

**Optimizing Communications in  
Wireless Sensor Networks with  
Machine Learning**  
Part 1

Tutorial at NexTech 2009, Sliema, Malta

Dr. Anna Förster, University of Lugano, Switzerland  
anna.foerster@ieee.org

## Tutorial goal

- Learn how Machine Learning can help you optimize your WSN communications
  - Select the most appropriate ML technique for your problem
  - Design the ML based solution and implement it efficiently
  - Transfer the obtained knowledge to other wireless networking problems

Copyright: Anna Förster 2009

## Overview

### Part 1:

- Introduction to Wireless Sensor Networks
- Machine Learning techniques and their properties

### Part 2:

- State of the art applications of ML to WSNs
- Discussion of further application areas and problems

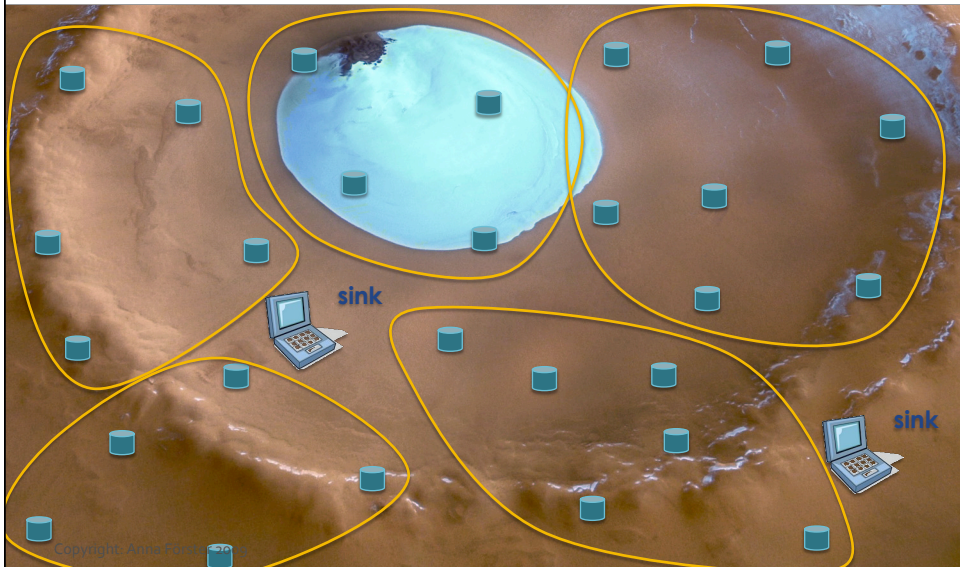
Copyright: Anna Förster 2009

## Wireless Sensor Networks

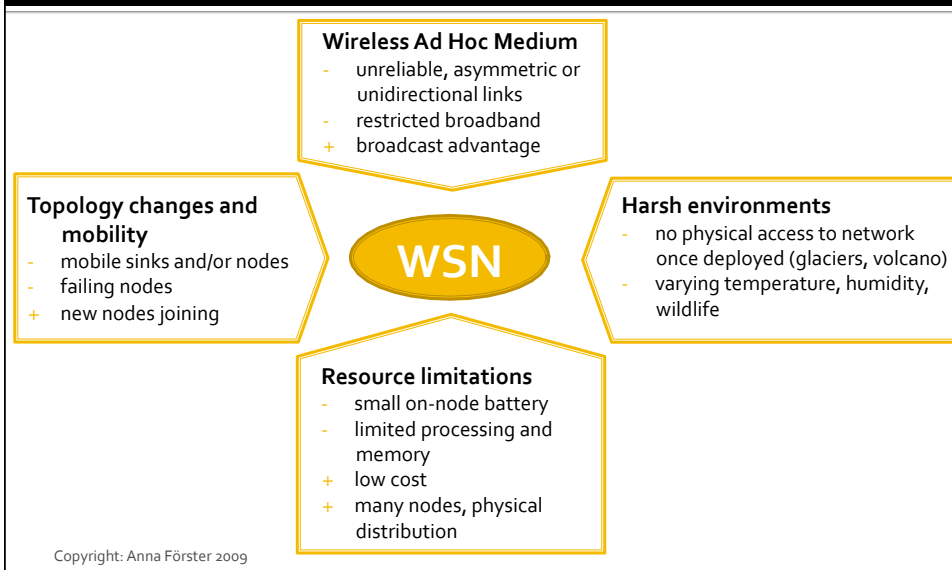
Overview, problems and challenges

Copyright: Anna Förster 2009

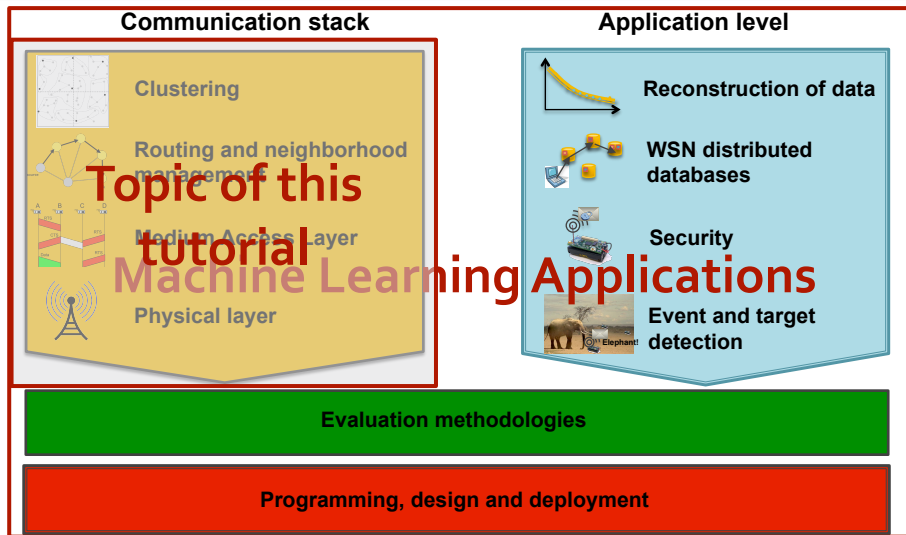
# Wireless Sensor Networks



# Challenges of Wireless Sensor Networks



# Developing Wireless Sensor Networks



Copyright: Anna Förster 2009

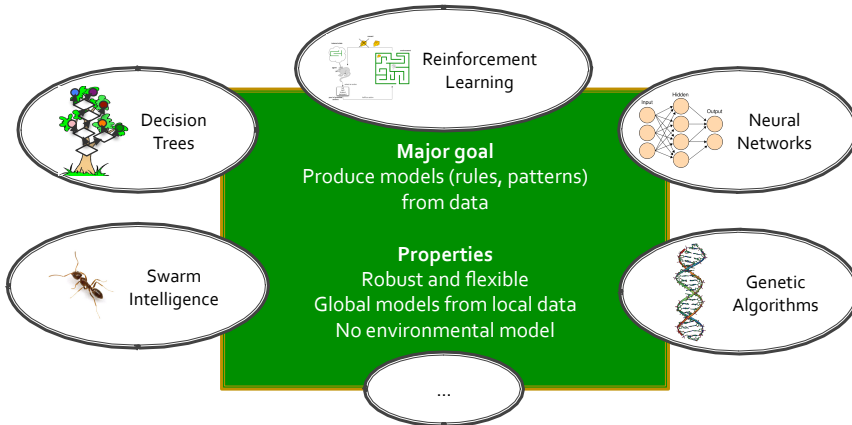
# Machine Learning

What is Machine Learning and why should we use it?

Copyright: Anna Förster 2009

# Machine Learning for WSNs

Machine learning can innately solve various challenges in WSNs and improve their performance significantly



Copyright: Anna Förster 2009

# Reinforcement Learning

Trial and error: Learn from your environment

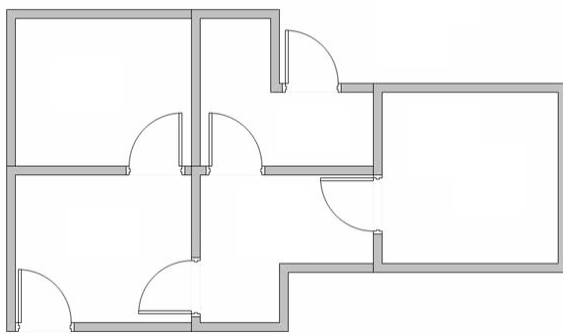
Copyright: Anna Förster 2009

## Reinforcement Learning

- A learning agent
- A pool of possible actions
- Goodness of actions
- A reward function
  
- Select one action
- Execute the action
- Observe the reward
- Correct the goodness of the executed action

Copyright: Anna Förster 2009

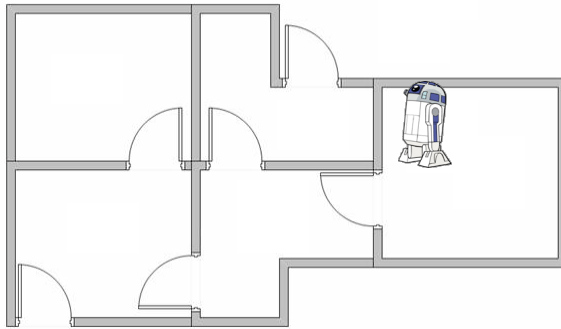
## Introduction to Q-Learning



Copyright: Anna Förster 2009

# Introduction to Q-Learning

▣ Learning agent

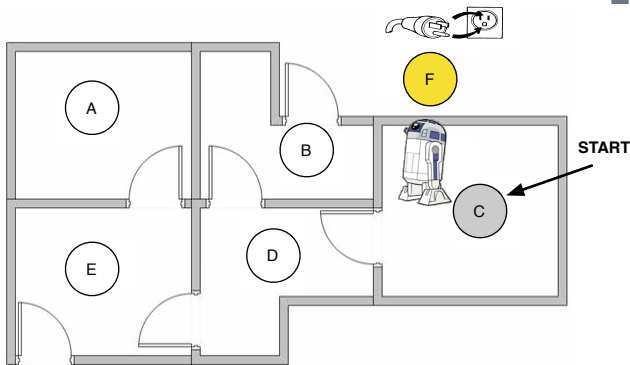


Copyright: Anna Förster 2009

# Introduction to Q-Learning

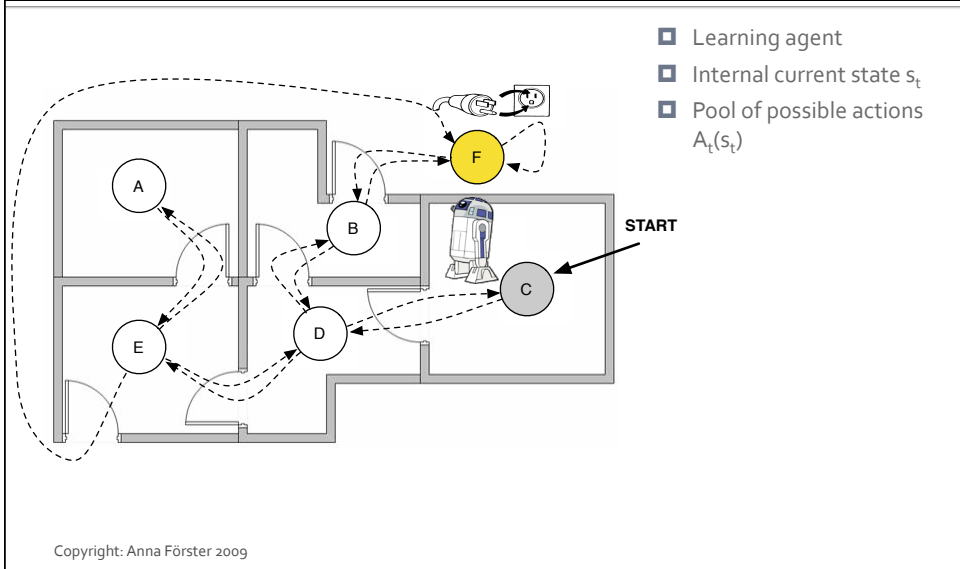
▣ Learning agent

▣ Internal current state  $s_t$

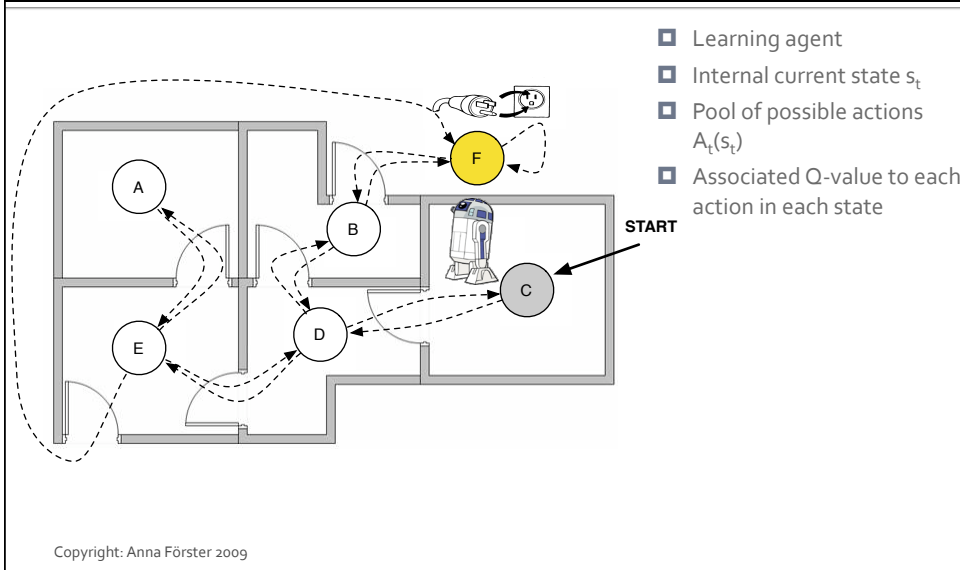


Copyright: Anna Förster 2009

# Introduction to Q-Learning



# Introduction to Q-Learning





# Introduction to Q-Learning

The diagram shows a robot in a maze with six states: A, B, C, D, E, and F. State C is the starting point. Transitions between states are labeled with actions: 0 (reward 0, cost -1) and -100 (reward 100, cost -2). State F is a goal state with a reward of 100. A dashed line indicates a path from C to B to A to E to D to B to F.

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action

- 0 --- action with immediate reward 0 and cost -1  
 - 100 --- action with immediate reward 100 and cost -2

Copyright: Anna Förster 2009

# Introduction to Q-Learning

The diagram is identical to the previous one, but with state C highlighted in grey and an orange arrow pointing from C to D. The text "1. select an action" is written in orange below the arrow.

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action
- ▣ Learning procedure:
  - ▣ select an action

- 0 --- action with immediate reward 0 and cost -1  
 - 100 --- action with immediate reward 100 and cost -2

Copyright: Anna Förster 2009

# Introduction to Q-Learning

**1. select an action**  
**2. execute the action**

immediate reward  $u$  and cost  $-1$   
 action with immediate reward 100 and cost  $-2$

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action
- ▣ Learning procedure:
  - ▣ select an action
  - ▣ execute the action

Copyright: Anna Förster 2009

# Introduction to Q-Learning

**1. select an action**  
**2. execute the action**  
**3. receive reward**

immediate reward  $u$  and cost  $-1$   
 action with immediate reward 100 and cost  $-2$

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action
- ▣ Learning procedure:
  - ▣ select an action
  - ▣ execute the action
  - ▣ observe reward

Copyright: Anna Förster 2009

# Introduction to Q-Learning

Copyright: Anna Förster 2009

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action
- ▣ Learning procedure:
  - ▣ select an action
  - ▣ execute the action
  - ▣ observe reward
  - ▣ update state and Q-values

# Introduction to Q-Learning

Copyright: Anna Förster 2009

- ▣ Learning agent
- ▣ Internal current state  $s_t$
- ▣ Pool of possible actions  $A_t(s_t)$
- ▣ Associated Q-value to each action in each state
- ▣ Immediate reward after each action
- ▣ Learning procedure:
  - ▣ select an action
  - ▣ execute the action
  - ▣ observe reward
  - ▣ update state and Q-values

## How to recompute the Q-values?

$$Q(s_{t+1}, a_t) = Q(s_t, a_t) + \gamma (R(s_t, a_t) - Q(s_t, a_t))$$

new Q-Value      old Q-Value      learning constant      old Q-Value  
 immediate reward received after executing action a in state s at time t

- Learning constant: avoid oscillations of Q values at the beginning of the learning process (smooth the Q-Values)
- $\gamma \approx 0$  : new Q-Value is exchanged with the reward
- $\gamma \approx 1$  : new Q-Value is the same as the old one

Copyright: Anna Förster 2009

## How to define the reward function?

- Two main types:
  - Pre-defined
  - Computed after each action
- Often used :
  - zero awards for actions leading directly to the goal
  - negative for all others (e.g. -1)
- Also used:
  - Manhattan distance to the goal
  - Geographic distance to the goal
  - Currently best available Q value at the state (!!)

Copyright: Anna Förster 2009

## How to decide which action to take?

- **Exploration strategy (action selection policy)**
- Cannot be random, need to use accumulated knowledge
- Cannot be greedy, need to explore all possibilities
- Often used:  $\epsilon$ -greedy
  - select a random action with probability  $\epsilon$
  - select the best available one (best Q-value) with probability  $(1-\epsilon)$

Copyright: Anna Förster 2009

## Properties of Reinforcement Learning

- Simple, flexible model
- Adapts to changing environments, re-learn quickly
- Copes successfully with mobile or unreliable environments
- Simple to design and implement
- Small to moderate processing and memory needs
- Can be implemented fully distributed

Copyright: Anna Förster 2009

## Reinforcement Learning for WSNs?

- All distributed problems:
  - Routing protocols
  - Clustering protocols
  - Neighborhood management protocols
  - Medium Access protocols
- Further
  - Parameter optimization and learning
  - Application-level cooperation among nodes

Copyright: Anna Förster 2009

## Decision Based Learning

Data classification

Copyright: Anna Förster 2009

# Decision Based Learning

- Classifying objects into groups based on attribute pairs



form = round  
color = orange  
taste = sour

orange



form = round  
color = red, orange, green  
taste = sweet

apple



?

Copyright: Anna Förster 2009

# Decision Based Learning

- Classifying objects into groups based on attribute pairs



form = round  
color = orange  
taste = sour

orange



form = round  
color = red, orange, green  
taste = sweet

apple



form = ?  
color = ?  
taste = ?

Copyright: Anna Förster 2009

# Decision Based Learning

- Classifying objects into groups based on attribute pairs



orange

form = round  
color = orange  
taste = sour



apple

form = round  
color = red, orange, green  
taste = sweet

???



form = round  
color = ?  
taste = ?

Copyright: Anna Förster 2009

# Decision Based Learning

- Classifying objects into groups based on attribute pairs



orange

form = round  
color = orange  
taste = sour



apple

form = round  
color = orange, red, green  
taste = sweet

???



form = round  
color = orange  
taste = ?

Copyright: Anna Förster 2009



# Decision Based Learning

- Classifying objects into groups based on attribute pairs



orange

form = round  
color = orange  
taste = sour



apple

form = round  
color = orange, red, green  
taste = sweet

apple!



form = round  
color = orange  
taste = sweet

Copyright: Anna Förster 2009

# Decision Based Learning

- Classifying objects into groups based on attribute pairs



orange

form = round  
color = orange  
taste = sour



apple

form = round  
color = orange, red, green  
taste = sweet

???



taste = ?  
color = ?  
form = ?

Copyright: Anna Förster 2009

## Decision Based Learning

- Classifying objects into groups based on attribute pairs



orange

form = round  
color = orange  
taste = sour



apple

form = round  
color = orange, red, green  
taste = sweet

apple!



taste = sweet  
color = ?  
form = ?

Copyright: Anna Förster 2009

## Decision Based Learning

- Classifying objects into groups based on attribute pairs
- Which questions to ask first, which next?
- Compute information gain of attributes
  - How well does an attribute separates the testing set?

Copyright: Anna Förster 2009

## C4.5 algorithm

**Goal: construct a decision tree with attribute at each node**

1. Start at root
2. Find the attribute with maximal information gain, which is not an ancestor of the node
3. Put a child node for each value of this attribute
4. Add all examples from the training set to the corresponding child
5. If all examples of a child belong to the same class, put the class there and go back up in the tree
6. If not, continue with step 2 while attributes are left
7. When no more attributes are left, put the classification of the majority of the examples to this node

Copyright: Anna Förster 2009

## C4.5 algorithm: Example

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange

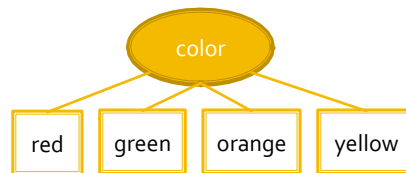


- Information gain of FORM: zero
- Information gain of COLOR: more

Copyright: Anna Förster 2009

## C4.5 algorithm: Example

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange

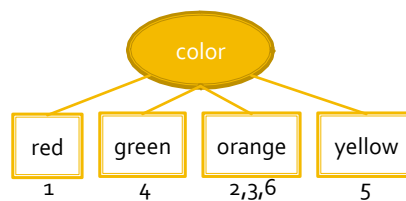


- Information gain of FORM: zero
- Information gain of COLOR: more

Copyright: Anna Förster 2009

## C4.5 algorithm: Example

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange

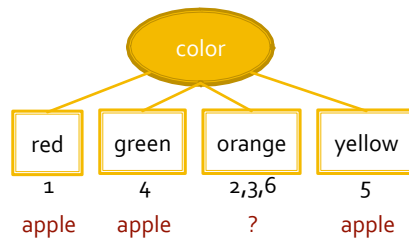


- Information gain of FORM: zero
- Information gain of COLOR: more

Copyright: Anna Förster 2009

## C4.5 algorithm: Example

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange

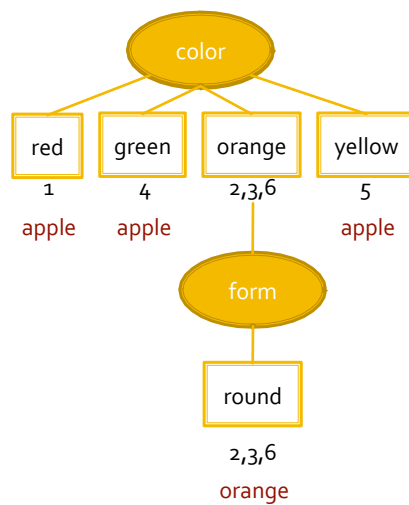


- Information gain of FORM: zero
- Information gain of COLOR: more

Copyright: Anna Förster 2009

## C4.5 algorithm: Example

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange

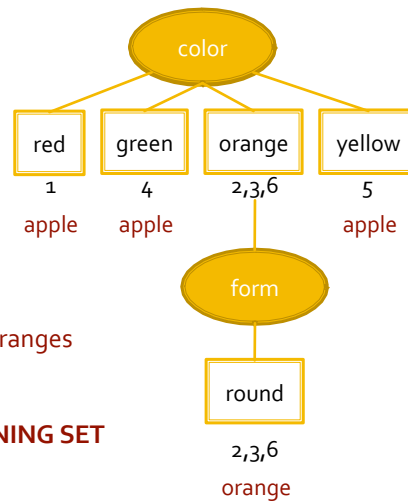


- Information gain of FORM: zero
- Information gain of COLOR: more
- Only left attribute: FORM

Copyright: Anna Förster 2009

## C4.5 algorithm: Problems

example	form	color	class
1	round	red	apple
2	round	orange	apple
3	round	orange	orange
4	round	green	apple
5	round	yellow	apple
6	round	orange	orange



- All orange apples will be classified as oranges
- Leaf node FORM unnecessary
- **DECISION TREE DEPENDS ON TRAINING SET**

Copyright: Anna Förster 2009

## Properties of Decision Based Learning

- Good for fast classification of fuzzy, overlapping groups
- Tree generated only once
- Well-suited for static, but error-prone environments
- Needs a good large training set
- Moderate processing and large memory requirements (to hold the training set)

Copyright: Anna Förster 2009

## Decision Based Learning for WSNs?

- Static problems
  - Link quality classification
  - Network status classification
  - Battery level classification
  - ...

Copyright: Anna Förster 2009

## Genetic algorithms

Evolve and survive

Copyright: Anna Förster 2009

## Genetic algorithms

- Basic idea
  - Represent a solution to a problem as an individual (bit string)
  - Create many individuals
  - Evaluate the individuals based on a fitness function
  - Select the best individuals, use them to create new individuals
  - Repeat until solution good enough

Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

- Individual: bit string of cities

1 2 3 4 5 6 7

- Fitness function: the length of the tour

- Create random tours

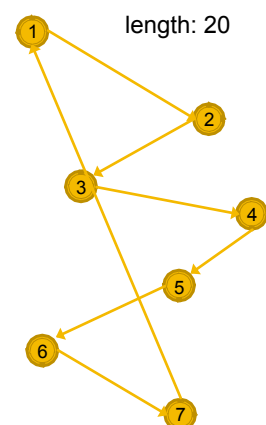
1 2 3 4 5 6 7 length: 20

7 2 1 4 3 6 5 length: 17

2 1 6 5 3 4 7 length: 22

3 5 1 4 7 2 6 length: 25

7 6 3 2 5 4 1 length: 21

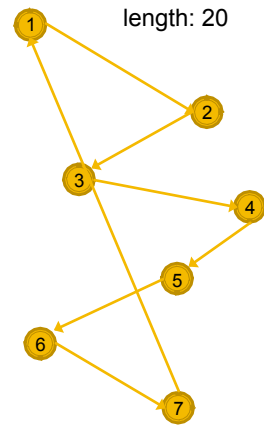


Copyright: Anna Förster 2009



## Example: Traveling Salesman Problem

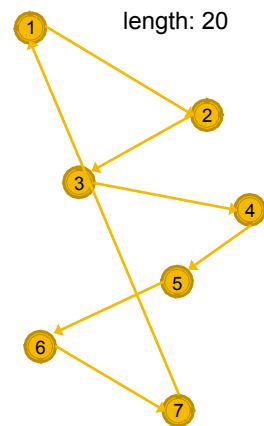
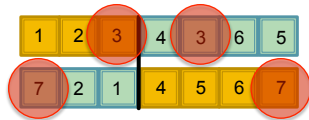
- Crossover



Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

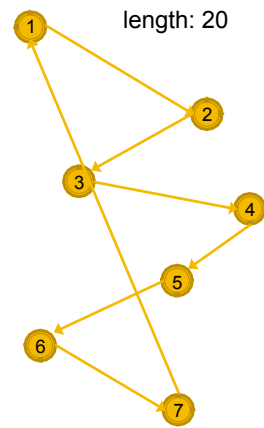
- Crossover



Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

- Crossover



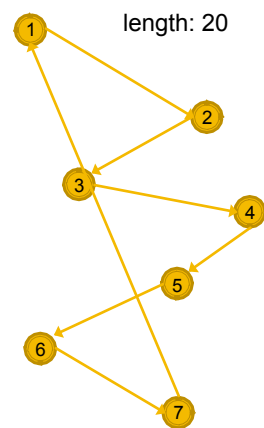
Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

- Crossover



- Mutation



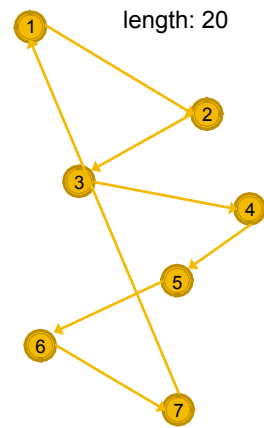
Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

- Crossover



- Mutation



Copyright: Anna Förster 2009

## Example: Traveling Salesman Problem

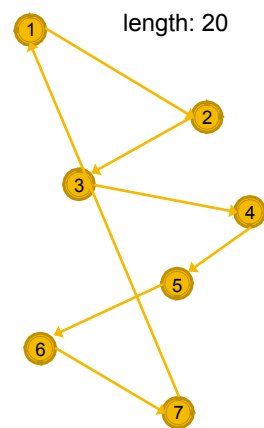
- Crossover



- Mutation



- Evaluate new generation, REPEAT



Copyright: Anna Förster 2009

## Properties of Genetic Algorithms

- Search algorithms for very large search spaces
- Find near-optimal solutions to NP-hard problems (e.g. TSP)
- Very high memory and processing requirements
- Not flexible to changes in the environment (add a new city to TSP)

Copyright: Anna Förster 2009

## Genetic algorithms for WSNs?

- Good for static problems with centralized data processing:
  - centralized scheduling and clustering
  - optimal sensing coverage
  - optimal transmission power of nodes
  - deployment planning
- Not suited for distributed error-prone problems like routing, MAC protocols or large-scale clustering

Copyright: Anna Förster 2009

# Swarm Intelligence

Distribute the thinking

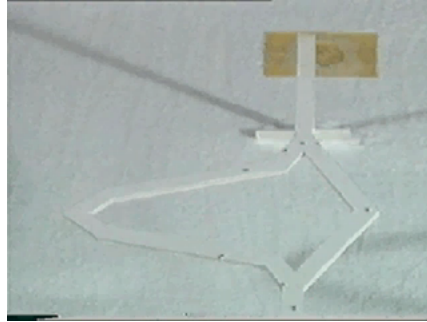
Copyright: Anna Förster 2009

# Swarm Intelligence

- Biologically inspired from ants, bees, etc.
- Small, restricted entities (agents)
- Communicate through the environment (e.g. pheromones)
- Solve problems not solvable for individual agents

Copyright: Anna Förster 2009

## Swarm Intelligence: Shortest Paths



Video cordially provided by Prof. Luca M. Gambardella

Copyright: Anna Förster 2009

## Swarm Intelligence: Bridging the gap



Video cordially provided by Prof. Luca M. Gambardella

Copyright: Anna Förster 2009

## Swarm Intelligence

- Particle Swarm Optimization
- Ant Colony Optimization
- Honey Bee Algorithm

Copyright: Anna Förster 2009

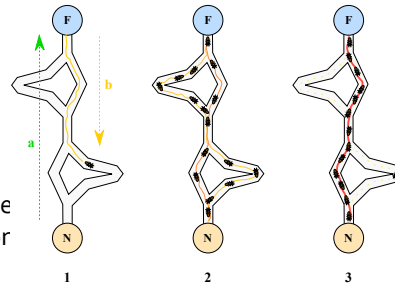
## Swarm Intelligence

- Particle Swarm Optimization
- **Ant Colony Optimization**
- Honey Bee Algorithm

Copyright: Anna Förster 2009

## Ant Colony Optimization

- Represent solution as a graph (e.g. find shortest path between 2 nodes)
- Initialize all edges in the graph with some small amount of pheromone
- Place ants at the source node
- Ants take random decisions on which edge to proceed based on placed pheromone or that edge
- When the destination is reached, ant calculates fitness (goodness) of the solution (path) and traverses the same path back to lay additional pheromone
- Ants converge on the shortest path, but continue exploring other routes too



Copyright: Anna Förster 2009

## Properties of Ant Colony Optimization

- Fully distributed
- Low memory and processing requirements
- Moderate management requirements (see next slide)
- Very flexible to environmental changes
- Very robust against failures and topology changes

Copyright: Anna Förster 2009



## Ant Colony Optimization for WSNs?

- Distributed problems such as unicast routing
  - Very flexible against topology changes, even high speed mobility
  - Some additional communication overhead needed for backward ants and pheromone sharing
- Less suited for broadcast problems: breaks the analogy with ants and changes the model significantly

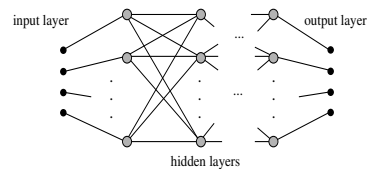
Copyright: Anna Förster 2009

## Other Machine Learning Techniques

Copyright: Anna Förster 2009

## Neural Networks

- High memory and processing requirements for training
- Small memory and processing requirements for using
- Need large, carefully prepared training set
- Solution design non-intuitive
- Online versions exist, but have high resources requirements
- Not suited when environmental changes expected

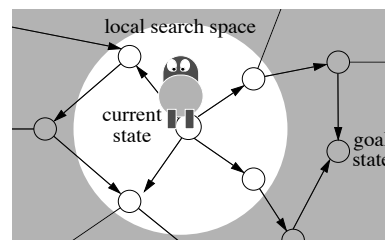


Picture from "A Systematic Introduction to Neural Networks", Raul Rojas, 1996

Copyright: Anna Förster 2009

## Heuristic Search

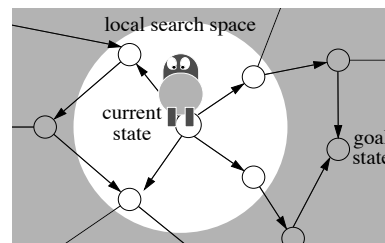
- Well-known examples from routing in WSNs:
  - number of hops
- Traditional heuristic search
  - Build the full search tree
  - Compute the value (goodness) of each node
  - Take the minimum value path
- Online heuristic search, agent-centered search
  - Evaluate the direct neighborhood of the agent (directly reachable states)
  - Take the minimum valued one



Copyright: Anna Förster 2009

## Heuristic Search

- Important difference to Reinforcement Learning:  
the true, exactly calculated goodness values need to be available a-priori at the nodes!
- Resembles Q-Learning after the Q-Values have stabilized



Copyright: Anna Förster 2009

## Mobile Agents

- Often called (mistakenly!) ants
- Small entities, traveling through the network and gathering fresh information (e.g. routing table entries)
- Have nothing to do with machine learning, swarm intelligence or any other type of intelligence!
- Good for keeping important information fresh

Copyright: Anna Förster 2009

# Machine Learning Techniques

Summary and Property Overview

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
---------------	--------	-------------	-------------------------------	------------	------------	------------

required memory for on-node storage

required processing on the node or base station

flexibility of the found solution to environmental changes

optimality of derived solution compared to a centrally computed optimal solution

required communication or processing costs before starting normal work

additional communication or processing costs during runtime

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
---------------	--------	-------------	-------------------------------	------------	------------	------------

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium
Heuristics	low	low	low/medium	medium	high	low

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium
Heuristics	low	low	low/medium	medium	high	low
Mobile Agents	low	low	medium	low	low	medium /high

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium
Heuristics	low	low	low/medium	medium	high	low
Mobile Agents	low	low	medium	low	low	medium /high
Neural networks	medium	medium	low	high	high	low

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add. costs
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium
Heuristics	low	low	low/medium	medium	high	low
Mobile Agents	low	low	medium	low	low	medium /high
Neural networks	medium	medium	low	high	high	low
Genetic algorithms	high	medium	low	high	high	low

Copyright: Anna Förster 2009

## Comparison of properties

ML Techniques	Memory	Computation	Tolerance to topology changes	Optimality	Init.costs	Add.
Reinforcement Learning	low	low	high	high	medium	low
Swarm Intelligence	medium	low	high	high	high	medium
Heuristics	low	low	low/medium	medium	high	low
Mobile Agents	low	low	medium	low	low	mediur /high
Neural networks	medium	medium	low	high	high	low
Genetic algorithms	high	medium	low	high	high	low

Distributed problems

Optimization

Centralized and localized problems

Copyright: Anna Förster 2009



## Further readings



M. Dorigo and T. Stuetzle.  
**Ant Colony Optimization.**  
MIT Press, 2004.



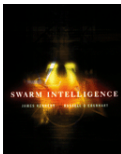
T.M. Mitchell.  
**Machine Learning.**  
McGraw-Hill, 1997.



R. S. Sutton and A. G. Barto.  
**Reinforcement Learning:  
An Introduction.**  
The MIT Press, March 1998.



A. Förster.  
**Teaching Networks How to  
Learn**  
SVH Verlag, 2009



J. Kennedy and R.C. Eberhart.  
**Swarm Intelligence.**  
Morgan Kaufmann, 2001.



S.J. Russell and P. Norvig.  
**Artificial Intelligence:  
A Modern Approach.**  
Prentice Hall International, 2003.

Copyright: Anna Förster 2009



**Optimizing Communications in  
Wireless Sensor Networks with  
Machine Learning**  
Part 2

Tutorial at NexTech 2009, Sliema, Malta

Dr. Anna Förster, University of Lugano, Switzerland  
anna.foerster@ieee.org

## Overview

### Part 1:

- Introduction to Wireless Sensor Networks
- Machine Learning techniques and their properties

### Part 2:

- State of the art applications of ML to WSNs
- Discussion of further application areas and problems

## Reinforcement Learning for WSNs

State of the art

Copyright: Anna Förster 2009

## Q-Learning in WSN Routing

- **Agents:** the packets
- **States:** the nodes
- **Actions:** next hops
- **q-values:** estimations of routing costs
- Initial q-values: some first guess about routing costs
- **Reward function:** the best cost estimation of the next hop
- **Exploration strategy:** simple, e.g.  $\epsilon$ -greedy

Copyright: Anna Förster 2009

# Unicast routing with RL

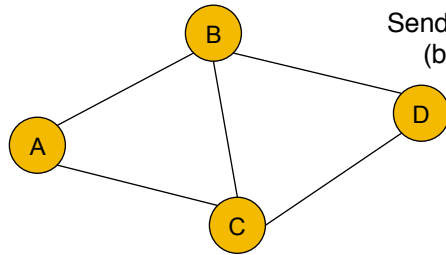
Sending a packet from A to D  
Init all q values to 10 (guess)

Rewards:

$r = q_{best}$ , if not sink

$r = 0$ , if sink

Send rewards to all neighbors  
(broadcast)



Copyright: Anna Förster 2009

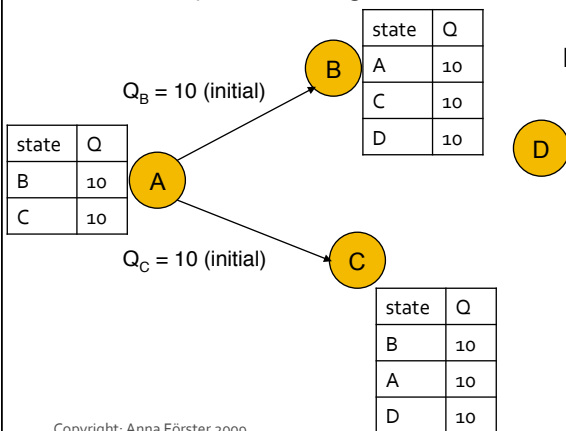
# Unicast routing with RL

Sending a packet from A to D  
Init all q values to 10 (guess)

Action selection policy  
(Exploration strategy)

$\epsilon$ -greedy

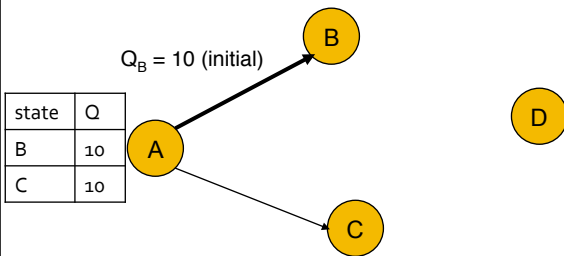
Balance exploration/exploitation



Copyright: Anna Förster 2009

# Unicast routing with RL

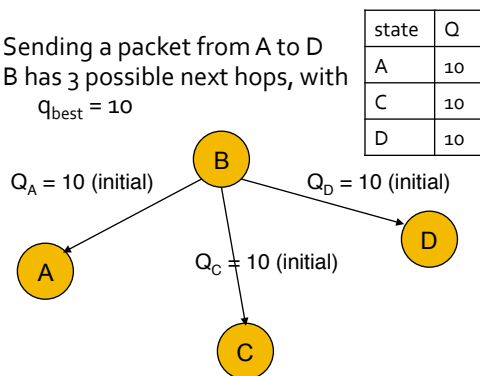
Sending a packet from A to D  
 Select next hop (state) B



Copyright: Anna Förster 2009

# Unicast routing with RL

Sending a packet from A to D  
 B has 3 possible next hops, with  
 $Q_{best} = 10$

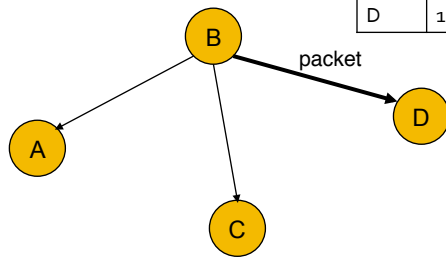


Copyright: Anna Förster 2009

# Unicast routing with RL

Sending a packet from A to D  
B selects D as next hop,

state	Q
A	10
C	10
D	10

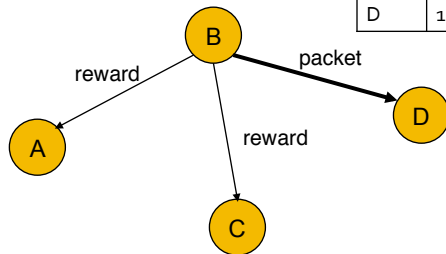


Copyright: Anna Förster 2009

# Unicast routing with RL

Sending a packet from A to D  
B selects D as next hop,  
reward =  $q_{best} = 10$

state	Q
A	10
C	10
D	10

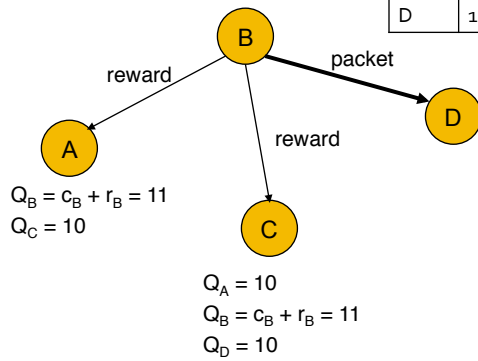


Copyright: Anna Förster 2009

# Unicast routing with RL

Sending a packet from A to D  
 B selects D as next hop,  
 reward =  $q_{best} = 10$

state	Q
A	10
C	10
D	10

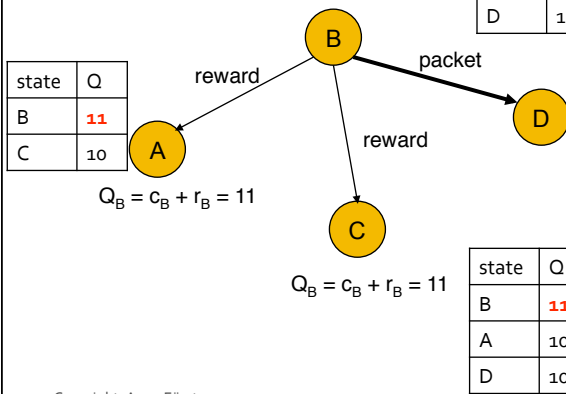


Copyright: Anna Förster 2009

# Unicast routing with RL

Sending a packet from A to D  
 B selects D as next hop,  
 reward =  $q_{best} = 10$

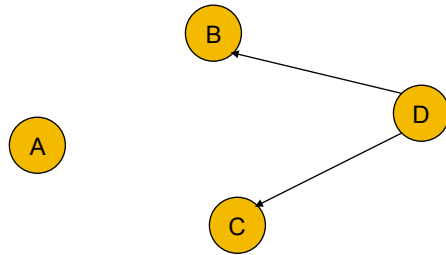
state	Q
A	10
C	10
D	10



Copyright: Anna Förster 2009

## Unicast routing with RL

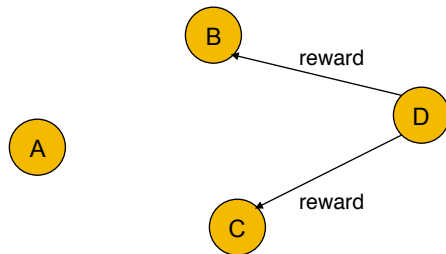
Sending a packet from A to D  
D is the sink, goal reached



Copyright: Anna Förster 2009

## Unicast routing with RL

Sending a packet from A to D  
D is the sink, goal reached  
reward = 0 (real costs)



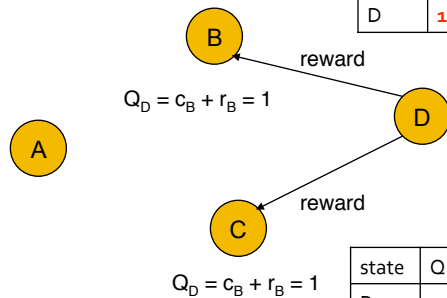
Copyright: Anna Förster 2009



# Unicast routing with RL

Sending a packet from A to D  
D is the sink, goal reached  
reward = 0 (real costs)

state	Q
A	10
C	10
D	1

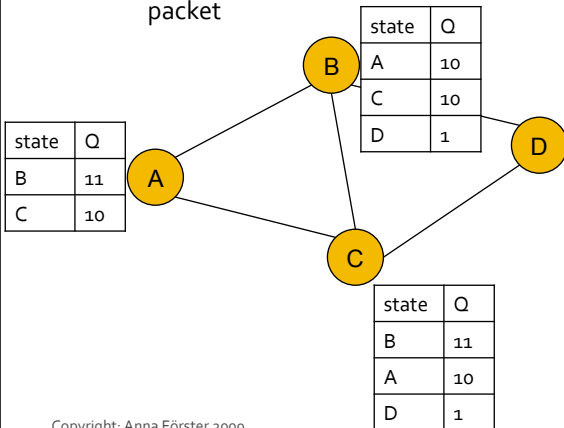


state	Q
B	11
A	10
D	1

Copyright: Anna Förster 2009

# Unicast routing with RL

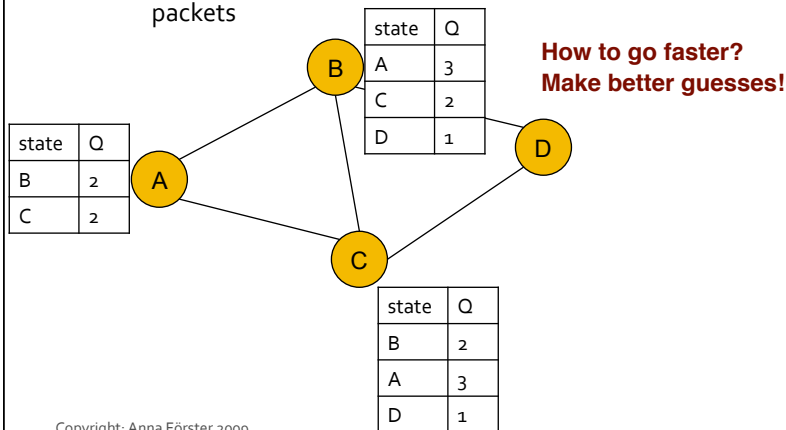
Sending a packet from A to D  
State of the network after first  
packet



Copyright: Anna Förster 2009

## Unicast routing with RL

Sending a packet from A to D  
State of the network after many packets



## Unicast routing with RL Benefits

- Simple and powerful
- Reacts immediately to changes:
  - New rewards propagate quickly
  - New routes are learnt
  - Only necessary changes in the immediate neighborhood of failure
- Route initialization is sink/source driven
- Low memory and processing overhead

Copyright: Anna Förster 2009

## Unicast Routing with RL

- Hops: too trivial to deserve a publication...

- Maximum aggregation rate:

P. Beyens, M. Peeters, K. Steenhaut, and A. Nowe. **Routing with compression in wireless sensor networks: A Q-learning approach.** In Proceedings of the 5th European Workshop on Adaptive Agents and Multi-Agent Systems (AAMAS), page 12pp., Paris, France, 2005.

- Combined with geographic routing:

R. Arroyo-Valles, R. Alaiz-Rodrigues, A. Guerrero-Curieses, and J. Cid-Suiero. **Q-probabilistic routing in wireless sensor networks.** In Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pages 1–6, Melbourne, Australia, 2007.

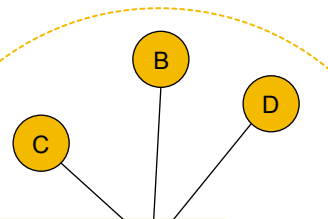
- Minimum delay:

J. A. Boyan and M. L. Littman. **Packet routing in dynamically changing networks: A reinforcement learning approach.** Advances in Neural Information Processing Systems, 6:671–678, 1994.

Copyright: Anna Förster 2009

## Multicast Routing with RL

- Challenges:
  - Actions need to reflect not the next hop, but HOPS
  - Reward function is distributed among several neighbors
  - Set of actions very large – needs a lot of exploration!
- Solution steps:
  - Separate actions into sub-actions
  - Smart initial Q values



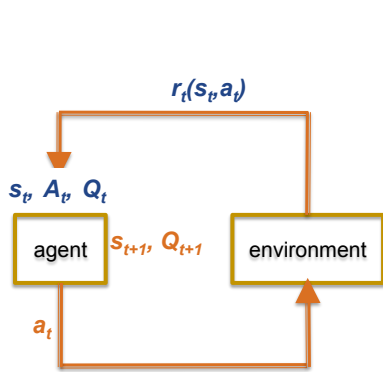
A. Förster and A. L. Murphy.

**FROMS: Feedback routing for optimizing multiple sinks in WSN with reinforcement learning.**

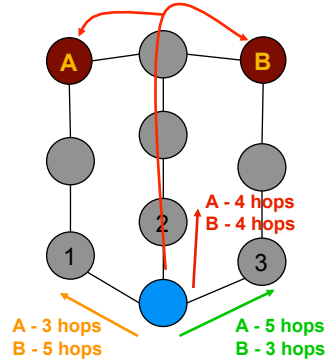
In Proceedings 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), pp. 371–376, Australia, 2007.

Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning



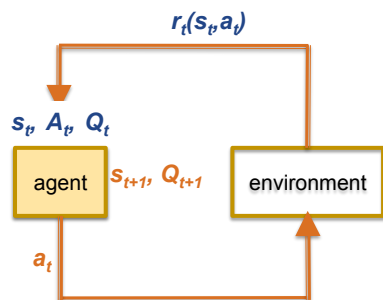
- Localized view after sink announcement
- The minimum estimated is not the optimal:
  - best estimate for (A,B): 3 + 3 - 1 = 5 hops
  - optimal for (A,B): 4 hops



Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

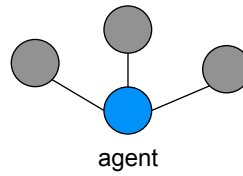
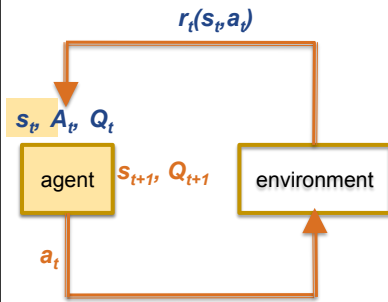
- Agent: each node in the network



Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

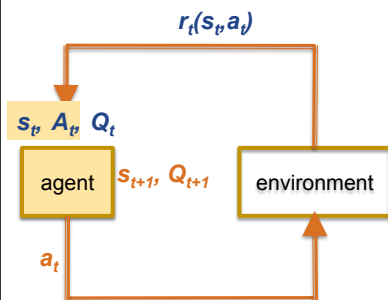
- Agent: each node in the network
- State: agent's neighbors



Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of neighbors to reach all sinks

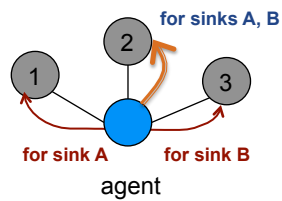


**sub-actions**

**Actions:**

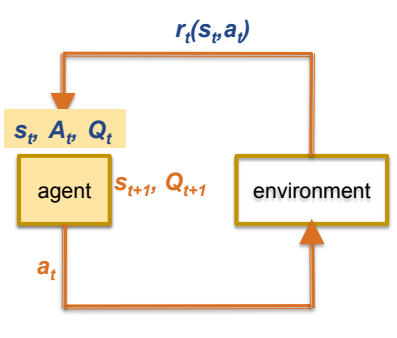
$a_i = \{n_1 \text{ for } A\}, \{n_3 \text{ for } B\}$

$a_j = \{n_2 \text{ for } A, B\}$



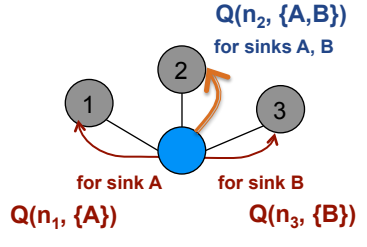
Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning



The diagram shows an agent and an environment in a loop. The agent provides action  $a_t$  to the environment. The environment returns state  $s_{t+1}$  and Q-value  $Q_{t+1}$  to the agent. The environment also receives a reward  $r_t(s_t, a_t)$  from the agent. The agent's current state is  $s_t$  and its action is  $A_t$ , with a Q-value  $Q_t$ .

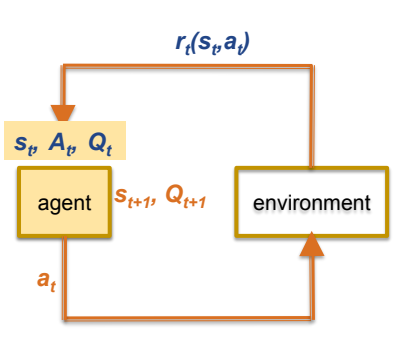
- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of neighbors
- Q Values: associate with
  - each sub-action
  - computable for each (full) action



A network diagram with a central blue node (node 2) and two peripheral nodes (node 1 and node 3). Node 2 is connected to both node 1 and node 3. Arrows point from node 2 to node 1 and from node 2 to node 3. Labels below the nodes indicate Q-values:  $Q(n_1, \{A\})$  for sink A,  $Q(n_3, \{B\})$  for sink B, and  $Q(n_2, \{A, B\})$  for sinks A, B.

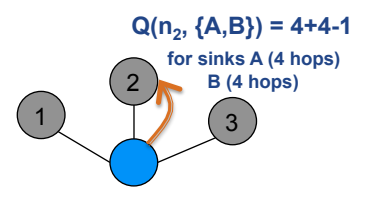
Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning



The diagram shows an agent and an environment in a loop. The agent provides action  $a_t$  to the environment. The environment returns state  $s_{t+1}$  and Q-value  $Q_{t+1}$  to the agent. The environment also receives a reward  $r_t(s_t, a_t)$  from the agent. The agent's current state is  $s_t$  and its action is  $A_t$ , with a Q-value  $Q_t$ .

- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of neighbors
- Q Values: associate with sub-actions, compute for actions
- Initialize Q Values with number of estimated hops



A network diagram with a central blue node (node 2) and two peripheral nodes (node 1 and node 3). Node 2 is connected to both node 1 and node 3. Arrows point from node 2 to node 1 and from node 2 to node 3. Labels below the nodes indicate Q-values:  $Q(n_1, \{A\})$  for sink A (4 hops),  $Q(n_3, \{B\})$  for sink B (4 hops), and  $Q(n_2, \{A, B\}) = 4+4-1$  for sinks A (4 hops) B (4 hops).

Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of neighbors
- Q Values: associate with sub-actions, compute for actions
- Initialize Q Values with number of estimated hops
- Environment: all other nodes

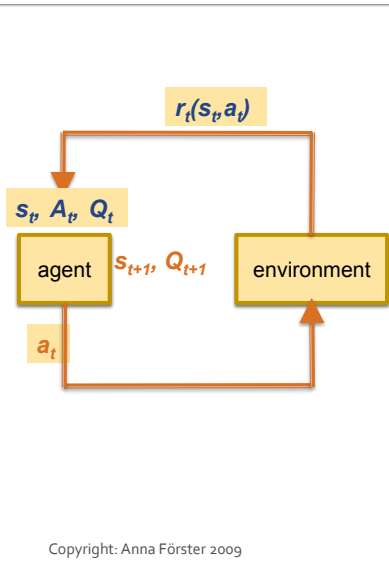
Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

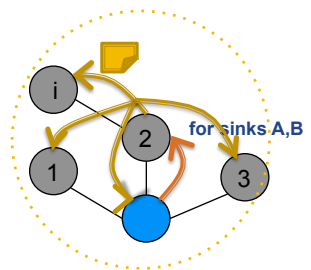
- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of
- Q Values: associate with sub-actions, compute for actions
- Initialize Q Values with number of estimated hops
- Environment: all other nodes

Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning

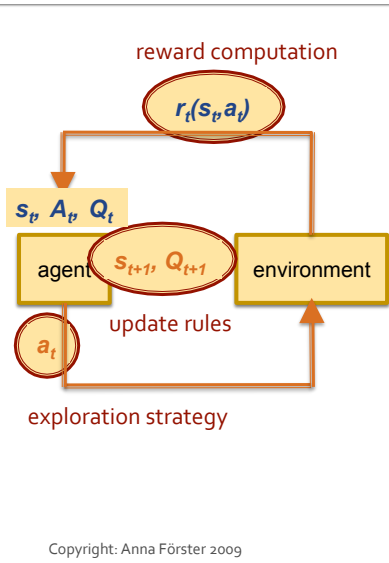


- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of
- Q Values: associate with sub-actions, compute for actions
- Initialize Q Values with number of estimated hops
- Environment: all other nodes
- Reward: the best available Q value + 1 hop

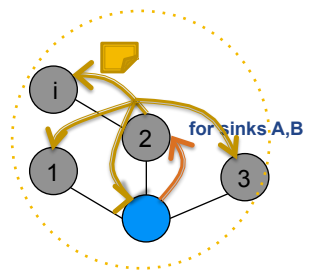


Copyright: Anna Förster 2009

## FROMS: Multicast routing with Q-Learning



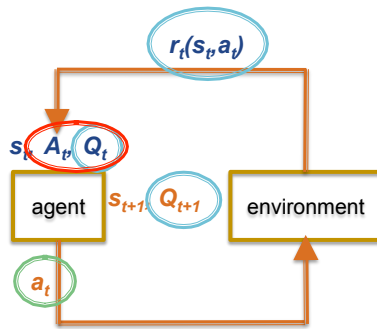
- Agent: each node in the network
- State: agent's neighbors
- Possible actions: combination of
- Q Values: associate with sub-actions, compute for actions
- Initialize Q Values with number of estimated hops
- Environment: all other nodes
- Reward: the best available Q value + 1 hop
- Update at neighboring nodes (learn)



Copyright: Anna Förster 2009



## Parameters of FROMS



- Possible cost functions:
  - Any cost function defined over the edges or nodes of the communication graph
  - Here: **minimum hops to destinations**
  - Further: minimum delay to the sinks; minimum geographic progress; minimum transmission power; **maximum remaining energy on the nodes**; combinations; ...
- Exploration strategy
  - Balance exploration against exploitation
  - Depend on the used cost function
- Memory management
  - Heuristics for pruning the available actions and sub-actions

Copyright: Anna Förster 2009

## Further Applications of RL to WSNs

### ■ Clustering for WSNs:

Anna Förster and Amy L. Murphy, **Clique: Role-free Clustering with Q-Learning for Wireless Sensor Networks**, in Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS) 2009, 9pp., Canada, June 2009

### ■ MAC protocols:

Z. Liu and I. Elahany. **RL-MAC: A reinforcement learning based MAC protocol for wireless sensor networks**. International Journal on Sensor Networks, 1(3/4):117–124, 2006.

### ■ Best coverage:

M.W.M. Seah, C.K. Tham, K. Srinivasan, and A. Xin. **Achieving coverage through distributed reinforcement learning in wireless sensor networks**. In Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2007.

Copyright: Anna Förster 2009

## Reinforcement Learning for WSNs

Take away

**First choice when solving complex distributed problems in WSNs**

Copyright: Anna Förster 2009

## Decision Based Learning for WSNs

State of the art

Copyright: Anna Förster 2009

# Link Quality with Decision Trees

RSSI	received signal strength indication	local
sendBuf	send buffer size	local
fwdBuf	forward buffer size	local
depth	node depth from base station	non-local
CLA	channel load assessment	local
pSend	forward probability	local
pRecv	backward probability	local

### MetricMap

Wang, Y., Martonosi, M. & Peh, L.-S. (2006). A supervised learning approach for routing optimizations in wireless sensor networks, Proceedings of the 2nd International Workshop on Multi-hop ad hoc networks: from theory to reality (REALMAN), Florence, Italy, pp. 79–86.

- Gathers information about links
- Uses LQI (Link Quality Indication) to classify links as good or bad

Copyright: Anna Förster 2009

# Link Quality with Decision Trees

```
rssl <= 212
|
|  depth <= 5
|  |  rssi <= 211: bad (320.0/37.0)
|  |  rssi > 211: good (79.0/34.0)
|  |  depth > 5: bad (125.0/31.0)
|  rssi > 212
|  |  rssi <= 223
|  |  |  cla <= 216
|  |  |  |  depth <= 3: good (352.0/82.0)
|  |  |  |  depth > 3
|  |  |  |  |  depth <= 4
|  |  |  |  |  |  rssi <= 220: bad (49.0/1.0)
|  |  |  |  |  |  rssi > 220
|  |  |  |  |  |  |  cla <= 8: good (69.0/29.0)
|  |  |  |  |  |  |  cla > 8: bad (14.0/4.0)
|  |  |  |  |  depth > 4
|  |  |  |  |  |  depth <= 6
|  |  |  |  |  |  |  rssi <= 216
|  |  |  |  |  |  |  |  depth <= 5: good (198.0/71.0)
|  |  |  |  |  |  |  |  depth > 5
|  |  |  |  |  |  |  |  |  rssi <= 214: bad (8.0/1.0)
|  |  |  |  |  |  |  |  |  rssi > 214
|  |  |  |  |  |  |  |  |  |  sendbuf <= 0
|  |  |  |  |  |  |  |  |  |  |  cla <= 21: bad (29.0/13.0)
|  |  |  |  |  |  |  |  |  |  |  cla > 21: good (2.0)
|  |  |  |  |  |  |  |  |  |  |  sendbuf > 0: good (2.0)
|  |  |  |  |  |  |  |  rssi > 216: good (178.0/34.0)
|  |  |  |  |  depth > 6
|  |  |  |  |  |  rssi <= 219
|  |  |  |  |  |  |  rssi <= 215: good (157.0/55.0)
|  |  |  |  |  |  |  rssi > 215
|  |  |  |  |  |  |  |  depth <= 7
|  |  |  |  |  |  |  |  |  rssi <= 217: bad (129.0/29.0)
|  |  |  |  |  |  |  |  |  rssi > 217
|  |  |  |  |  |  |  |  |  |  cla <= 0: good (20.0/6.0)
|  |  |  |  |  |  |  |  |  |  cla > 0: bad (12.0/3.0)
|  |  |  |  |  |  |  |  depth > 7
|  |  |  |  |  |  |  |  |  |  rssi <= 217: good (37.0/17.0)
|  |  |  |  |  |  |  |  |  |  rssi > 217
|  |  |  |  |  |  |  |  |  |  |  cla <= 0: bad (21.0/3.0)
|  |  |  |  |  |  |  |  |  |  |  cla > 0: good (2.0)
|  |  |  |  |  |  rssi > 219
|  |  |  |  |  |  |  |  depth <= 7
|  |  |  |  |  |  |  |  |  cla <= 3: good (102.0/35.0)
|  |  |  |  |  |  |  |  |  cla > 3: bad (30.0/12.0)
|  |  |  |  |  |  |  |  |  |  depth > 7: good (85.0/17.0)
|  |  |  |  |  |  |  |  |  |  |  cla > 116: good (62.0/8.0)
|  |  |  |  |  |  |  |  |  |  |  cla > 116: bad (58.0/17.0)
```

**RSSI value is the root!**

**Three initial features ignored: fwdBuffer, sendP and recP**

## Further Applications of Decision Trees to WSNs

- Classification of any localized, relatively static properties:
  - Battery levels
  - Network conditions
  - Channel load
  - Node status
  - ...
- **A lot of future work!**

Copyright: Anna Förster 2009

## Decision Trees for WSNs

Take away

**Good choice when classifying  
complex properties**

Copyright: Anna Förster 2009

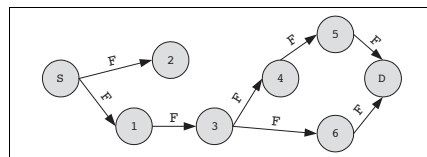
# Ant Colony Optimization for WSNs

State of the art

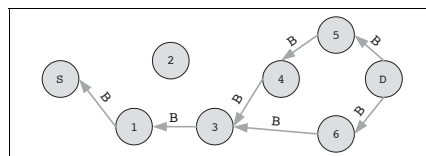
Copyright: Anna Förster 2009

## ARA – Ant Routing Algorithm for MANETs

- Route discovery: forward ants



- Route discovery: backward ants



Mesut Günes; Udo Sorges & Imed Bouazizi.  
**ARA - The Ant-Colony Based Routing Algorithm for MANETs.** Proceedings of the 2002 ICPP Workshop on Ad Hoc Networks (IWAHN 2002), Los Alamitos, CA, USA, 2002.

## ARA – Ant Routing Algorithm for MANETs

- Choosing next hop:

$$p_{i,j} = \begin{cases} \frac{\varphi_{i,j}}{\sum_{j \in N_i} \varphi_{i,j}} & \text{if } j \in N_i \\ 0 & \text{if } j \notin N_i \end{cases}$$

- Probabilities of next hops sum to 1:

$$\sum_{j \in N_i} p_{i,j} = 1, \quad i \in [1, N]$$

- Leave pheromone when choosing an edge:

$$\varphi_{i,j} := \varphi_{i,j} + \Delta\varphi$$

- Decrease pheromone level with time (evaporate):

$$\varphi_{i,j} := (1 - q) \cdot \varphi_{i,j}, \quad q \in (0, 1]$$

Copyright: Anna Förster 2009

## ARA – Ant Routing Algorithm for MANETs

- Route discovery with forward/backward ants
- Data packets follow the pheromones in one of two modes:

- Greedy:

$$p_{d,j}^i = \begin{cases} 1 & \text{wenn } \varphi_{d,j}^i = \max_{k \in N_i} \{\varphi_{d,k}^i\} \\ 0 & \text{sonst} \end{cases}$$

- Probabilistic:

$$p_{d,j}^i = \begin{cases} \frac{\varphi_{d,j}^i}{\sum_{k \in N_i} \varphi_{d,k}^i} & \text{wenn } j \in N_i \\ 0 & \text{wenn } j \notin N_i \end{cases} \quad \begin{aligned} \varphi_{i,j} &:= \varphi_{i,j} + \Delta\varphi \\ \varphi_{i,j} &:= (1 - q) \cdot \varphi_{i,j}, \quad q \in (0, 1] \end{aligned}$$

Copyright: Anna Förster 2009

## ARA – Ant Routing Algorithm for MANETs Discussion

- What are the consequences of greedy/probabilistic routing to the development of pheromone levels on the edges of the graph?
- How failure resistant are both versions?
- Is there any better option for taking routing decisions?
- Compare the pheromone levels against Q-values

Leave pheromone when choosing an edge

$$\varphi_{i,j} := \varphi_{i,j} + \Delta\varphi$$

Evaporate pheromone level

$$\varphi_{i,j} := (1 - q) \cdot \varphi_{i,j}, \quad q \in (0, 1]$$

Copyright: Anna Förster 2009

Greedy routing

$$p_{d,j}^i = \begin{cases} 1 & \text{wenn } \varphi_{d,j}^i = \max_{k \in N_i} \{\varphi_{d,k}^i\} \\ 0 & \text{sonst} \end{cases}$$

Probabilistic routing

$$p_{d,j}^i = \begin{cases} \frac{\varphi_{d,j}^i}{\sum_{k \in N_i} \varphi_{d,k}^i} & \text{wenn } j \in N_i \\ 0 & \text{wenn } j \notin N_i \end{cases}$$

## Further options: AntHocNet

- Separates route management from data routing
  - Ants traverse the network continuously and update pheromone levels (using pervious formulas)
  - Data follows highest pheromone levels only
- Discussion:
  - More communication overhead
  - Better resilience against mobility and failures

G. Di Caro, F. Ducatelle, and L.M. Gambardella. **AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks**. European Transactions on Telecommunications, 16:443–455, 2005.

Copyright: Anna Förster 2009

## Ant Colony Optimization for WSNs

Take away

**Needs careful design and planning**

**Good choice when topology is very dynamic**

Copyright: Anna Förster 2009

## Genetic Algorithms for WSNs

State of the art

Copyright: Anna Förster 2009



## Genetic algorithms for WSNs

- High memory and processing requirements
- Inflexible to environmental changes
- Better suited for static, centralized problems
  - Optimal scheduling of a one-hop static network
  - Optimal placement (with error guard) of sensors
  - Optimal transmission radius (with error guard) for each node
  - Optimal clustering of the network
  - ...

Copyright: Anna Förster 2009

## Genetic algorithms for WSNs

- Environmental model needed to compute fitness of given network instance
  - Simple, perfect communication model
  - Simulation
  - Real system (very costly when used with machine learning!)
- Error analysis of solution required
  - Evaluate the effect of expected failures in the network on the total behavior

Copyright: Anna Förster 2009

## Genetic algorithms for WSNs

- Multihop routing (aggregation trees and their usage):
  - O. Islam and S. Hussain. **An intelligent multi-hop routing for wireless sensor networks.** In Proceedings of the IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), pages 239–242, Hong Kong, 2006.
- Optimal clustering:
  - S. Hussain, A. W. Matin, and O. Islam. **Genetic algorithm for energy efficient clusters in wireless sensor networks.** In Proceedings of the 4th International Conference on Information Technology (ITNG), pages 147–154, Las Vegas, Nevada, USA, 2007.
- Optimal scheduling in a static, reliable environment:
  - Q. Tang, N. Tummala, S.K.S. Gupta, and L. Schwiebert. **Communication scheduling to minimize thermal effects of implanted biosensor networks in homogeneous tissue.** IEEE Transactions on Biomedical Engineering, 52(7):1285–1294, 2005.

Copyright: Anna Förster 2009

## Genetic Algorithms for WSNs

Take away

**Very useful to solve complex multi-parameter problems**

**Suited for static, centralized environments**

Copyright: Anna Förster 2009

# Challenges for ML in WSNs

Open discussion

Copyright: Anna Förster 2009

## Main challenges for current algorithms

- Efficient on-node implementation
- Real-world deployments

Copyright: Anna Förster 2009

## Future applications

- Automatic parameter learning for individual protocols
- Automatic parameter learning for cross-layer architectures
- Distributed implementations of PSO and neural networks
- Efficient and robust routing protocol design with ant colony optimization
- Applications to neighborhood management and MAC protocols

Copyright: Anna Förster 2009

# Optimizing Communications in Wireless Sensor Networks with Machine Learning

## Bibliography and Further Readings

Anna Förster  
University of Lugano, Switzerland  
anna.foerster@ieee.org

October 11th, 2009, Sliema, Malta

Some recent surveys give an overview of applications of machine learning and computational intelligence for wireless sensor networks [1, 12, 18, 23, 29, 36, 44, 58].

## 1 Neural Networks

General information about neural networks can be found in [3, 46].

**Energy Aware Routing and Clustering.** Neural networks have been widely applied in WSNs. SIR [4] is an energy-efficient routing protocol, which assigns a neural network to each node in the network. The nodes use beacons to find out the quality of links to their neighbors and the information is fed into the NN to learn the quality of the links. Routing is performed based on a modified Dijkstra shortest-path algorithm from a source to a single sink using the learnt link quality. The protocol performs well compared to Directed Diffusion [57], but results in a high beacon overhead. Additionally, the implementation of a neural network on each of the nodes has high memory requirements and might be hard on memory-restricted sensor hardware.

**Scheduling and Medium Access Protocols.** A centralized neural network has been applied to solve the optimal TDMA scheduling for a WSN in [55]. However, a centralized computation of schedules does not take into account link asymmetry, link and node failures, mobility etc. Additionally, it incurs high communication overhead to disseminate the schedules to the nodes.

## 2 Decision trees and case-based reasoning

A formal description can be found in [40].

**Energy Aware Routing and Clustering.** An application to link quality classification in WSNs is presented in [66]. The authors use simple rules to classify links into good and bad, based on the RSSI level of received packets, buffer sizes, etc. The computation is done centrally on the base station and the data model is disseminated to all nodes in the network.

### 3 Reinforcement learning

Reinforcement learning (RL) [31, 62] is biologically inspired, where the learning agent acquires its knowledge by actively exploring its environment.

**Energy Aware Routing and Clustering.** One of the fundamental and earliest works in packet routing using machine learning is Q-Routing [8]. The authors describe a very simple, Q-Learning based algorithm, which learns the best paths considering the least latency to the destinations. Possible actions are next hops at the nodes, and a Q-Value is assigned to each pair (*sink*, *neighbor*) representing the time which a packet needs through this neighbor to reach the sink. Simulations proved the algorithm to be efficient under high network loads and to perform also well under changing network topologies. Although the approach was developed for wired, packet-switched networks, it inspired a lot of works in the wireless ad hoc and WSN communities, because it is fully distributed. A recent implementation on Crossbow motes [11] has demonstrated its practicality.

Many other routing protocols have been inspired from Q-Routing [2, 6, 25, 37, 43, 53, 67, 68]. The main difference between them is the used cost metric for routing. Delivery time is used in [37, 59], maximum compression paths are learnt in [6, 25, 67], and geographic-based routing is implemented in [2, 53]. A novel cost metric is used by [43], where the routing protocol learns to avoid “important” nodes: nodes, which after failing might disconnect the network. Neighboring nodes exchange information about their importance (computed locally at the nodes based on full topology information) and the best routes (with least important nodes on them) are learnt. A more general cost function is defined in [68], where any combination of number of hops, delay, and remaining energy on the nodes can be applied.

Another difference between the above approaches is the used reinforcement learning algorithm. The authors of [37] use dual reinforcement learning, which gives rewards not only for previous actions, but also to next ones. Thus, learning converges faster and the protocol shows better performance. Q-Learning is used by [2, 53, 67, 68, 19]

An energy-aware multicast routing protocol based on Q-Learning called FROMS is presented in [19, 21]. Its goal is to minimize the energy spent in a network, while delivering packets to many sinks simultaneously. The idea is based on an optimal broadcast Steiner tree, where a minimum number of broadcasts are needed to deliver one packet from an independent source to all sinks.

Team-partitioned, opaque-transition reinforcement learning (TPOT-RL) has been developed for simulated robotic soccer [60] and applied to packet routing [59]. It allows a team of independent learning agents to collaboratively learn a shared task, like soccer playing. It differs from traditional RL in its value function, which is partitioned among the agents and each agent learns only the part of it directly relevant to its localized actions. Also, the environment is *opaque* to the agents, which means that they have no information about the next possible actions of their mates or their goodness.

A formal definition of RL in a distributed environment and a learning algorithm is given in [17]. It presents a reinforcement learning algorithm, designed especially for solving the point-to-point routing problem in MANETs. Collaborative RL (CRL) is greatly based on Q-Learning, but uses also a decay function (similar to pheromone evaporation in ACO, see further Section 4) to better meet the properties of ad-hoc networks.

An additional contribution of [25] beside the Q-Learning routing protocol is the automatic learning of the optimal values of the parameters of the algorithm with a Bayesian exploration strategy. The paper presents an idea which can be applied to all other RL-based algorithms, which need parameter pre-setting and should be further explored and refined.

The setting of [65] is similar to those presented above: many source nodes are sending data to a single base station. The algorithm takes into account the aggregation ratio, the residual energy on the nodes, the hop cost to the base station and the link reliability between the nodes. The algorithm runs in learning episodes. The learning agents are again the nodes and Q-Values are assigned to each possible next hop at each node. During each episode, the current Q-Values are used to route a packet to the base station. At each hop, the full hop information is appended to the packet (residual energy, rewards, etc.). Rewards are generated at the base station. When the base station has enough such packets (undefined how many), it calculates the Q-Values offline for the nodes in the network and disseminates them via a network-wide broadcast.

Clique [20] solves the problem by avoiding all-over the cluster head selection process. It assumes the nodes in the WSN have some a-priori clustering information, like a simple geographic grid or room or floor information in a building. It further assumes that the possibly multiple sinks in the network announce themselves through network-wide data requests. During the propagation of these requests all network nodes are able to gather 1-hop neighborhood information consisting of the remaining energy, hops to individual sinks and cluster membership. When data becomes available for sending, nodes start routing it directly to the sinks. At each intermediate node they take localized decisions whether to route it further to some neighbor or to act as a cluster head and aggregate data from several sources. Clique uses Q-Learning to select the best decision.

Although all of the above studies show promising results from applying various reinforcement learning algorithms to routing in WSNs, none of them has reached the state of a mature communication protocol with implementation and evaluation in a realistic simulation and real hardware environment. Their evaluations are rather preliminary and concentrate on a few of their properties,

leaving out important questions about overhead and efficient implementation.

**Scheduling and Medium Access Protocols.** Actor Critic Algorithm [47] is a early reinforcement learning algorithm, where the policy is detached from the learnt action values. In current RL algorithm like Q-Learning the policy is fully dependent on the learnt Q-Values, which represent the current state of the value function. This incurs search overhead when the best Q-Value needs to be found. In actor critic algorithm a separate table (called the *actor*) can be defined together with the value table (called the *critic*) to speed up action selection. This algorithm has been applied for example to point to point communication in sensor networks [42]. The goal of the algorithm is to maximize throughput per total consumed energy in a sensor network, based on node-to-node communication. Given its current buffer size and last channel transmission gain, the node decides the best modulation level and transmit power to maximize the total throughput per consumed energy. For this, the authors use the standard RL algorithm and test their algorithm on a two-node and multinode scenarios. Unfortunately no comparison to other state-of-the-art protocols is presented in order to evaluate the gain of the RL algorithm.

RL-MAC [38] applies reinforcement learning to adjust the sleeping schedule of a MAC protocol in a WSN setting. The MAC protocol is very similar in its idea to the other WSN MAC protocols such as S-MAC or T-MAC. It divides the time into frames and the frames into slots, where each node is allowed to transmit messages only during its own reserved slot. However, unlike other protocols, it changes the duration of the frames and slots according to the current traffic. At the beginning of its reserved slot, the node first transmits some control information, including also a reward for the other nodes. The reward function depends on the number of waiting messages on the nodes and on the number of successfully transmitted messages during the reserved slot. The paper reports higher data throughput and lower energy expenditure compared to S-MAC.

COORD, a distributed reinforcement learning based solution to achieve best coverage in a WSN is presented in [51]. The goal of the algorithm is to cooperatively find a combination of active and sleeping sensor nodes in a sensor network, which is still able to perform full covered sensing of the desired phenomena. For this the authors propose three similar approaches, all based on Q-Learning. The possible actions are two: transitioning from sleeping to active mode and back. The sensor network is divided into a rectangular grid and the goal is to cover each grid vertex by some sensors, best by exactly one. A Q-Value is assigned to each grid vertex, which represents the number of sensor nodes, currently covering this vertex. In each run of the algorithm, each node evaluates its current Q-Value table with all grid vertices it covers and takes an action. After that, all nodes evaluate again their Q tables and so on.

The other two solutions are very similar and the results they show are also comparable. A comparison to some state-of-the-art approach is not provided and thus the results cannot be properly evaluated. Also, a clear protocol implementation is missing, leaving open many questions about coordination and



exchange of Q-Values and the states of the grid vertices. However, the approach is fully distributed and can be run online if needed. Also, it shows a nice modeling work of converting a centralized problem into a distributed one and solving it with RL.

## 4 Swarm Intelligence

A good introduction to swarm intelligence for wireless communications is presented in [32]. A more general overview of Swarm Intelligence can be found in [33, 15, 16].

**Energy Aware Routing and Clustering.** Four variants of PSO are proposed for energy aware clustering in [24]. The difference between them are the PSO parameters - initial speed, acceleration, etc. Although PSO is a distributed algorithm, here the algorithm is centralized and run on the base station with full topology information about the network. The algorithm is based on a simple idea that for a group of nodes that lie in a neighborhood, the node closest to the base station becomes the clusterhead. The approach has some drawbacks: Clustering depends solely on the physical distribution of nodes and is centralized. Thus, in case of failures or any topology changes, the new information needs to be gathered at the base station and clustering needs to be re-computed.

A novel clustering approach for WSNs called CRAWL is defined in [5] with the use of soldier ants. Biological soldier ants that have the support of other soldier ants are found to be more aggressive in nature. An ant is observed to exhibit higher eagerness to fight when it is amidst strong ants. This fact inspires the collaborative clustering algorithm for wireless sensor network longevity (*CRAWL*) that possesses good scalability and adaptability features. Here, each node has an *Eagerness* value to serve as a clusterhead, which is computed based on its own remaining battery and the remaining batteries of its neighbors. At regular intervals, each node computes its *Eagerness* value and broadcasts it over the network. The node that has the highest *Eagerness* value decides to act as a clusterhead, and the other nodes accept it. The clusterhead floods the new clustering information, which helps other nodes to readjust their power levels just enough for them to transmit to the clusterhead.

The method assures that only the nodes that have sufficient energy in their reservoir, and have strong neighbors, opt to become clusterheads. The algorithm has a significant communication overhead due to the fact that each node has to flood its *Eagerness* value at regular intervals. In addition, the traffic of packets might flow away from a sink node just because a node in that direction has higher *Eagerness*. Thus, the algorithm is sub-optimal in terms of minimizing energy expenditure of individual nodes, but optimal in terms of making effective use of the energy available to the whole network.

AntNet [13] is an ACO application in communication networks used to find near-optimal routes in a communication graph without global information. The agents are divided into *forward* and *backward* ants. Forward ants are initial-

ized at the data source and sent to all known destinations at regular intervals. They travel through the network graph by *randomly* choosing the next hop and leave pheromones on their way. The more ants have chosen the same path the higher the pheromone level of that path. During their travel, forward ants gather routing information, indicating the arrival time at each node on their way. At destination arrival, the forward ants are transformed into backward ants and use the cached route they have traveled to traverse the same route again and to update the pheromone tables according to the gathered routing information. Details of this computation can be found in [13, 14]. A decay function is implemented as evaporation of the pheromone levels, indicating which routes are the most freshly used ones. The version of AntNet for MANETs is called AntHocNet [14] and is developed by the some of the authors of AntNet.

AntNet and AntHocNet use both reactive path setup and proactive path maintenance for single source - single sink. However, the approach requires ants to be traveling independent from data packets and even to trace each path twice (forward and backward), which causes a great overhead and is not well suited for energy-restricted WSNs. Nevertheless, the method is fully distributed and is the one best explored and described in the literature for using swarm intelligence in wireless networks.

MANSI [54] (Multicast for Ad Hoc Networks with Swarm Intelligence) is a multicast routing protocol for MANETs, based on swarm intelligence. The protocol is similar to traditional multicast protocols, where a core node initiates the building of the multicast tree through a forward Join Request Packet and a backward Join Reply Packet. However, nodes different from the core send ants into the network at regular intervals to explore the network for better routes to the core, leaving routing information (pheromones) on their way. This information is later used by following ants for opportunistically selecting their next hops. The approach is similar to AntHocNet [14], however, optimization is applied to multicast instead of unicast routing.

In [41], the authors propose an AntHocNet [14] based approach for routing in a sensor network installed in a building. Its main disadvantage is that the returning ants in the network create unnecessary overhead for a sensor network.

Ant-Based Control [50] is similar to AntNet in many aspects, but also has some important differences. There is only one class of ants, started at regular intervals at the data sources, traversing the network probabilistically and updating the routing tables as they travel to the destinations. Once reaching their destination, the ants are eliminated. The update of the routing tables is thus not based on the trip times to the destination, but rather on the present *lifetime* of the ant, calculated as the delay from its launching node to the present one. Because of its relatively smaller communication overhead (only forward ants), ABC is better suited for energy-restricted scenarios like WSN. However, it is still costly to send ants at regular intervals and the advantages of using it should be carefully evaluated.

UniformAnts [61] presents a simple ant-optimization based technique for finding and maintaining routes in a MANET. Similarly to the original ABC algorithm, it uses only forward ants, updating the probability-based routing

tables on the nodes as the ant travels towards the sink. Two different ant types are used, the difference is how the next hop is selected - greedy or uniformly between all options. The method achieves fairly good results and shares the properties of ABC.

Mobile agents are often mistaken for a machine learning or swarm intelligence approach. However, they refer to the usage of simple, small entities (packets), which traverse the system (in our case the network) and deliver fresh information to the system's nodes. In the case of routing, for example, the agents update routing information (paths or next hops) on the nodes [7, 9, 64]. Although very efficient in some applications (like routing in less mobile scenarios), they cannot be classified as a learning nor as a swarm intelligence algorithm. They represent a good optimization to traditional routing approaches in mobile scenarios. However, they also increase the communication cost for sending the agents.

## 5 Genetic algorithms

General information about genetic algorithms can be found for example in [49].

**Energy Aware Routing and Clustering.** A GA based multi-hop routing technique named *GA-Routing* is proposed in [28] for maximizing network longevity in terms of time to first node death. The proposed GA approach generates aggregation trees, which span all the sensor nodes. Although the best aggregation tree is the most efficient path in the network, continuous use of this path would lead to failure of a few nodes earlier than others. The goal of the study in [28] is to find an aggregation tree, and the number of times a particular tree is used before the next tree comes in force. The spanning trees are modeled as individuals. Simulation results show that GA gives better lifetime than the *single best tree* (SBT) algorithm, and the same lifetime as the cluster based maximum lifetime data aggregation algorithm [10] for small network sizes. However, the algorithm's overhead is not evaluated.

Another application of GA in energy efficient clustering is described in [27]. The proposed GA represents the sensor nodes as bits of chromosomes, cluster-heads as 1 and ordinary nodes as 0. The number of bits in a chromosome is equal to the number of nodes. The fitness of the chromosomes are computed based on the distances between the nodes and the cluster heads, the distance between the cluster heads and the sink and the energy spent to deliver packets to the sink. The results show that the GA approach possesses better energy efficiency than do hierarchical cluster based routing (HCR) and LEACH [45]. However, clustering overhead is not considered.

There are also some other similar ideas based on GAs, where a base station computes the optimal routing, aggregation or clustering scheme for a network based on the information about the topology, remaining energy on the nodes, etc. [26, 39, 63]. Such algorithms are only feasible if the network is expected to have a static topology, perfect communication, symmetric links and constant

energy. Under these restrictions, a centrally computed routing or aggregation tree makes sense and is probably easier to implement. However, these properties are in conflict with the nature of WSNs.

**Scheduling and Medium Access Protocols.** A model based on GA is proposed for sleep scheduling of nodes in a randomly deployed large scale WSN in [56]. Such networks deploy a large number of redundant nodes for better coverage, and how to manage the combination of nodes for a prolonged network operation is a major problem. The scheme proposed in the article divides the network life into rounds. In each round, a set of nodes is kept active and the rest of the nodes are put in sleep mode. It is ensured that the set of active nodes has adequate coverage and connectivity. When some of the active nodes die, blind spots appear. At this time, all nodes are woken up for a decision on the next set of nodes to remain active in the next round. This is clearly a multi-objective optimization problem. The first objective is to minimize the overall energy consumption of the active set, and the second objective is to minimize the number of active nodes. Again, gathering the topology information on a single base station is critical and not feasible in a realistic scenario.

A similar scheduling problem called the active interval scheduling problem in hierarchical WSNs for long-term periodical monitoring is introduced in [30]. In this scenario, nodes are partitioned into clusters with local cluster heads, which dictate active intervals to the nodes. Active intervals need to be coordinated among clusters to avoid intra-cluster interference and minimized to minimize energy expenditure. Again, the proposed algorithm is centralized and does not take into account crucial WSN properties such as failures.

## 6 Heuristic Search

General information can be found in [34, 35].

**Energy Aware Routing and Clustering.** Real time heuristic search methods are very well suited for wireless ad-hoc scenarios - the nodes in the network can be modeled as the agent states, the packets as the agents and the information available at the nodes about their one-hop neighbors can be used for evaluating the search neighborhood. LRTA\* is applied to routing in ad-hoc networks in [48, 52] with good results. However, the need of a global heuristic limits the applicability of the algorithm in distributed environments.

## 7 Implementation of Machine Learning for WSNs

Experience from implementing a Q-Learning based routing algorithm on real hardware is presented in [22].

Standard libraries for implementing various ML techniques are Spider for

Matlab<sup>1</sup>, Pybrain for Python<sup>2</sup>, many others for neural networks and decision trees.

## References

- [1] P. Arabshahi, A. Gray, I. Kassabalidis, El M. A. Sharkawi, R. J. Marks, A. Das, and S. Narayanan. Adaptive routing in wireless communication networks using swarm intelligence. In *Proceedings of the 19th AIAA International Communications Satellite Systems Conference (AIAA-ICSSC)*, page 9pp., Toulouse, France, 2001.
- [2] R. Arroyo-Valles, R. Alaiz-Rodrigues, A. Guerrero-Curieses, and J. Cid-Suiero. Q-probabilistic routing in wireless sensor networks. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 1–6, Melbourne, Australia, 2007.
- [3] P.F. Baldi and K. Hornik. Learning in linear neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(4):837–858, 1995.
- [4] J. Barbancho, C. León, J. Molina, and A. Barbancho. Giving neurons to sensors: QoS management in wireless sensors networks. In C. Leon, editor, *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 594–597, Prague, Czech Republic, 2006.
- [5] S. Bashyal and G.K. Venayagamoorthy. Collaborative routing algorithm for wireless sensor network longevity. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 515–520, Melbourne, Australia, 2007.
- [6] P. Beyens, M. Peeters, K. Steenhaut, and A. Nowe. Routing with compression in wireless sensor networks: A Q-learning approach. In *Proceedings of the 5th European Workshop on Adaptive Agents and Multi-Agent Systems (AAMAS)*, page 12pp., Paris, France, 2005.
- [7] E. Bonabeau, F. Henaux, S. Guérin, D. Snyers, P. Kuntz, and G. Theraulaz. Routing in telecommunications networks with “smart” ant-like agents. In *Proceedings of the 2nd International Workshop on Intelligent Agents for Telecommunications Applications (IATA)*, pages 60–71, Paris, France, 1998.
- [8] J. A. Boyan and M. L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in Neural Information Processing Systems*, 6:671–678, 1994.

---

<sup>1</sup><http://www.kyb.tuebingen.mpg.de/bs/people/spider/>

<sup>2</sup><http://www.pybrain.org/>

- [9] D. Camara and A. A. F. Loureiro. A novel routing algorithm for ad hoc networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*, page 8pp., Hawaii, USA, 2000.
- [10] K. Dasgupta, K. Kalpakis, and P. Namjoshi. An efficient clustering-based heuristic for data gathering and aggregation in sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking (WCNC)*, volume 3, pages 1948–1953, New Orleans, USA, 2003.
- [11] M.T. Dela Cruz, M. Whyte, Z. Yu, and T. Hanselmann. Q learning routing protocol. Display demonstration of real hardware implementation at the International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Melbourne, Australia, 2007.
- [12] M. Di and E.M. Joo. A survey of machine learning in wireless sensor networks. In *Proceedings of the 6th International Conference on Information, Communications and Signal Processing (ICICS)*, pages 1–5, Singapore, 2007.
- [13] G. Di Caro and M. Dorigo. AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.
- [14] G. Di Caro, F. Ducatelle, and L.M. Gambardella. AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16:443–455, 2005.
- [15] M. Dorigo. *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Italy, 1992.
- [16] M. Dorigo and T. Stuetzle. *Ant Colony Optimization*. MIT Press, 2004.
- [17] J. Dowling, E. Curran, R. Cunningham, and V. Cahill. Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing. *IEEE Transactions on Systems, Man and Cybernetics*, 35(3):360–372, 2005.
- [18] A. Förster. Machine learning techniques applied to wireless ad-hoc networks: Guide and survey. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 365–370, Melbourne, Australia, 2007.
- [19] A. Förster and A. L. Murphy. FROMS: Feedback routing for optimizing multiple sinks in WSN with reinforcement learning. In *Proceedings 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 371–376, Melbourne, Australia, 2007.

- [20] A. Förster and A. L. Murphy. CLIQUE: Role-Free Clustering with Q-Learning for Wireless Sensor Networks. In *Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS)*, Montreal, Canada, 2009.
- [21] A. Förster and A. L. Murphy. FROMS: A Failure Tolerant and Mobility Enabled Multicast Routing Paradigm with Reinforcement Learning for WSNs. Technical Report TR 2009/04, University of Lugano, June 2009.
- [22] A. Förster, A. L. Murphy, J. Schiller, and K. Terfloth. An Efficient Implementation of Reinforcement Learning Based Routing on Real WSN Hardware. *Proceedings of the 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMOB)*, 2008.
- [23] Anna Förster. *Teaching Networks How To Learn: Machine Learning for Data Dissemination in Wireless Sensor Networks*. PhD thesis, University of Lugano, Switzerland, 2009.
- [24] S.M. Guru, S.K. Halgamuge, and S. Fernando. Particle swarm optimisers for cluster formation in wireless sensor networks. In *Proceedings of the 2nd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 319–324, Melbourne, Australia, 2005.
- [25] S. Hao and T. Wang. Sensor networks routing via bayesian exploration. In *Proceedings of the 31st IEEE Conference on Local Computer Networks (LCN)*, pages 954–955, Tampa, FL, USA, 2006.
- [26] S. Hussain and A. W. Matin. Hierarchical cluster-based routing in wireless sensor networks. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN)*, page 2pp., Nashville, TN, USA, 2006.
- [27] S. Hussain, A. W. Matin, and O. Islam. Genetic algorithm for energy efficient clusters in wireless sensor networks. In *Proceedings of the 4th International Conference on Information Technology (ITNG)*, pages 147–154, Las Vegas, Nevada, USA, 2007.
- [28] O. Islam and S. Hussain. An intelligent multi-hop routing for wireless sensor networks. In *Proceedings of the IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pages 239–242, Hong Kong, 2006.
- [29] S. S. Iyengar, H.-C. Wu, N. Balakrishnan, and S. Y. Chang. Biologically inspired cooperative routing for wireless mobile sensor networks. *IEEE Systems Journal*, 1(1):29–37, 2007.
- [30] M.H. Jin, W.Z. Liu, D.F. Hsu, and C.Y. Kao. Compact genetic algorithm for performance improvement in hierarchical sensor networks management.

- In *Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN)*, page 6pp., Las Vegas, Nevada, USA, 2005.
- [31] L.P. Kaelbling, M.L. Littman, and A.P. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
  - [32] I. Kassabalidis, M. A. ElSharkawi, R. J. Marks, P. Arabshahi, and A. A. Gray. Swarm intelligence for routing in communication networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, pages 3613–3617, San Antonio, TX, USA, 2001.
  - [33] J. Kennedy and R.C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2001.
  - [34] S. Koenig. Agent-centered search. *AI Magazine*, 22(4):109–131, 2001.
  - [35] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, 1990.
  - [36] S.B. Kulkarni, A. Förster, and G.K. Venayagamoorthy. A survey on applications of computational intelligence for wireless sensor networks. *IEEE Communications Surveys Tutorials*, to be published, 2009.
  - [37] S. Kumar and R. Miikkulainen. Dual reinforcement Q-routing: An on-line adaptive routing algorithm. In *Proceedings of the Conference on Artificial Neural Networks in Engineering (ANNIE)*, pages 231–238, St. Louis, MI, USA, 1997.
  - [38] Z. Liu and I. Elahany. RL-MAC: A reinforcement learning based MAC protocol for wireless sensor networks. *International Journal on Sensor Networks*, 1(3/4):117–124, 2006.
  - [39] A. W. Matin and S. Hussain. Intelligent hierarchical cluster-based routing. In *Proceedings of the International Workshop on Mobility and Scalability in Wireless Sensor Networks (MSWSN)*, page 9pp., San Francisco, CA, USA, 2006.
  - [40] T.M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
  - [41] R. Muraleedharan and L. A. Osadciw. A predictive sensor network using ant system. In *Proceedings of the SPIE Conference on Digital Wireless Communications*, volume 5440, pages 181–192, Orlando, FL, USA, 2004.
  - [42] C. Pandana and K. J. R. Liu. Near-optimal reinforcement learning framework for energy-aware sensor communications. *IEEE Journal on Selected Areas in Communications*, 23(4):788–797, 2005.
  - [43] C. Pandana and K.J.R. Liu. Robust connectivity-aware energy-efficient routing for wireless sensor networks. *IEEE Transactions on wireless communications*, 7(10):3904–3916, 2008.



- [44] J.B. Predd, S.B. Kulkarni, and H.V. Poor. Distributed learning in wireless sensor networks. *IEEE Signal Processing Magazine*, 23(4):56–69, 2006.
- [45] W. Rabiner-Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*, page 10pp., Hawaii, USA, 2000.
- [46] R. Rojas. *Neural Networks - A Systematic Introduction*. Springer-Verlag, 1996.
- [47] M.T. Rosenstein and A.G. Barto. *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*, chapter Supervised actor-critic reinforcement learning, pages 359–380. John Wiley & Sons, 2004.
- [48] M. Rossi, M. Zorzi, and R. R. Rao. Statistically assisted routing algorithms (SARA) for hop count based forwarding in wireless sensor networks. *Wireless Networks*, 14(1):55–70, 2008.
- [49] S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall International, 2003.
- [50] R. Schoonderwoerd, O.E. Holland, J.L. Bruten, and L.J.M. Rothkrantz. Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, 2:169–207, 1996.
- [51] M.W.M. Seah, C.K. Tham, K. Srinivasan, and A. Xin. Achieving coverage through distributed reinforcement learning in wireless sensor networks. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2007.
- [52] Y. Shang, M. P. J. Fromherz, Y. Zhang, and L. S. Crawford. Constraint-based routing for ad-hoc networks. In *Proceedings of the International Conference on Information Technology: Research and Education (ITRE)*, pages 306–310, Newark, New Jersey, USA, 2003.
- [53] D. Shaoqiang, P. Agrawal, and K. Sivalingam. Reinforcement learning based geographic routing protocol for UWB wireless sensor network. In *Proceedings of the IEEE Global Telecommunications Conference 2007 (GLOBECOM)*, pages 652–656, Washington, DC, USA, 2007.
- [54] C.-C. Shen and C. Jaikaeo. Ad hoc multicast routing algorithm with swarm intelligence. *Mobile Networks and Applications*, 10(1-2):47–59, 2005.
- [55] Y. J. Shen and M. S. Wang. Broadcast scheduling in wireless sensor networks using fuzzy hopfield neural network. *Expert Systems with Applications*, 34(2):900–907, 2008.

- [56] Zhang Shi, Zhang Zhe, Lu Qian-nan, and Chen Jian. Dynamic alliance based on genetic algorithms in wireless sensor networks. In *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, pages 1–4, Wuhan, China, 2006.
- [57] F. Silva, J. Heidemann, R. Govindan, and D. Estrin. *Frontiers in Distributed Sensor Networks*, chapter Directed Diffusion, page 25pp. CRC Press, Inc., 2003.
- [58] K. M. Sim and W. H. Sun. Ant colony optimization for routing and load-balancing: Survey and new directions. *IEEE Transactions on Systems, Man and Cybernetics*, 33(5):560–572, 2003.
- [59] P. Stone. TPOT- RL applied to network routing. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 935–942, San Francisco, CA, 2000.
- [60] P. Stone and M. Veloso. Team-partitioned, opaque-transition reinforcement learning. In *Proceedings of the 3rd annual Conference on Autonomous Agents (AGENTS)*, pages 206–212, Seattle, WA, USA, 1999.
- [61] D. Subramanian, P. Druschel, and J. Chen. Ants and reinforcement learning: A case study in routing in dynamic networks. In *Proceedings of the 15th Joint Conference on Artificial Intelligence (IJCAI)*, pages 832–838, Nagoya, Japan, 1997.
- [62] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, March 1998.
- [63] Q. Tang, N. Tummala, S.K.S. Gupta, and L. Schwiebert. Communication scheduling to minimize thermal effects of implanted biosensor networks in homogeneous tissue. *IEEE Transactions on Biomedical Engineering*, 52(7):1285–1294, 2005.
- [64] C. Tham, S. Marwaha, and D. Srinivasan. Mobile agents based routing protocol for mobile ad hoc networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, volume 1, pages 163–167, Taipei, Taiwan, 2002.
- [65] P. Wang and T. Wang. Adaptive routing for sensor networks using reinforcement learning. In *Proceedings of the 6th IEEE International Conference on Computer and Information Technology (CIT)*, pages 219–224, Bhubaneswar, India, 2006.
- [66] Y. Wang, M. Martonosi, and L.-S. Peh. A supervised learning approach for routing optimizations in wireless sensor networks. In *Proceedings of the 2nd International Workshop on Multi-hop ad hoc networks: from theory to reality (REALMAN)*, pages 79–86, Florence, Italy, 2006.

- [67] B. Yu, P. Scerri, K. Sycara, Y. Xu, and M. Lewis. Scalable and reliable data delivery in mobile ad hoc sensor networks. In *Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1071–1078, Hakodate, Japan, 2006.
- [68] Y. Zhang and M. P. J. Fromherz. A robust and efficient flooding-based routing for wireless sensor networks. *Journal of Interconnection Networks*, 7(4):549–568, 2006.