Coloring Unstructured Radio Networks

Thomas Moscibroda
Roger Wattenhofer

Distributed
Computing
Group

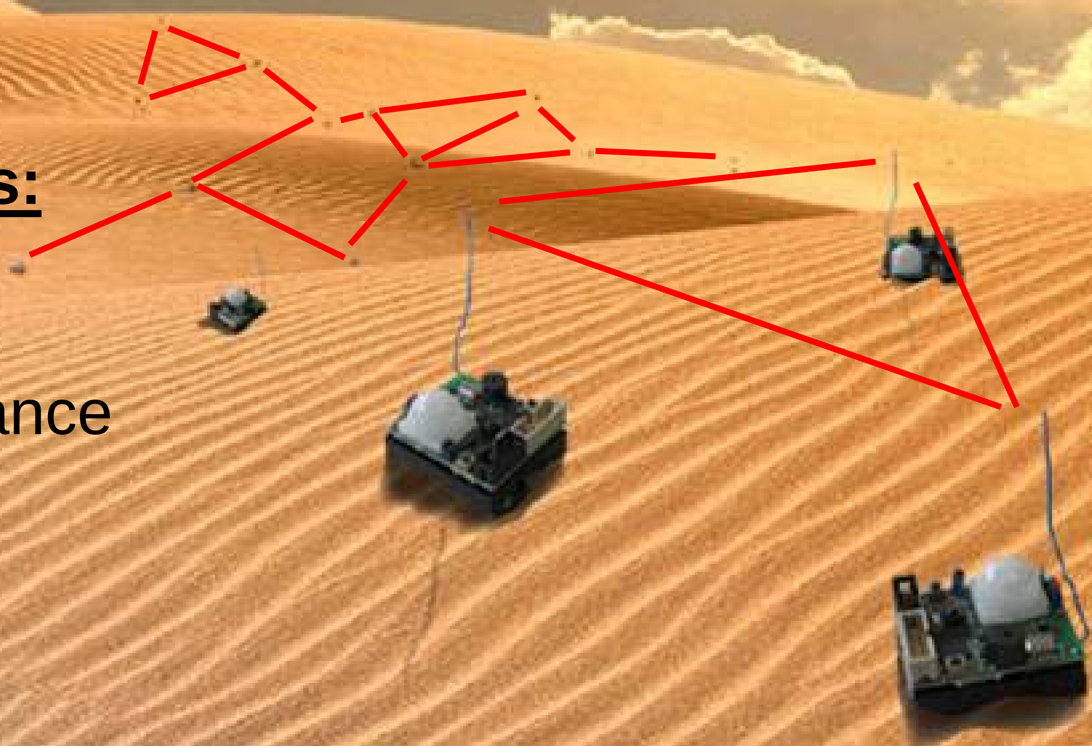
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

SPAA 2005

Wireless Ad Hoc and Sensor Networks

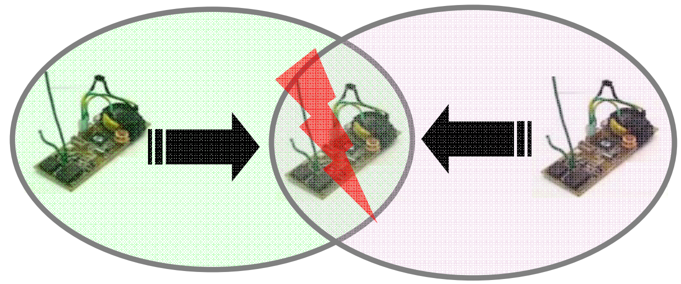
Application Scenarios:

- Data Gathering
- Monitoring, Surveillance
- Disaster Relief
- Many others....

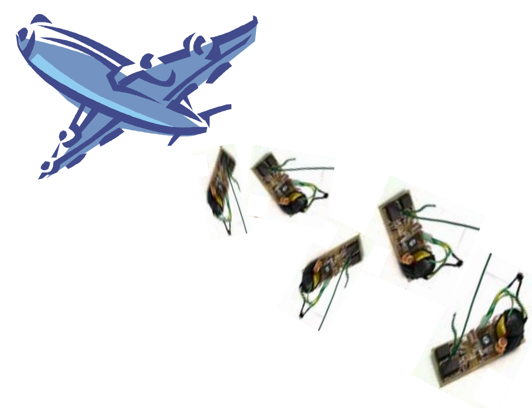


Challenges: Differences to Wired Networks

- No built-in infrastructure
  - Nodes need to set up their own infrastructure (Initially, no available MAC layer)

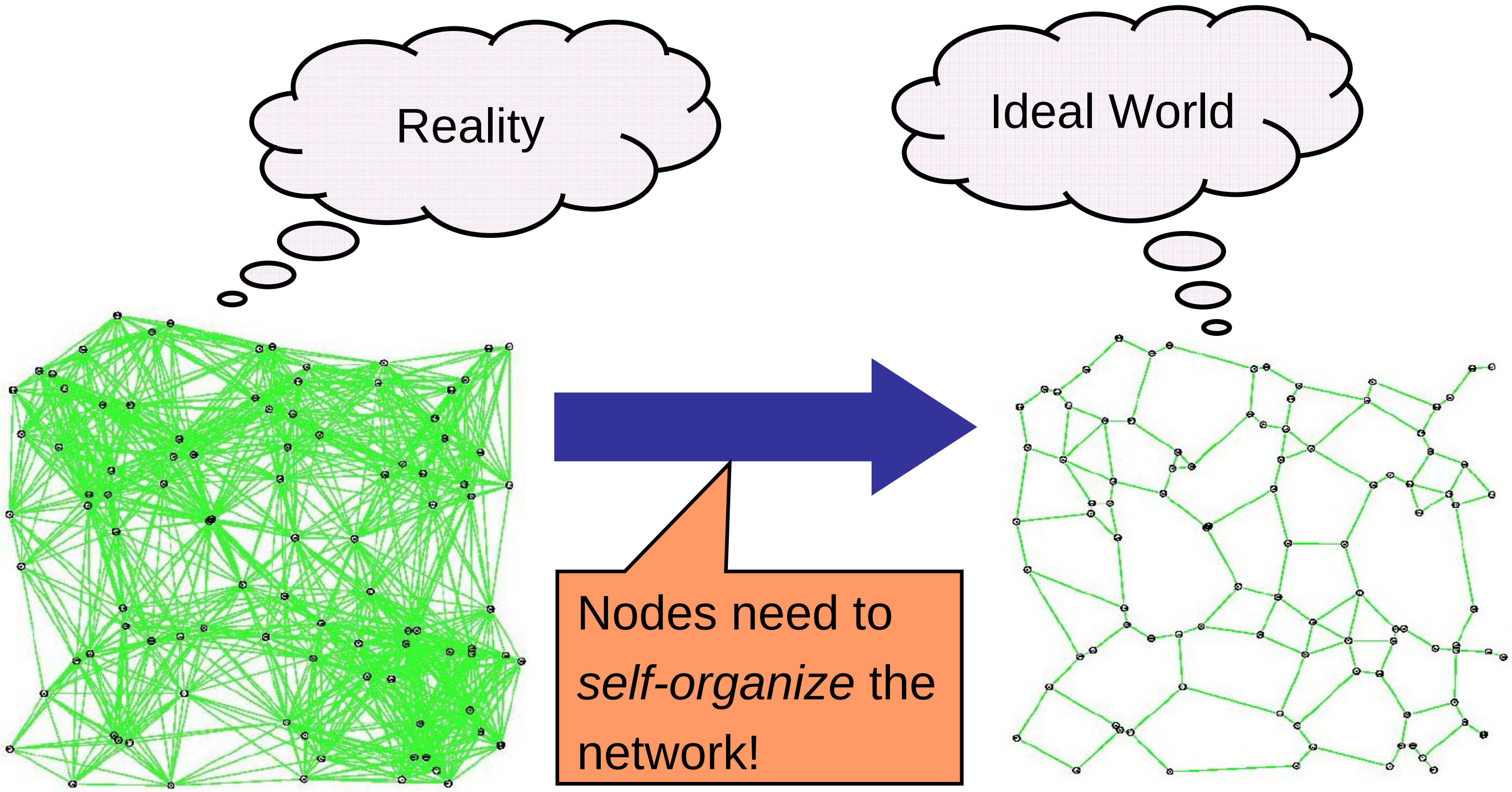


- Communication on shared medium
  - Collisions, Interference,...
- Absence of a-priori knowledge
  - Nodes do not know network topology
  - Nodes do not even know neighbors!



- Energy and memory are scarce
- Mobility, node failures,
- Nodes may be deployed at different times...

Challenges: Differences to Wired Networks

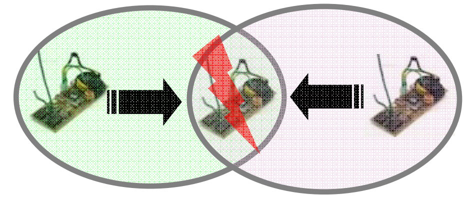


→ Nodes need to bring structure into the network.

→ Nodes must set up a MAC layer!

Distributed Vertex Coloring

- A particularly useful structure is a vertex coloring!
- We model the network as a graph G=(V,E).



Each node assigns itself a color such that,
No two neighbors have the same color

→ A coloring is a step towards a functional MAC layer!

- Frequency Division Multiple Access (FDMA) (identify each color with a frequency)
- Time Division Multiple Access (TDMA) (identify each color with a time-slot)

2-hop coloring yields a collision-free MAC layer

A good coloring should use as few colors as possible!

Distributed Vertex Coloring

- In our paper, we study 1-hop coloring!

- A 1-hop coloring is no MAC layer, but...
  - ...it avoids direct interference between nodes!
  - ...it can be turned into a 2-hop coloring by halving transmission ranges (in dense networks!)
  - ...it induces clusters

- And from a theory point of view...

The distributed complexity of coloring in unstructured radio networks.

Distributed Coloring: Related Work

- Three-coloring a ring in time $O(\log^* n)$ [Cole, Vishkin, 86]
- In time $O(\log^* n)$, rooted trees and bounded degree graphs can be colored with 3 and $\Delta+1$ colors, respectively.
  [Goldberg, Plotkin, Shannon STOC 87]
- All these results are asymptotically optimal [Linial, 92]

- Arbitrary graphs colorable with $\Delta+1$ colors in time $O(\Delta^2 + \log^* n)$ ...
  [Goldberg, Plotkin, Shannon, STOC 87]
- .. or in time $O(\Delta \log n)$ ... [Awerbuch, Goldberg, Luby, Plotkin, FOCS 89]
- Further improvements via network decomposition
  [Panconesi, Srinivasan, 96]

- Coloring for the purpose of obtaining a TDMA scheme
  [Ramanathan, Lloyd, SIGCOMM 92], [Krumke, Marathe, Ravi, 01]

Distributed Coloring: Related Work

In multi-hop radio network models...

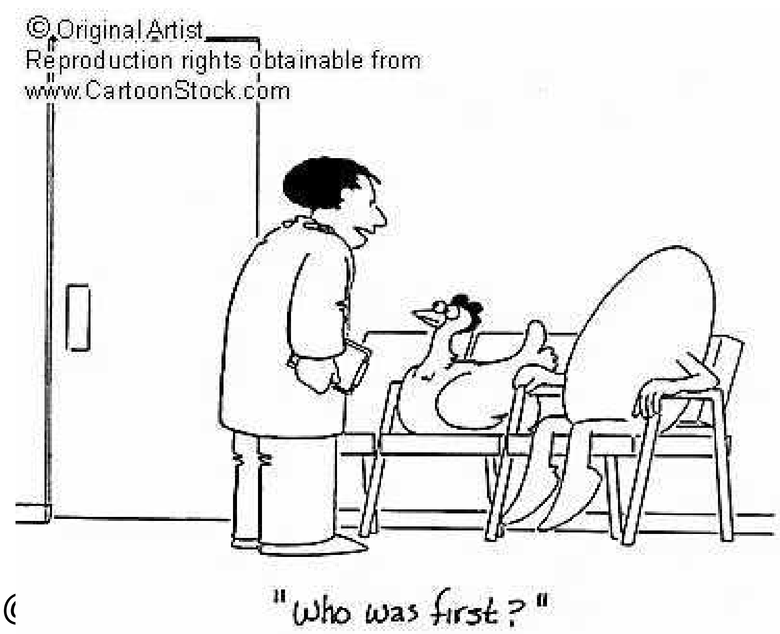
- Communication primitives such as *broadcast* or the *wake-up problem* have been thoroughly studied
- Less is known about local network coordination structures (e.g. colorings)

Distributed Coloring: Related Work

- Most existing algorithms assume...
  - ... point-to-point connections
  → Message-Passing Model
  - ... absence of interference issues
  → Collision detection mechanism
  - ... Synchronous wake-up
  - ... nodes know their neighbors, or even two hop neighbors

Chicken-and-Egg Problem:

1) Coloring algorithms are used to establish a MAC layer
2) Coloring algorithms are based on a MAC layer!

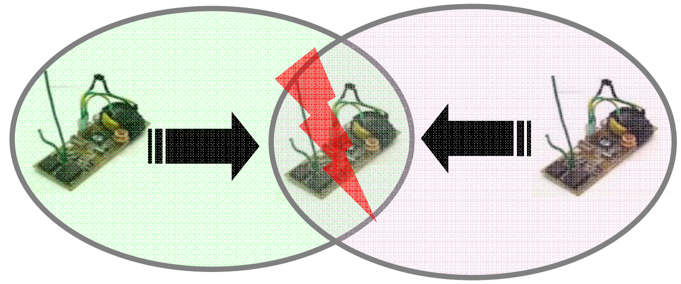


Overview

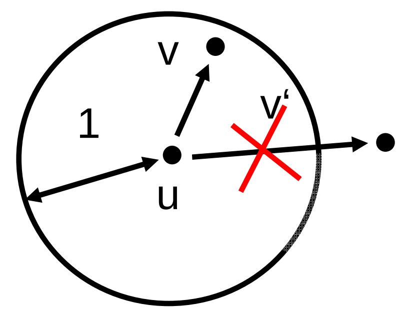
- Coloring in Ad Hoc and Sensor Networks
- Related Work
- Model
- Algorithm & Analysis
- Conclusions & Open Problems

Unstructured Radio Networks – Model (1)

- Multi-Hop
  - Hidden Terminal Problem



- No collision detection
  - Nodes cannot distinguish collisions from ambient noise
  - A sender does not know whether its transmission was correctly received!



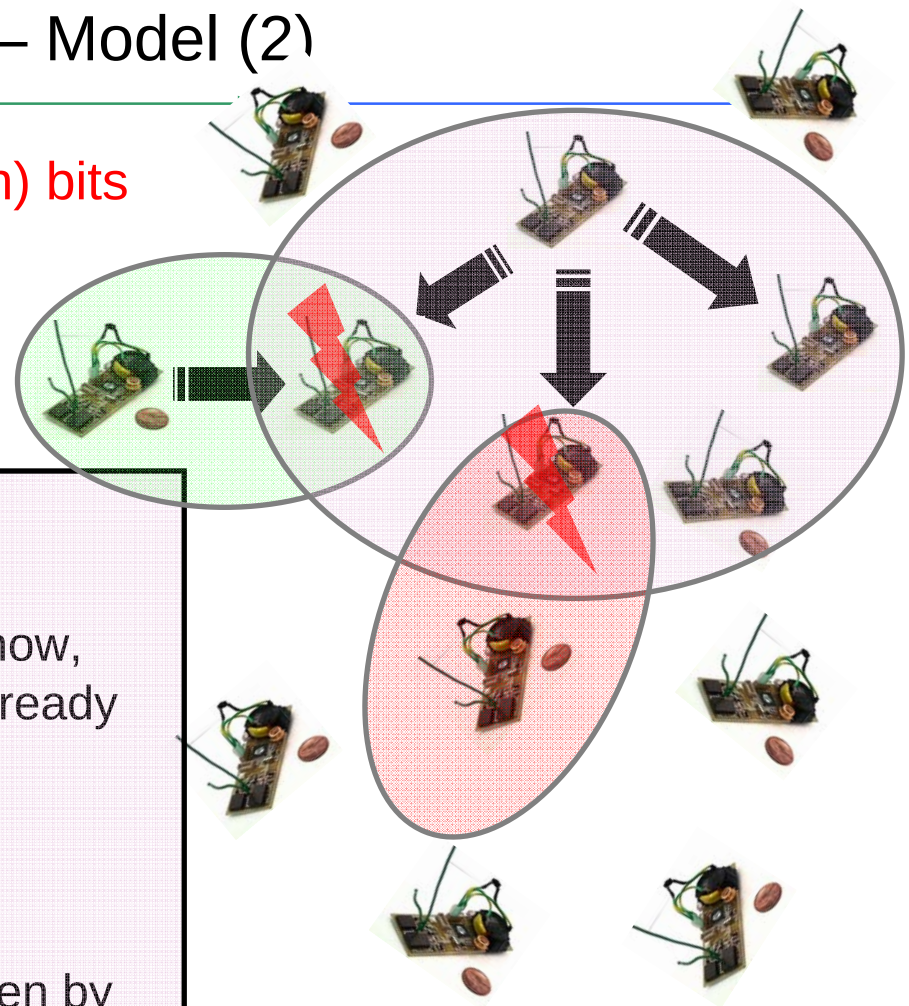
- Unit Disk Graph (UDG)
  - Two nodes can communicate iff Euclidean distance is at most 1

- No knowledge about (the number of) neighbors...

  ... except upper bounds n and Δ for number of nodes in network and the largest degree, respectively.

Unstructured Radio Networks – Model (2)

- Messages are restricted to O(log n) bits
- Nodes can wake-up at any time!
  → Asynchronous wake-up!



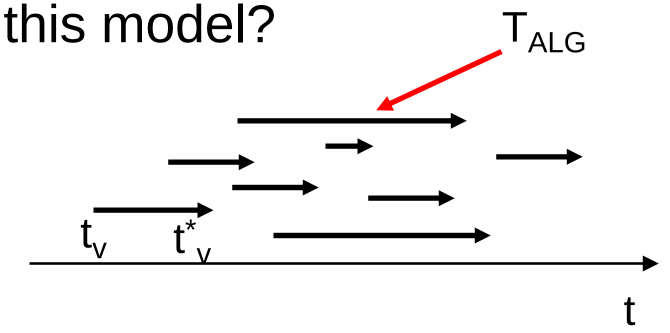
Asynchronous wake-up:

1) When waking up, a node does not know, how many neighboring nodes are already awake!
2) A node does not know when new neighbors wake up!
3) The nodes' wake-up pattern is chosen by an adversary.
4) Sleeping nodes do neither receive nor send messages

different from work on the wake-up problem or broadcast in radio networks

Unstructured Radio Networks – Model (3)

- What are the performance measures in this model?



Running Time:

- Let t_v be the time of node v's wake-up.
- Let t^*_v be the time of v's final, irrevocable decision on a color.

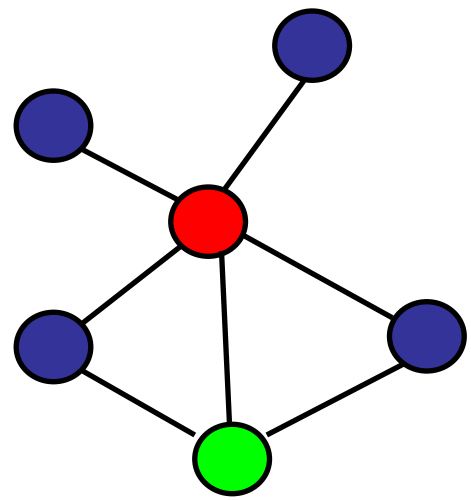
→ The running time of v is: $T_v = t^*_v - t_v$

→ The algorithm's running time is: $T_{ALG} = \max_{v \in V} T_v$

In UDG, $\Omega(\Delta)$ lower bound!

Colors:

- The *maximum color* used by the algorithm
- The *local distribution* of the colors!

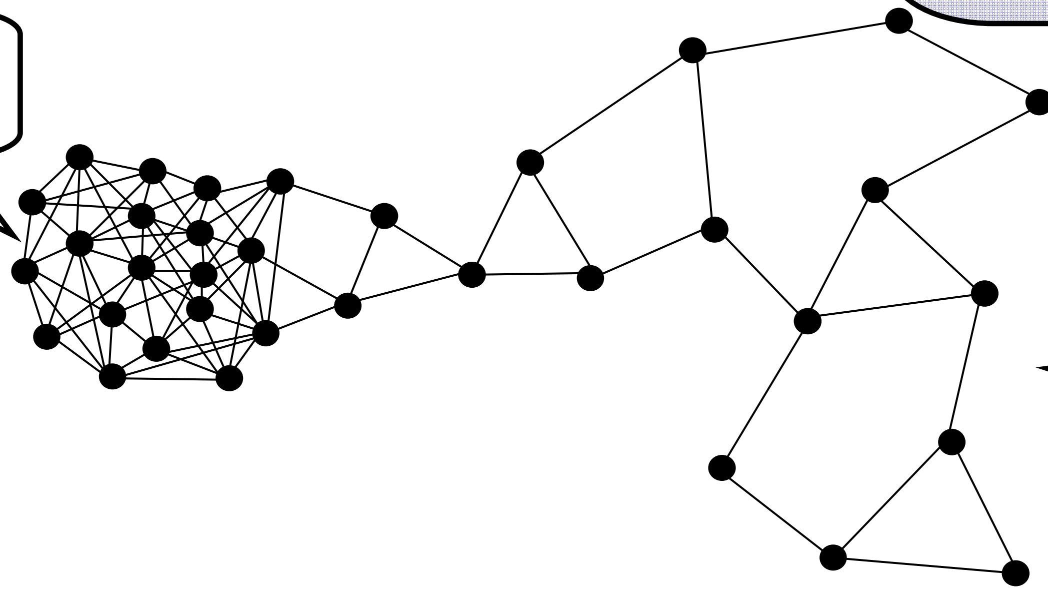


Locality in Vertex Coloring

- A good coloring should....
  - use few colors!
  - use high colors in dense areas only!

This precludes simple probabilistic algorithms In which every node chooses a random color

High colors are needed here!



Use only low colors here!

Highest color should depend on local graph properties!

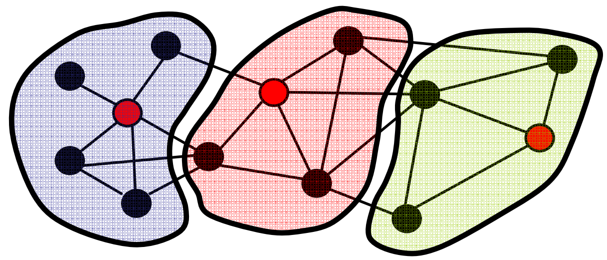
Overview

- Ad Hoc and Sensor Networks
- Clustering
- Model
- Algorithm & Analysis
- Conclusions & Open Problems

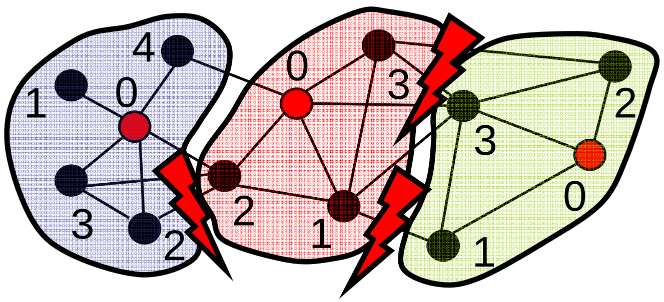
Algorithm Overview (system's view)

- Idea: Color in a two-step process!
- First, nodes select a (sparse) set of leaders among themselves

  → induces a clustering



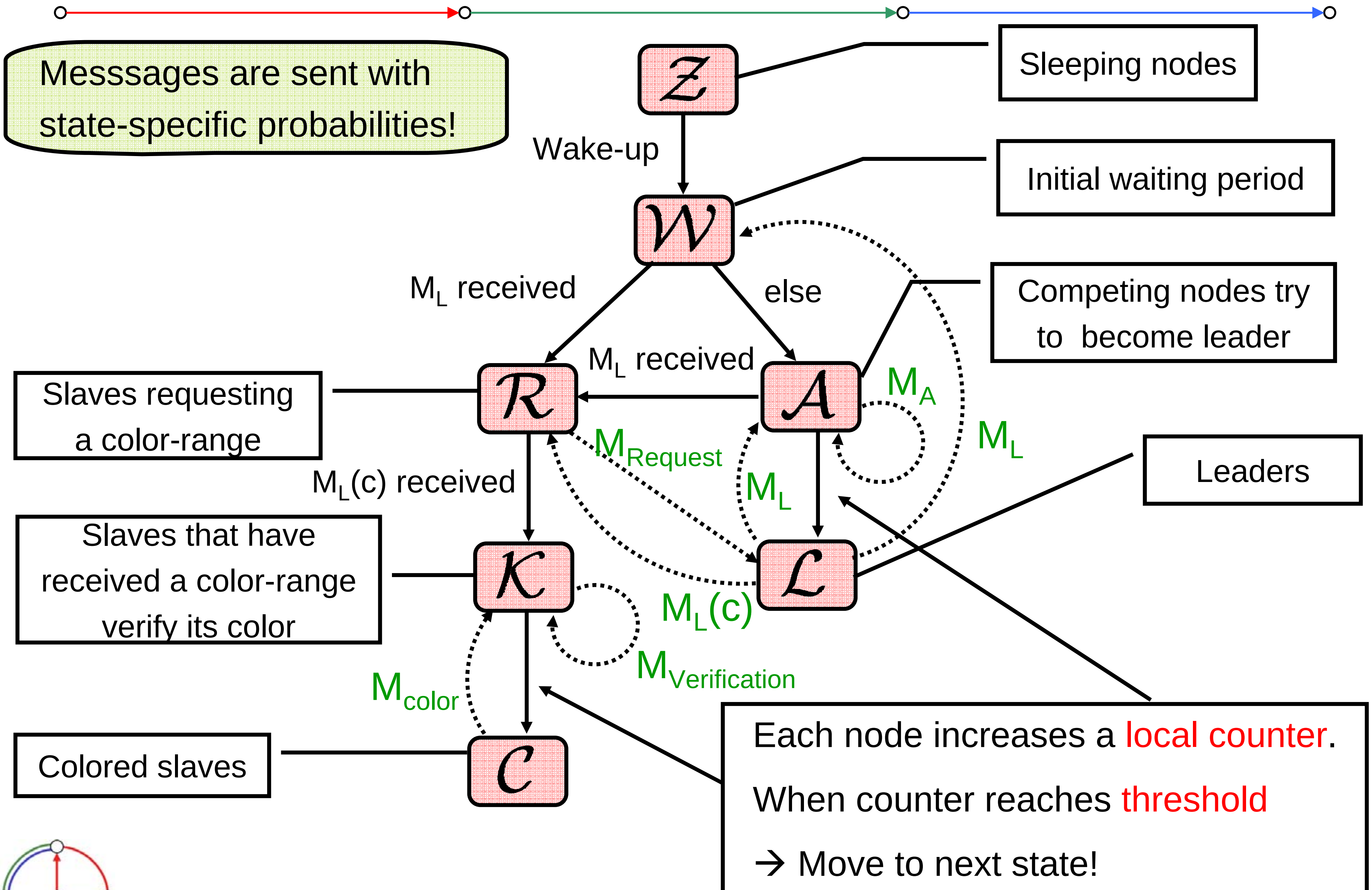
- Leaders assign initial coloring that is correct within the cluster
- Problem: Nodes in different clusters may be neighbors!



Interpret initial color as a color-range!

- In a final verification phase, nodes select final (conflict-free) color from color-range!

Algorithm Overview (a node's view)



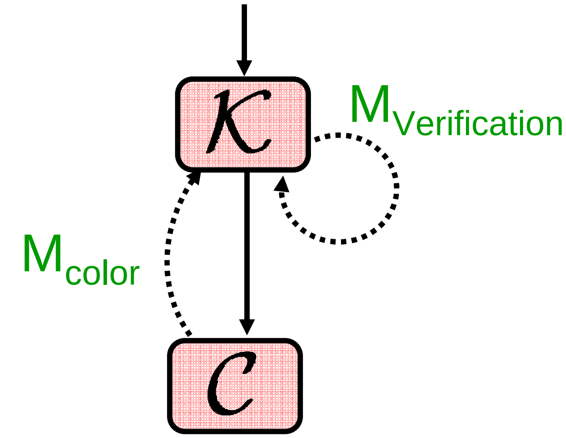
Algorithm Overview (Challenges)

- Problems:
  - → Everything happens concurrently!
  - → Nodes do not know in which state neighbors are (they do not even know whether there are any neighbors!)
  - → Messages may be lost due to collisions
  - → New nodes may join in at any time...
- Correctness!
  - → No two neighbors must choose the same color.
- No starvation!
  - → Every node must be able to choose a color within time $O(\Delta \log n)$ after its wake-up.

How to achieve both?

Avoid Starvation - Idea

- Use counters and appropriate thresholds
- Example: Consider state $\mathcal{K}$, node v verifies c



0) When receiving M_{color}(c) verify c+1

1) When entering state $\mathcal{K}$, set counter to 0.

2) In each time-slot, increase counter by 1.

3) When reaching $\sigma\Delta$log n, choose color and move to state $\mathcal{C}$

4) With probability p_K, transmit $M_{Verification}$(counter,c) and set counter to

$$counter := \max\{counter, \gamma\Delta \log n\} + 1$$

Cascading resets..?

5) When receiving $M_{Verification}$(counter*,c) from another node:

If counters are within $\gamma\Delta \log n$ of one another → Reset counter!

This method achieves both correctness and quick progress (in every region of the graph)!

Avoid Starvation - Idea

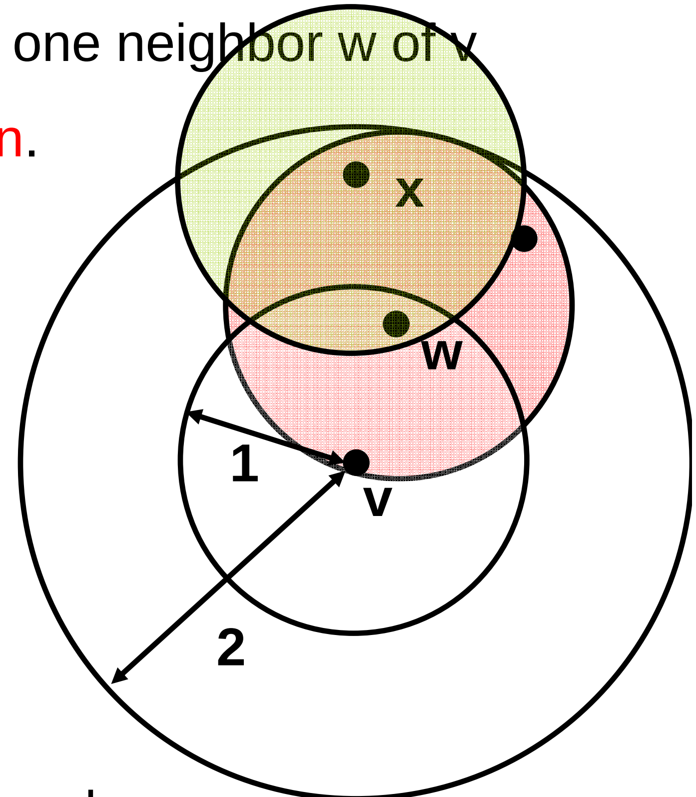
- Consider a node v entering state $\mathcal{K}$ at time t_v and verifying color c
- We show that by time $t_v+O(\Delta \log n)$, at least one neighbor w of v has transmitted (broadcast!) without collision.
- w has counter at least $\gamma\Delta \log n+1$
- All neighbors of w verifying c

  - either reset their counter

  - or have a counter that is at least $\gamma\Delta \log n$ away from w's counter.

→ w cannot be reset anymore by nodes in $\mathcal{K}$!

→ w may get M_{color} from a node $x \in \mathcal{C}$ that has chosen the color c earlier!



x covers a constant fraction of the disk of radius 2!

Avoid Starvation - Idea

Each taking time O(Δ log n)

- After a constant number of repetitions, the disk will be covered

  → node v either chooses c or receives M_{color} and verifies c+1

  → The argument repeats itself for c+1

- Because the set of leaders is sparse

  → v must verify only up to color c+μ, for $\mu \in O(1)$

W.h.p, every node spends only O(Δ log n) time-slots in state $\mathcal{K}$

In the proof, we similarly avoid starvation in all states!

- Specifically, we prove that: $T_{\mathcal{W}}, T_{\mathcal{A}}, T_{\mathcal{R}}, T_{\mathcal{K}} \in O(\Delta \log n)$

  Hence, $T = T_{\mathcal{W}} + T_{\mathcal{A}} + T_{\mathcal{R}} + T_{\mathcal{K}} \in O(\Delta \log n)$

Results

With high probability, the distributed coloring algorithm ...

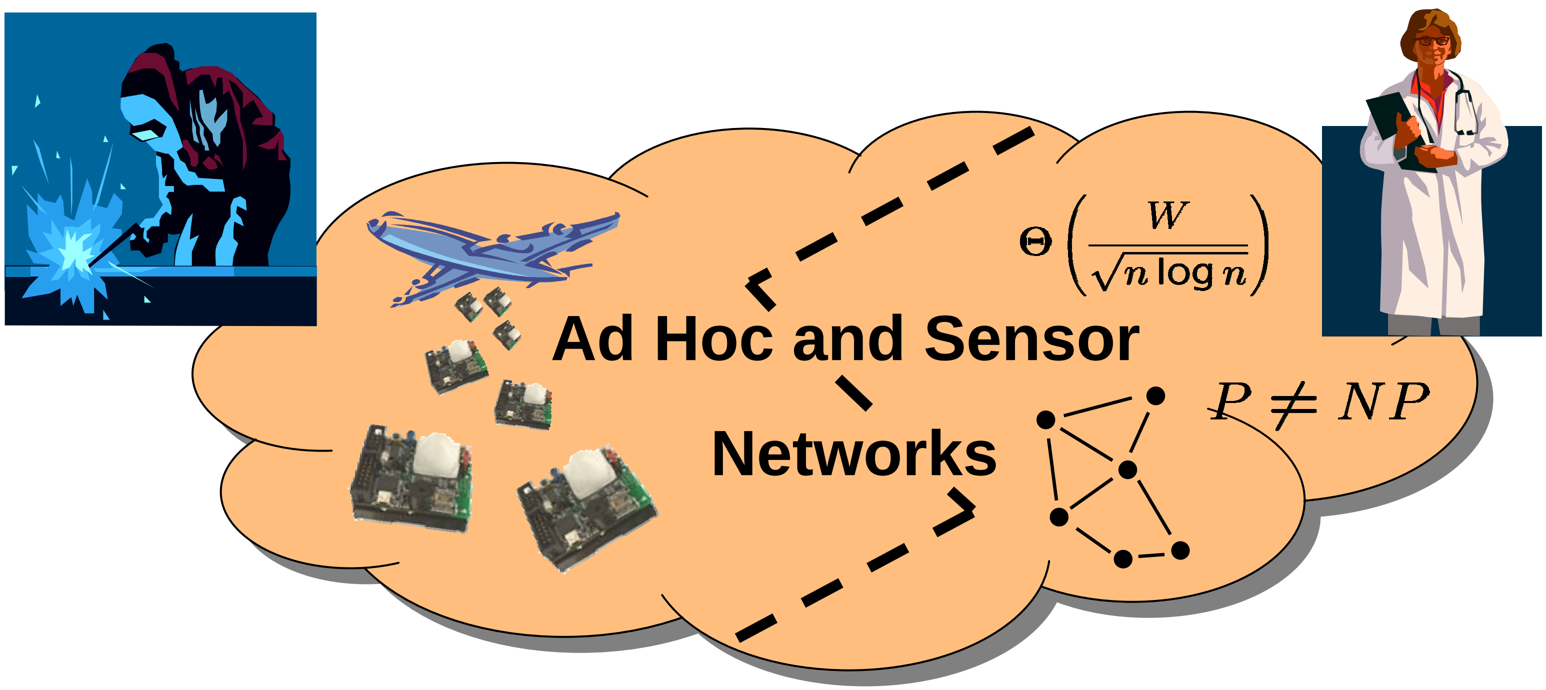
→... achieves a correct coloring using $O(\Delta)$ colors

→... every node irrevocably decides on a color within time $O(\Delta \log n)$ after its wake-up

→... the highest color depends only on the local maximum degree

Of Theory and Practice...

Practice **Theory**



There is often a big gap between theory and practice in the field of wireless ad hoc and sensor networks.

Conclusions / Open Problems

- O(Δ) coloring in harsh radio network model in time O(Δ log n) w.h.p.
  → Tight up to a factor of O(log n)
  → Color assignment according to local density

Future Directions:

- Close the remaining complexity gap
- Algorithm assumes knowledge of n and Δ
  → Remove this assumption
- 2-hop coloring ?
- Asynchronous wake-up: many open questions

Questions? Comments?

Questions?
Comments?

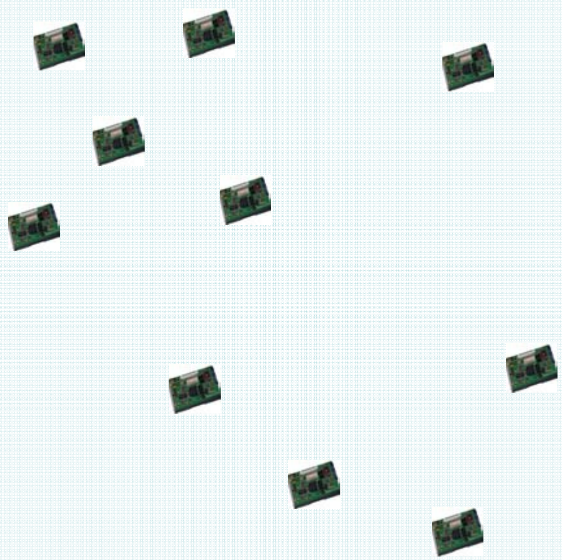
Distributed
Computing
Group

Thomas Moscibroda

Roger Wattenhofer

Of Theory and Practice...

Initial MAC layer



this talk → current work

Nodes know neighbors, etc.

Fair MAC layer
- Ad hoc networks

High-Throughput MAC layer
- Multimedia

Energy-Efficient MAC layer
- Long lifetime
- Sensor networks

There's more to deployment
- Time synchronization
- Topology control, etc.

Late arrivals

Mobility?

Failures?

time