember neighbours, connectivity to nonmembers, energy
- **Stimulus:**
  - Combined metric: elapsed time, number of clusters in the vicinity
  - Main idea: nodes with a long time not belonging to any cluster and without any cluster in the vicinity = higher stimulus

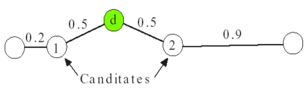
Membership Selection

- **It uses a Fitness function with the following parameters:**

- Distance to the cluster
  - Measured by the distance to the closest node in the cluster
  - Helps to reduce the cluster size



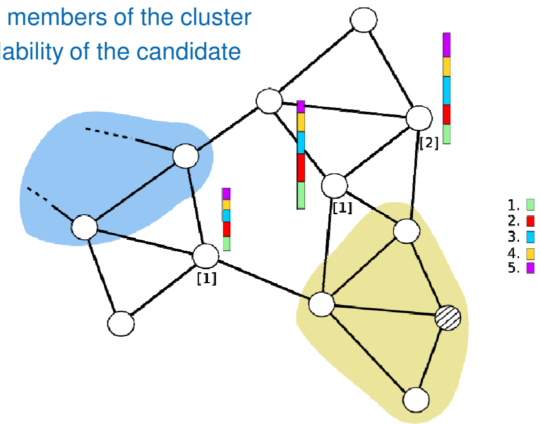
- Distance to the clusterhead
  - Responsible to shape the cluster and constraint its diameter
- Connectivity to nonmembers
  - Improve the potential future members
  - Just important in initial and intermediate phase of the clustering



- Connectivity to members
  - High probability that these connections will reduce the overhead cluster cost
- Resource availability
  - Nodes with plenty of resource are preferable in the initial phase, afterwards, best fit is suitable

Membership Selection

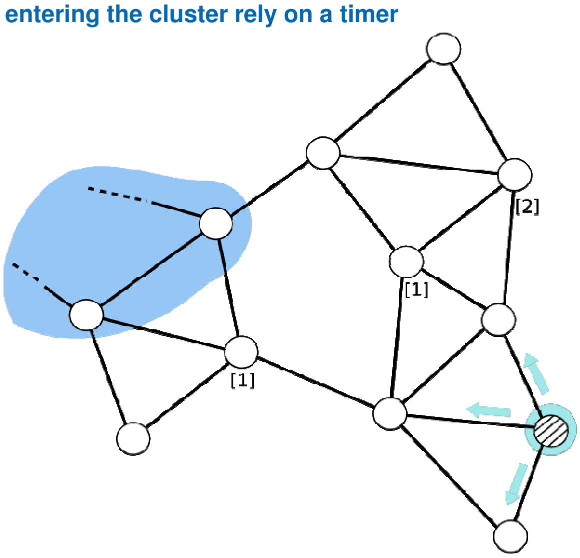
- **It uses a Fitness function with the following parameters:**
  1. Distance to the closest node already in the cluster
  2. Distance to the clusterhead
  3. Connectivity to nonmembers
  4. Connectivity to members of the cluster
  5. Resource availability of the candidate



Membership Selection

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

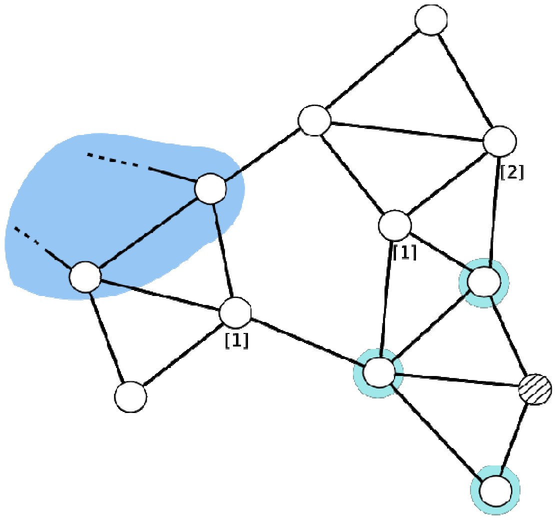
- **Priority of entering the cluster rely on a timer**



Membership Selection

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

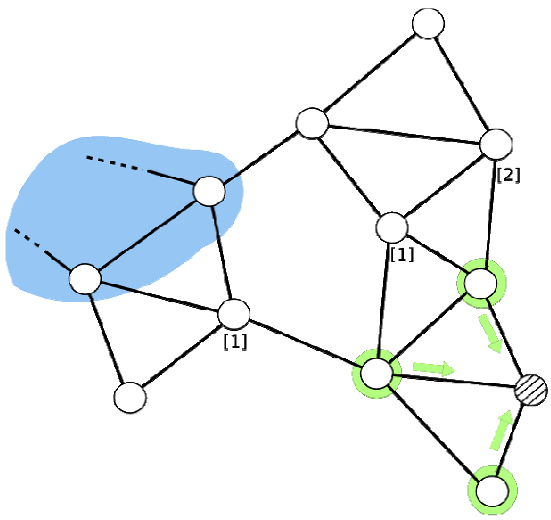
- Priority of entering the cluster rely on a timer



Membership Selection

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

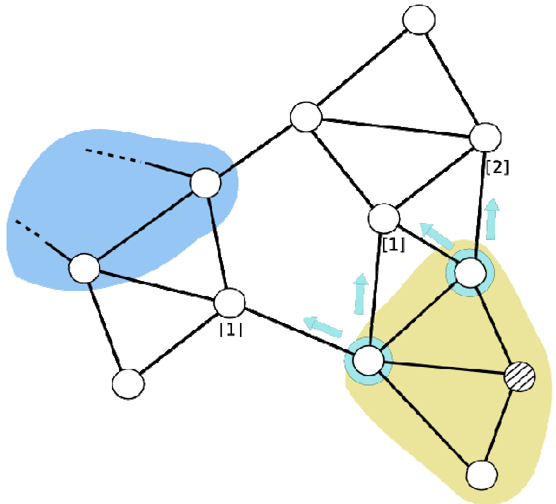
- Priority of entering the cluster rely on a timer



Membership Selection

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

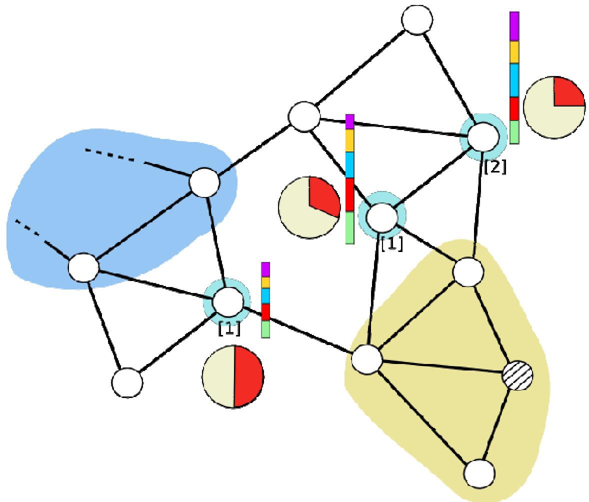
- Priority of entering the cluster rely on a timer



Membership Selection

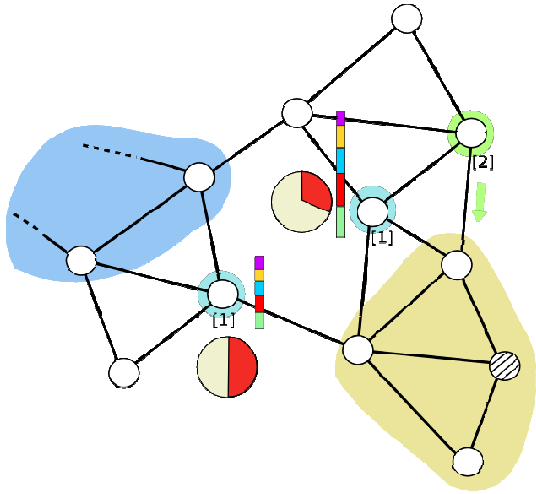
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- Priority of entering the cluster rely on a timer



HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

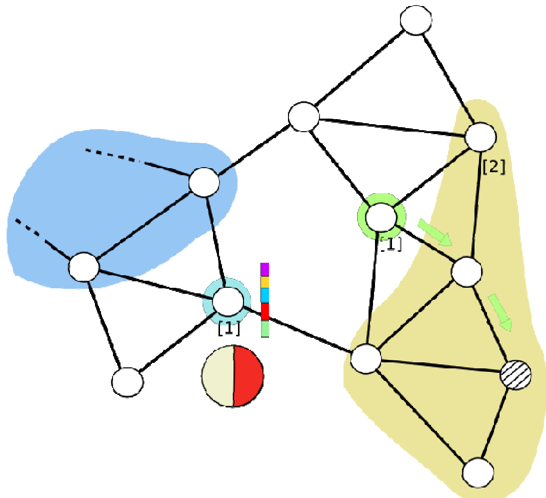
- Priority of entering the cluster rely on a timer



Membership Selection

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

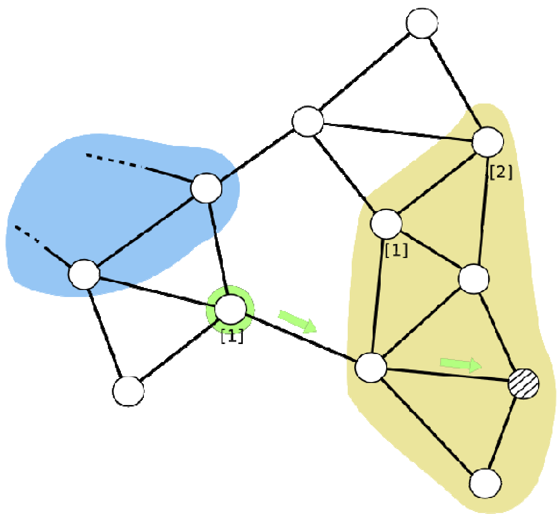
- Priority of entering the cluster rely on a timer



© Heinz Nixdorf Institute, University of Paderborn

Membership Selection

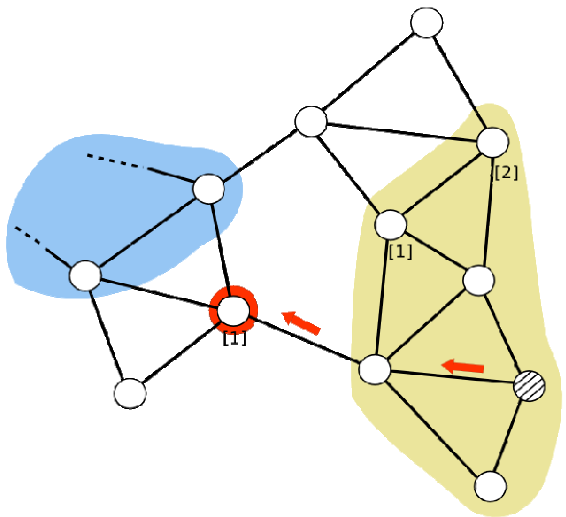
§ Priority of entering the cluster rely on a timer



HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

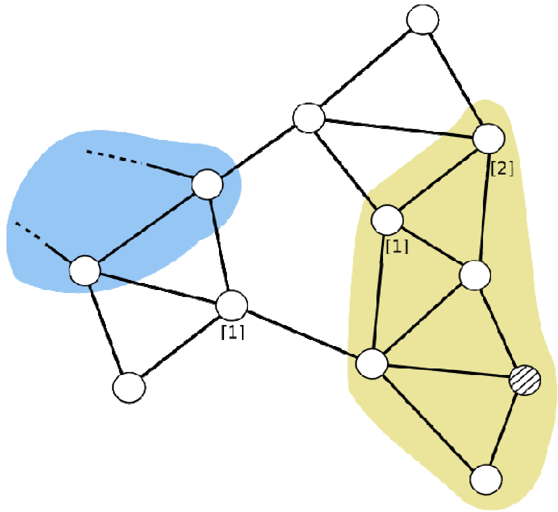
Membership Selection

§ Priority of entering the cluster rely on a timer



Membership Selection

- Complete Cluster



Second heuristic

Clustering dynamic networks (moderate topology changes)

Clustering Dynamic Networks with Positive and Negative Feedback

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

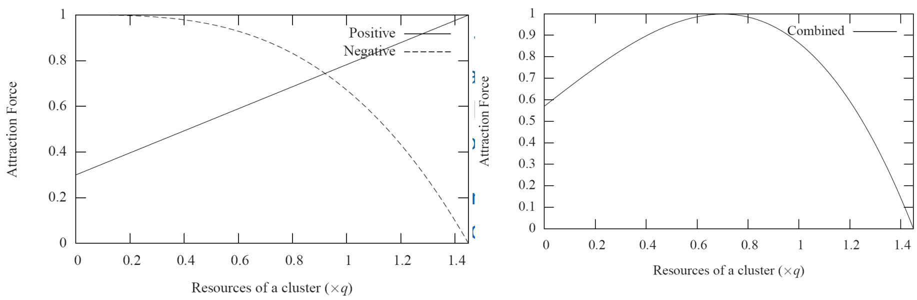
- **Positive feedback: promotes changes, amplification of fluctuations**
  - Lepomis macrochirus (bluegill) nest in large colonies because it provides benefits: detection of predators, reduced exposed area
    - Behavioral rule: I nest where other nest
- **Negative feedback: controls and shaping the system in a particular pattern.**
  - Two kinds of negative feedback can be found in the bluegills nesting process:
    - Avoid overcrowded areas, "I nest where other nest, unless the area is overcrowded"
    - Short range negative feedback: "Keep away, do not nest where I am nesting"

Source: http://fishandwildlife.mnr.gov.on.ca

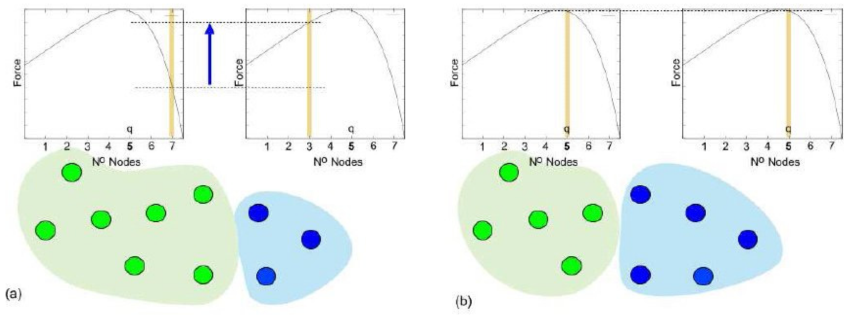
Clustering Dynamic Ad hoc Networks

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

- **Each cluster has a attraction force**
- **Response functions control the complete role exchange in the system**
    - Nonmember to Clusterhead
    - Nonmember, Member of x to Member of y (recruitment function)
      - Threshold: Measures how connected the candidate node is
      - Stimulus: volition to attract new members (using feedback)
    - Member to Nonmember



Positive/Negative Feedback mechanism



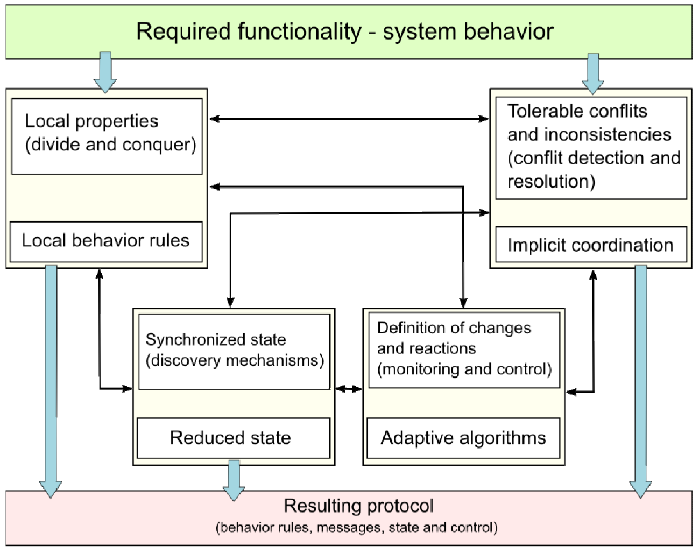
HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

Relation to Self-Organization Principles

- **Principles:**
  - #1 Design local behavior rules that achieve global properties
  - #2 Do not aim for perfect coordination: exploit implicit coordination
  - #3 Minimize long-lived global state information
  - #4 Design protocols that adapt to changes
- **Clusterhead selection:**
  - Uses just local information to select clusterheads (#1)
  - Tolerate conflicts (e.g. neighboring clusterheads) and use spoof techniques to discover the links (#2)
  - Each node just knows its own and immediate neighbourhood states (#3)
  - Clusterhead withdraws its role when minimum resource requirement can not be achieved anymore (#4)
- **Membership selection:**
  - Fitness function uses just local information. Attraction behavior is local interaction among neighbouring (#1,#3)
  - Inconsistencies are tolerated (a cluster much larger than the desired size) (#2)
  - Dynamic reconstruction of the cluster (#4)
- **However: propagation mechanism (wave-style broadcast) does not conform to self-organization principles**

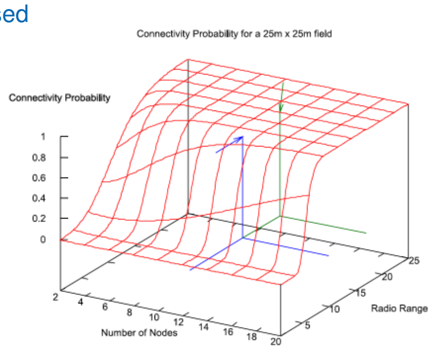
© Heinz Nixdorf Institute, University of Paderborn

Designing self-organizing systems:



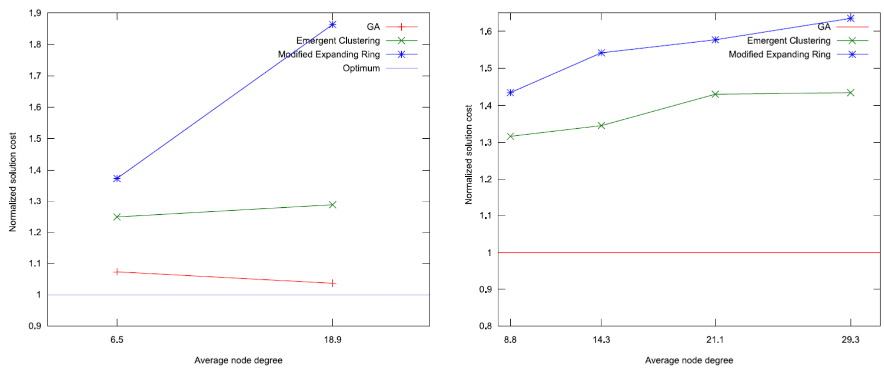
Results
"Quasi-Static" Clustering Heuristic Simulation

- **Shox wireless network simulator used**
- **Reference methods: integer programming (for small cases), genetic algorithm (for large ones)**
- **Assumptions**
  - Fixed power transmitters
  - Bidirectional links
  - Friis free space propagation model used
  - Link metric based on the RSSI
  - Every node has one unit of energy
  - Every node has one unit of resource
- **Simulation Scenarios**
  - Based on random graph theory
    - Connection probability



“Quasi-Static” Clustering Heuristic Simulation

§ Normalized Clustering Costs

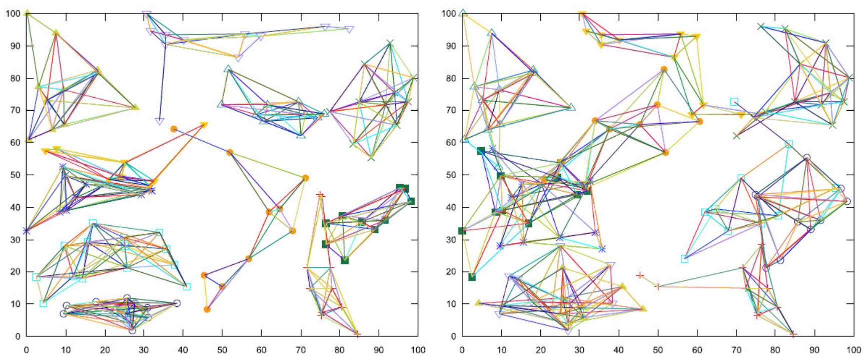


(a) Small Scenarios (b) Large Scenarios

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

“Quasi-Static” Clustering Heuristic Simulation

§ Example of Output



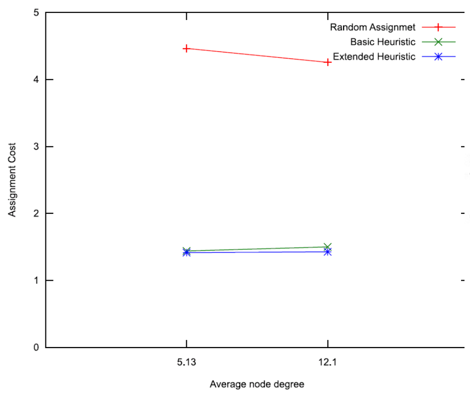
(a) Emergent Clustering (b) Modified Expanding Ring

© Heinz Nixdorf Institute, University of Paderborn

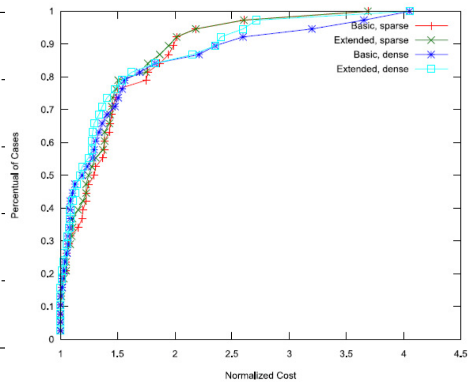
Service Distribution Simulation

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

§ **Normalized results**



§ **Cumulative Distribution**



Conclusions

HEINZ NIXDORF INSTITUTE
University Paderborn
Design of Parallel Systems
Prof. Dr. rer. nat. Franz J. Rammig

§ **Innovative OS for sensor networks supporting:**
- In-network processing
- Cooperative processing
- Self-optimization
- Aggregated capability and functionality using client-server paradigm

§ **Challenges**
- Automatic service placement
  - § Basic and extended ant based heuristic presented
  - § Basic and extended service distribution heuristics present very good results (max. 1.5 times the optimal)
- Network decomposition (clustering)
  - § Definition of the minimum intra-communication cost clustering problem
  - § Two heuristics presented
    - "Quasi" static networks, based on division of labour of ants
    - Dynamic networks, based on positive/negative feedback and emergence
  - § Emergent Clustering outperforms the modified expanding ring for all scenarios (with max. cost 1.44 times the reference)

Very good results based solely on local interactions and emergence!

"Principles encountered in nature can be transferred to computers with satisfactory results"

Thank you for your attention.

Heinz Nixdorf Institute
University of Paderborn
Fuerstenallee 11
33102 Paderborn
Germany

Phone: +49 (0) 52 51/60 65 01
Fax: +49 (0) 52 51/62 65 02
Email: franz@upb.de
http://www.heinz-nixdorf-institut.de