MACbeth

The Three Witches of Media Access Theory

Roger Wattenhofer

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

What has been studied …most ardently?

What is really important?!?

- #1 MAC Layer (e.g. Coloring)
- #2 Topology and Power Control
- Interference and Signal-to-Noise-Ratio
- #3 Clustering (e.g. Dominating Sets)
- Deployment (Unstructured Radio Networks)

Link Layer

- New Routing Paradigms (e.g. Link Reversal)
- #5 Geo-Routing
- #4 Broadcast and Multicast
- Data Gathering

Network Layer

- Location Services and Positioning
- Time Synchronization

Services

- #1 Capacity and Information Theory
- Lower Bounds for Message Passing
- Selfish Agents, Economic Aspects, Security

Theory/Models

Media Access Control (MAC) Layer

- The MAC layer protocol controls the access to the shared physical transmission medium
  - In other words, which station is allowed to transmit at which time (on which frequency, etc.)

- MAC layer principles/techniques
  - Space and frequency multiplexing (always, if possible)
  - TDMA: Time division multiple access (GSM)
  - CSMA/CD: Carrier sense multiple access / Collision detection (Ethernet)
  - CSMA/CA: Carrier sense multiple access / Collision avoidance (802.11)
  - CDMA: Code division multiple access (UMTS)

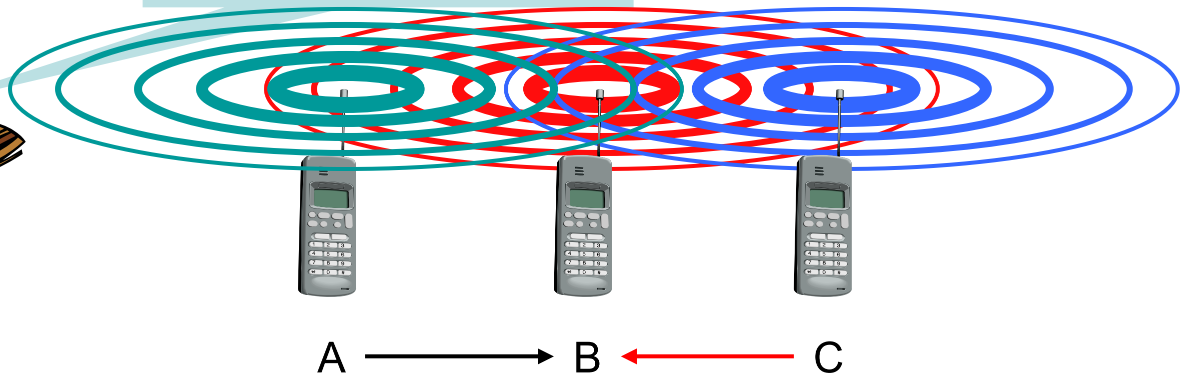
Why is the MAC layer so important?

- In a wireless multi-hop network, many design issues are central
  - Application
  - Hardware design
  - Physical layer (e.g. antenna)
  - Operating system
  - Sensor network: Sensors
  - … more topics not really related to algorithms/theory/fundamentals

- However, also really critical is the MAC Layer
  - In my opinion much more essential than, e.g. routing
  - Higher throughput
  - Saving energy (long sleeping cycles)

An Orthodox TDMA MAC algorithm

- Given a connectivity graph G, often a unit disk graph
  What?!?
- Interference? Two-hop neighbors! (“Hidden terminal problem”)
  Why?!



- Algorithm: G’ = G + two-hop links, min-color G’ How?
  – Frame length = number of colors, slot = color.

The Three Witches (Talk Outline)

- Introduction
  - Why MAC is important
  - Orthodox MAC
- Witch #1: The Chicken-and-Egg Problem
- Witch #2: Power Control is Essential
- Witch #3: Models, Models, Models!

Please mind, this is talk about theory/algorithms/fundamentals, not systems. Systems are more difficult, or at least different…

Witch #1: The Chicken-and-Egg Problem

- Excerpt from a typical paper:

Algorithm 2 $\mathrm{LP_{MDS}}$ approximation (Δ known)

$x_i := 0;$
for $\ell := k-1$ to 0 by -1 **do**
    $(* \, \tilde{\delta}(v_i) \leq (\Delta+1)^{(\ell+1)/k}, \; z_i := 0 \, *)$
    for $m := k-1$ to 0 by -1 **do**
        $(* \, a(v_i) \leq (\Delta+1)^{(m+1)/k} \, *)$
        send color_i to all neighbors;
        $\tilde{\delta}(v_i) := \left|\{j \in N_i \mid \mathrm{color}_j = \text{'white'}\}\right|;$
        if $\tilde{\delta}(v_i) \geq (\Delta+1)^{\ell/k}$ **then**
            $x_i := \max\left\{x_i, \frac{1}{(\Delta+1)^{m/k}}\right\}$
        fi;
        send x_i to all neighbors;
        if $\sum_{j \in N_i} x_j \geq 1$ **then** $\mathrm{color}_i :=$ 'gray' **fi**;
    od
    $(* \, z_i \leq 1/(\Delta+1)^{(\ell-1)/k} \, *)$
od

Coloring Algorithms Assume an Established MAC Layer...

6: **send** color$_i$ to all neighbors;

How do you know your neighbors?

How can you exchange data with them?

→ Collisions (Hidden-Terminal Problem)

Most papers assume that there is a MAC Layer in place!

This assumption may make sense in well-established, well-structured networks,...

...but it is certainly invalid during and shortly after the deployment of ad hoc and sensor networks, when there is not yet a MAC layer established

... Or a Global Clock

2: **for** $\ell := k - 1$ to 0 by -1 **do**

How do nodes know when to start the loop?

What if nodes join in afterwards?
→ Asynchronous wake-up!

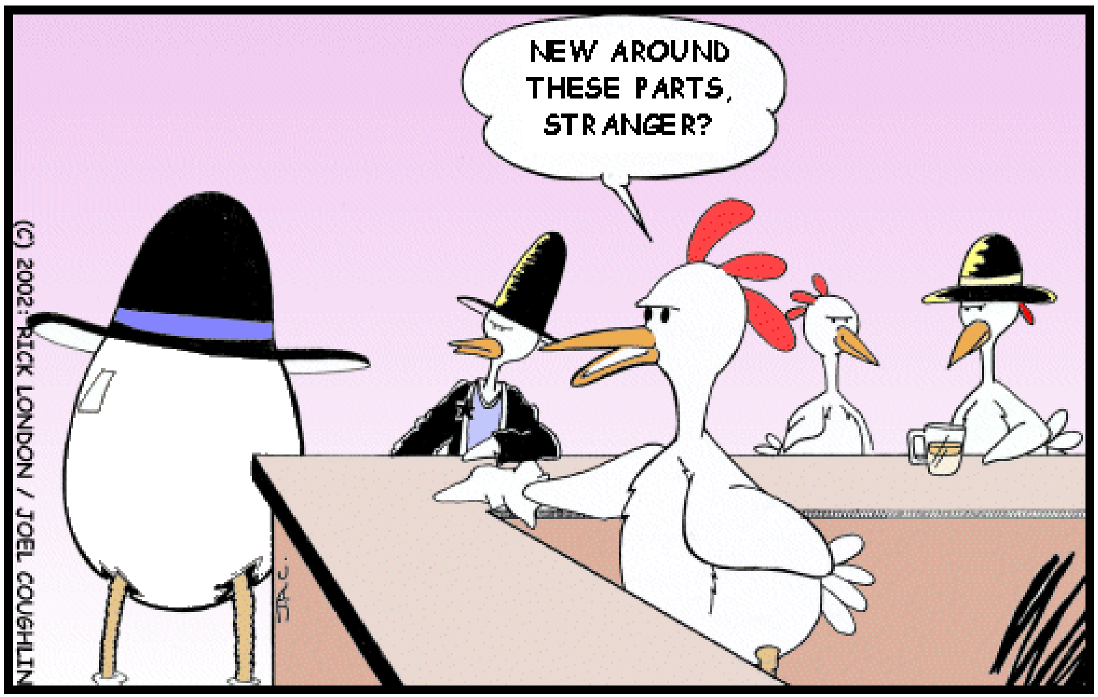
Paper assumes that there is a global clock and synchronous wake-up!

This assumption greatly facilitates the algorithm‘s analysis...

...but it is certainly invalid during and shortly after the deployment of ad hoc and sensor networks, when there is not yet a MAC layer established

We have a Chicken-And-Egg-Problem

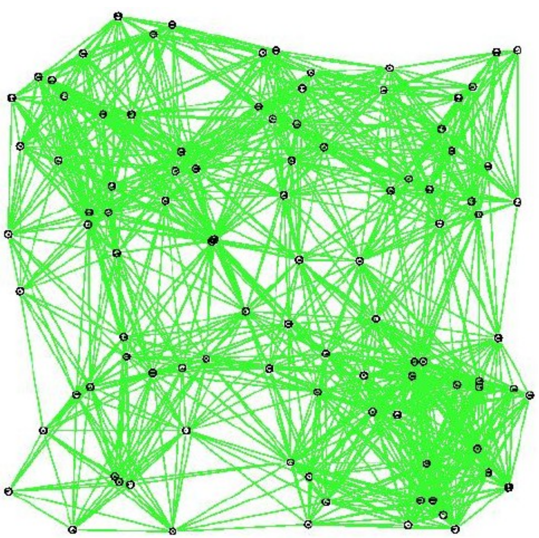
- TDMA MAC protocols can be reduced to two-hop coloring
- Coloring algorithms assume a working MAC layer



AND YET THE QUESTION REMAINED:
"WHO CAME FIRST?"

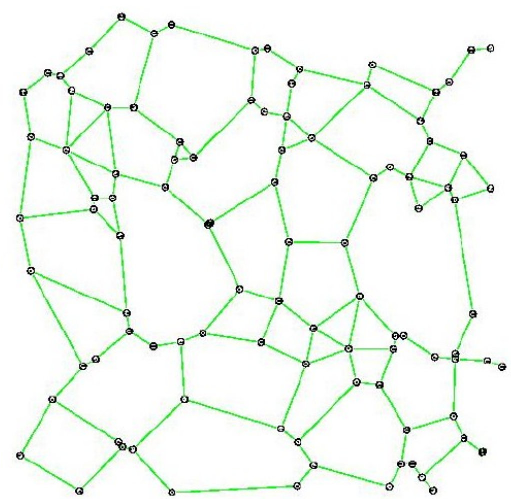
Deployment and Initialization

- Ad Hoc & Sensor Networks → no built-in infrastructure
- During and after the deployment → complete chaos
- Neighborhood is unknown
- There is no existing MAC-layer providing point-to-point connections!



Self-Organization

„Initialization“



Deployment and Initialization

- Initialization in current systems often slow (e.g. Bluetooth)

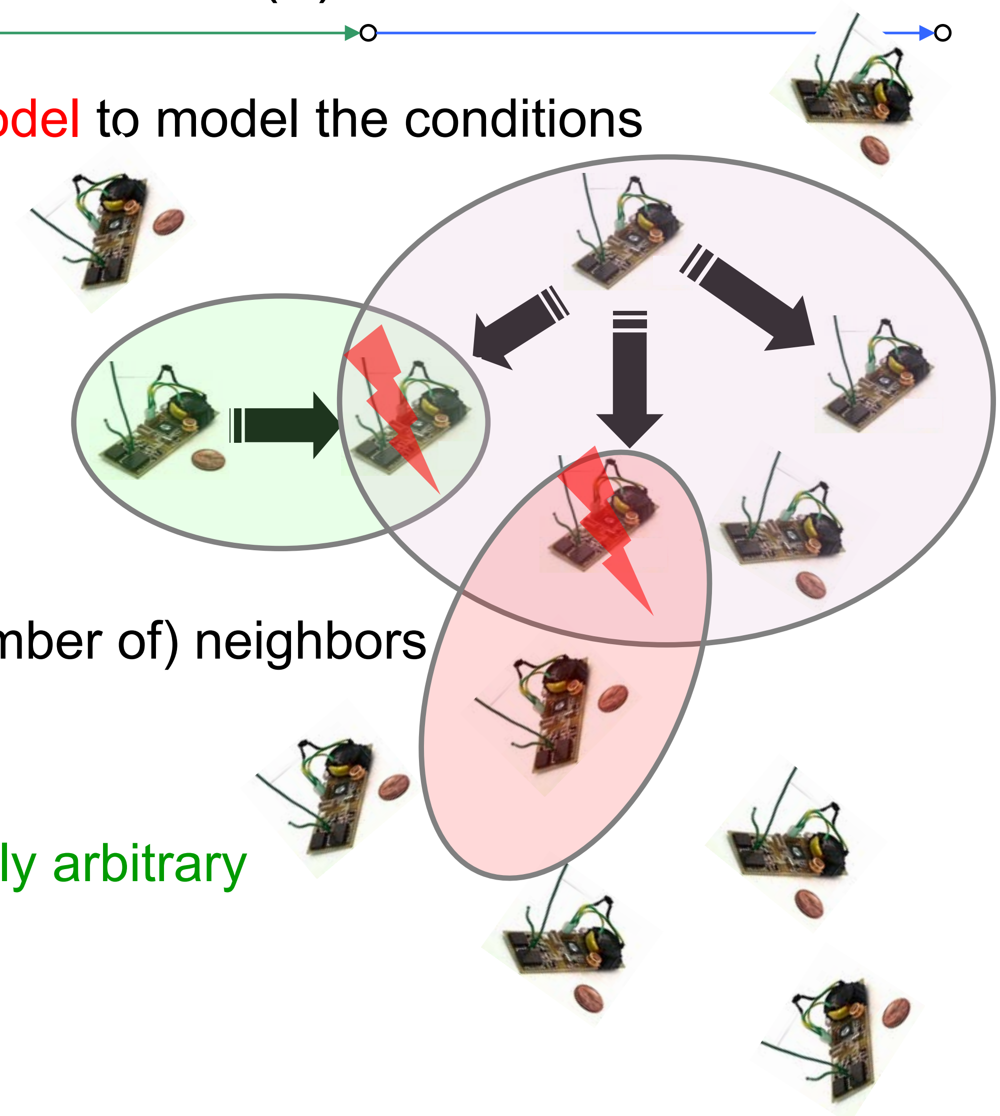
- Ultimate Goal: Come up with an efficient MAC-Layer quickly.
- Theory Goal: Design a *provably* fast and reliable initialization algorithm.

We have to consider the relevant technicalities!

- We need to define a model capturing the characteristics of the initialization phase.

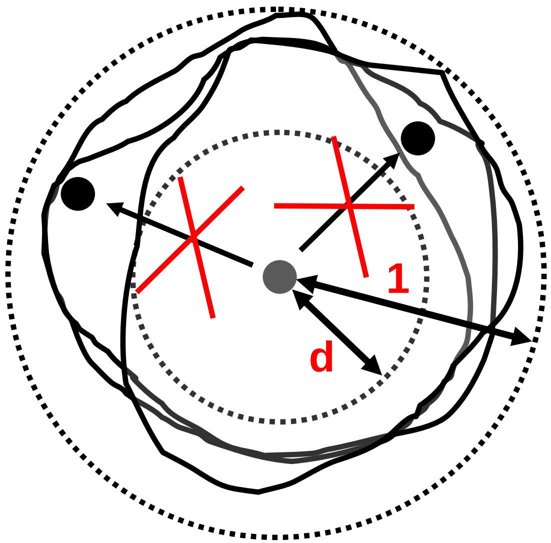
Unstructured Radio Network Model (1)

Adapt classic Radio Network Model to model the conditions immediately after deployment.



- Multi-Hop
  - Hidden-Terminal Problem
- No collision detection
  - Not even at the sender
- No knowledge about (the number of) neighbors
- Asynchronous Wake-Up
  - No global clock
- Node distribution is completely arbitrary
  - No uniform distribution

Unstructured Radio Network Model (2)



- Quasi Unit Disk Graph (QUDG) to model wireless multi-hop network
  - Two nodes can communicate if Euclidean distance is $\leq$ d
  - Two nodes cannot communicate if Euclidean distance is >1
  - In the range [d..1], it is unspecified whether a message arrives [Barrière, Fraigniaud, Narayanan, 2001]
- Upper bound N for number of nodes in network is known
  - This is necessary due to Ω(n / log n) lower bound [Jurdzinski, Stachowiak, 2002]

Q: Can we efficiently (and provably!) compute an *initial structure* in this harsh model?

Q: Can we efficiently (and provably!) compute an *MAC Layer* in this harsh model?

A: Yes, we can!

A: Hmm,...

Results

- Thomas Moscibroda, Roger Wattenhofer, SPAA 2005

With high probability, the distributed coloring algorithm ...

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

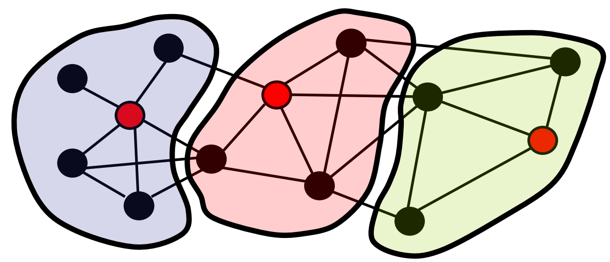
→... every node irrevocably decides on a color within time $O(\Delta \log n)$ after its wake-up

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

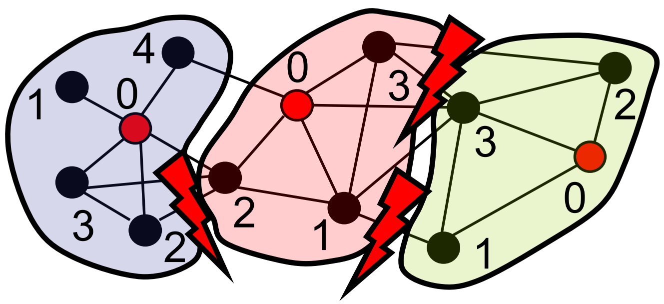
Algorithm Overview (system's view)

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

  → induces a clustering



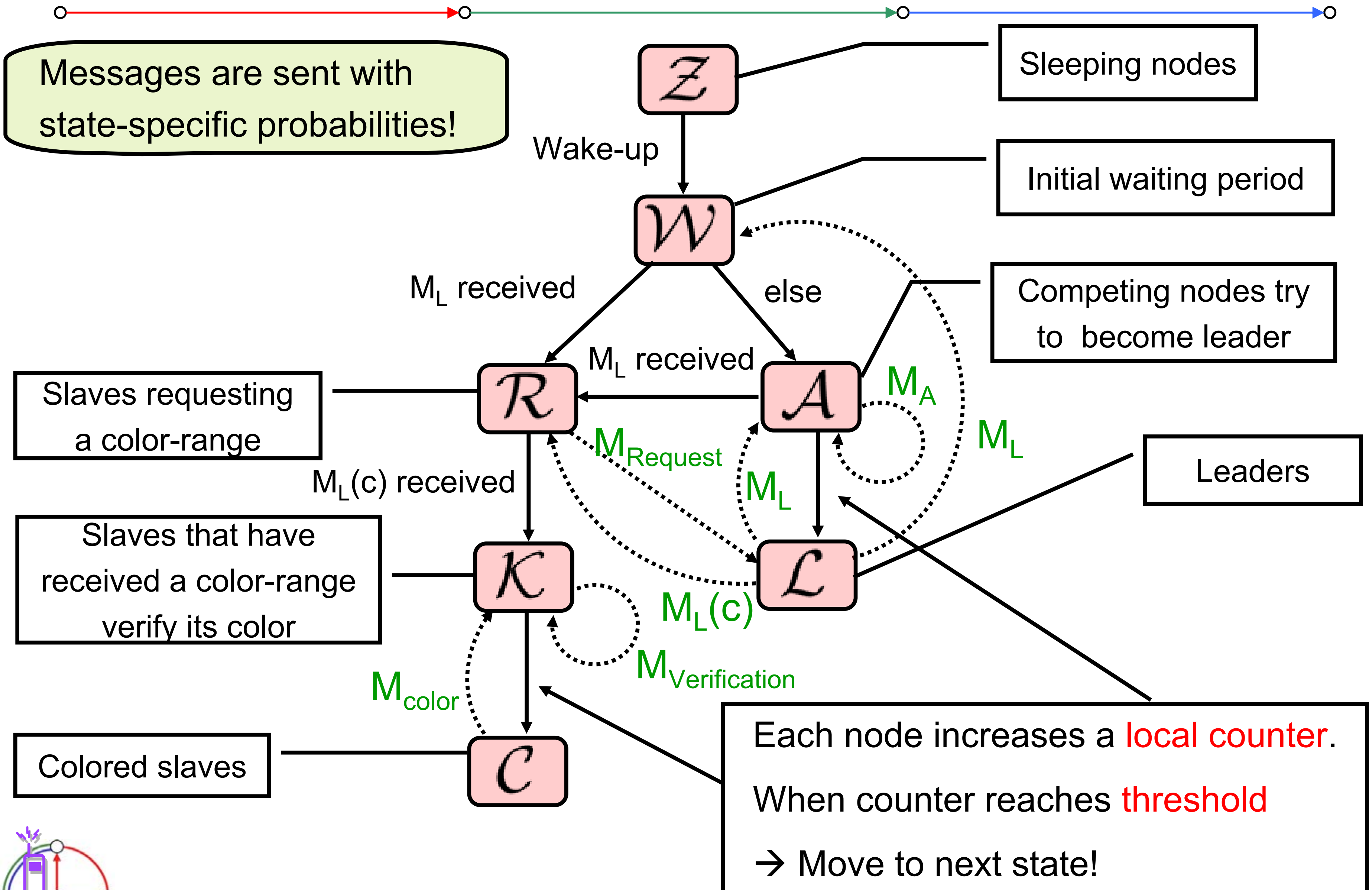
- Leaders assign initial coloring that is correct within the cluster
- Problem: Nodes in different clusters may be neighbors!



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)



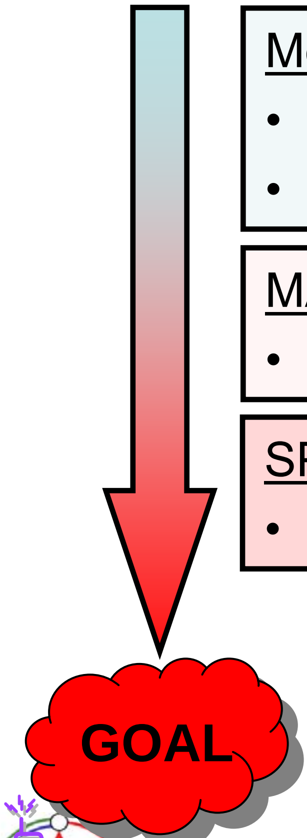
Algorithm Overview (Challenges)

- 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?

Conclusions

- Initialization of ad hoc and sensor network of great importance!
- Relevant technicalities must be considered!



MobiCom 2004 (Kuhn, Moscibroda, Wattenhofer)
- A model capturing the characteristics of the initialization phase
- A fast algorithm for computing a good dominating set from scratch

MASS 2004 (Moscibroda, Wattenhofer):
- A fast algorithm for computing more sophisticated structures (MIS)

SPAA 2005 (Moscibroda, Wattenhofer):
- A fast algorithm for computing a coloring

⋮

A fast algorithm for establishing a MAC Layer from scratch!

The Deployment Problem: Future Work

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

Algorithm Classes

Global Algorithm

Distributed Algorithm

- For some problems we don't even understand the non-distributed case
- "Reiceive msg X → Transmit msg Y"
- Every algo can be made distributed

Local

Localized

Unstructured

Local

+ Node can only communicate with neighbors k times.
+ Strict time bounds
– Often synchronous

Localized

+ Often simple
– Nodes can wait for neighbor actions
– Often linear chain of causality

Unstructured

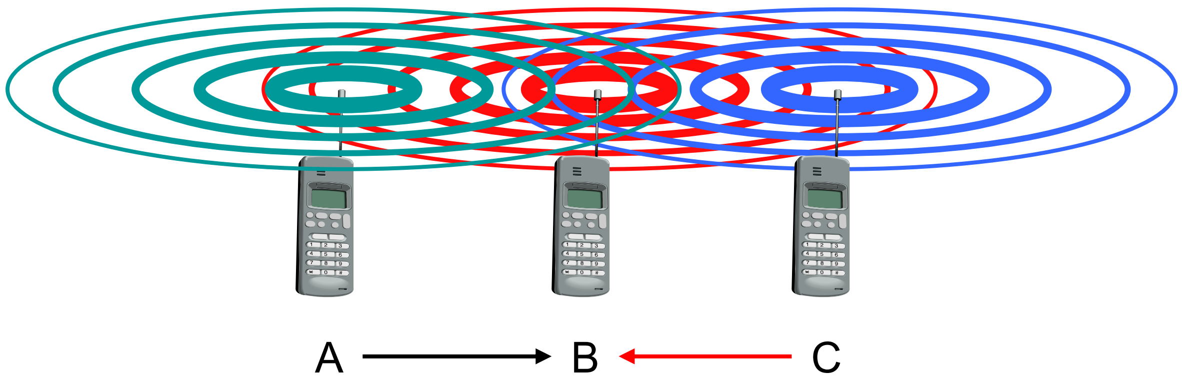
+ Implement MAC layer yourself; you control everything
– Often complicated
– Argumentation overhead

The Three Witches (Talk Outline)

- Introduction
  - Why MAC is important
  - Orthodox MAC
- Witch #1: The Chicken-and-Egg Problem
- Witch #2: Power Control is Essential
- Witch #3: Models, Models, Models!

Witch #2: Power Control is Essential

- Modeling interference in a typical algorithms paper:

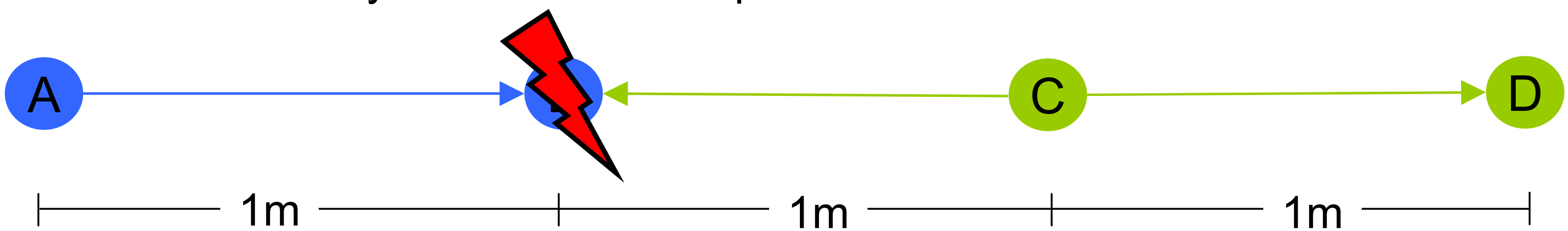


- The model is a simplification, sure, but is the hidden terminal problem really a problem?!?

The Hidden-Terminal Problem

Consider the following scenario:

- A wants to sent to B, C wants to send to D
- How many time slots are required?



Can A and C send simultaneously...?

No, they cannot!

This is the *Hidden-Terminal Problem*!

Interference causes a collision at B!

But is this really true...?

The Hidden-Terminal Problem

A wants to sent to B, C wants to send to D

- Let us look at the signal-to-noise-plus-interference (SINR) ratio!
- Message arrives if SINR is larger than β at receiver

$$\frac{\frac{P_u}{d(u,v)^\alpha}}{N + \sum_{w \in V \setminus \{u\}} \frac{P_w}{d(w,v)^\alpha}} \geq \beta$$

Power level of node u

Path-loss exponent

Noise

Minimum signal-to-interference ratio

Distance between two nodes

The Hidden-Terminal Problem

A wants to sent to B, C wants to send to D

- Let α=3, β=4, and N=1 (these are realistic values in sensor networks)
- Set the transmission powers as follows P_C=15 and P_A=70
- The SINR at D is:

$$\frac{15/1^3}{1+70/3^3} \approx 4.17 \geq \beta$$

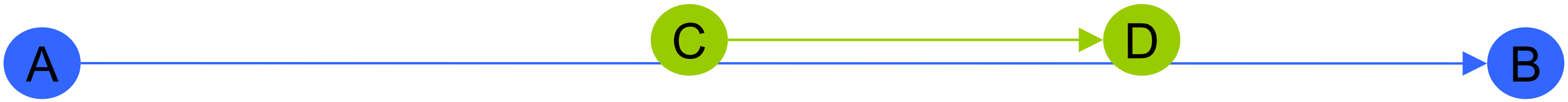
- The SINR at B is:

$$\frac{70/1^3}{1+15/1^3} \approx 4.37 \geq \beta$$

Simultaneous transmission *is* possible !

Let's make it tougher!

A wants to sent to B, C wants to send to D



Can A and C send simultaneously...?

But is this really true...?

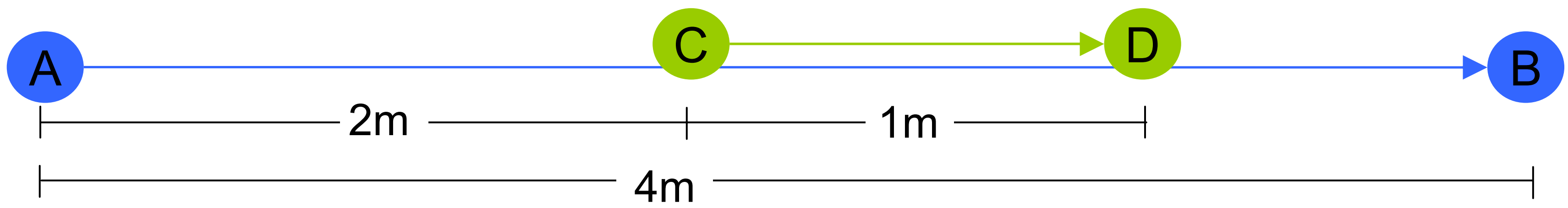
No, they cannot!

Reasons

- D is in sending range of A → collision at D
- B hears either C or a collision, but not A!
- Common Sense....

Let's make it tougher!

A wants to sent to B, C wants to send to D



- Let α=4, β=2, and N=1
- Set the transmission powers as follows P_C=100 and P_A=3900
- The SINR at D is:

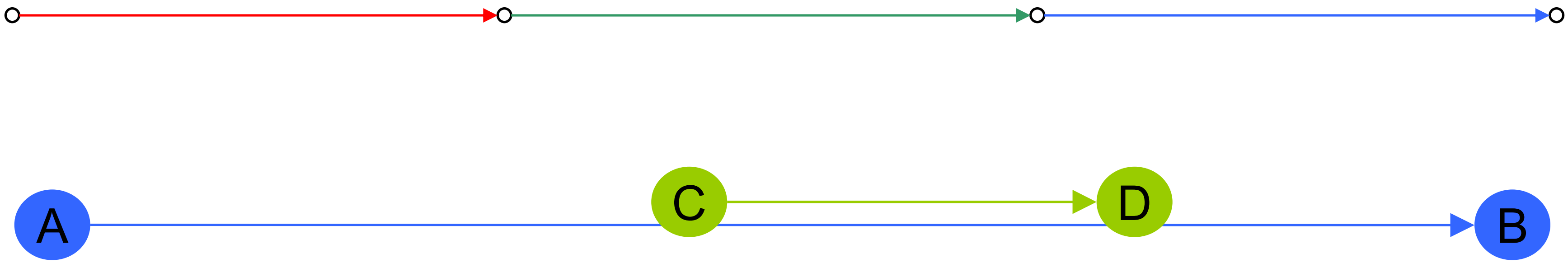
$$\frac{100/1^4}{1+3900/3^4} \geq \beta$$

- The SINR at B is:

$$\frac{3900/4^4}{1+100/2^4} \geq \beta$$

Again: Simultaneous transmission *is* possible !

Theory vs. Reality!



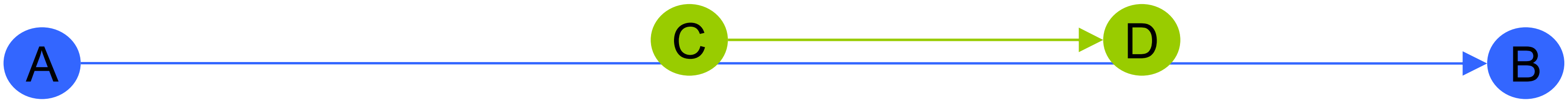
Graph Theoretical Models:

There exists no graph-theoretic model that can capture the above !

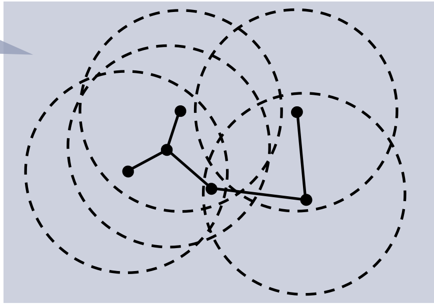
- Unit Disk Graph → No!
  (C cannot send to D in this model!)
- General Graph → No!
  (because success depends on A's power!)
- Radio Network Models → No!
  (Collision garbles messages!)
- Etc...

Modeling networks as graphs appears to be inherently wrong!!!

Theory vs. Reality!



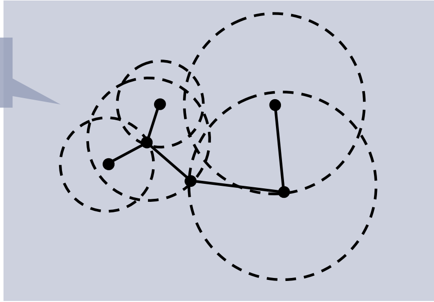
Constant power level



Power Assignment Policies:

- All nodes have uniform power → No!
  - Node B will receive the transmission of node C
  - Impossible even in SINR model!

Proportional to d^{α}



- Powers are according to $P \sim O(d^{\alpha})$ → No!
  - This linear power assignment often assumed in theory (minimum energy broadcast, topology control, etc...)
  - Node D will receive the transmission of node A

All typically studied power assignment schemes are bad!

Theory vs. Reality!

- We have seen....
  1) Graph models are inherently flawed!
  2) Standard power assignment assumptions are suboptimal!

- The question is....

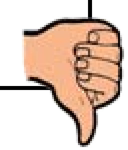
How far from reality are graph models...?

Some necessary, technical simplifications.

Fundamental aspects are captured and results remain essentially valid

Some necessary, technical simplifications.

Obtained results are fundamentally different from reality!



Theory vs. Reality!

- We have seen....
  1) Graph models are inherently flawed!
  2) Standard power assignment assumptions are suboptimal!

- The question is....

1) Uniform Power Levels...
2) Power according to $P \approx \Theta(d^{\alpha})$

How sub-optimal are common power assignment schemes...?

Achieved throughput is acceptably high

Simple power assignment schemes can be employed

The resulting throughput is way below the theoretical limits

More subtle power assignment schemes are required!

A Simple Scheduling Problem

1. How far from reality are graph models...?
2. How sub-optimal are common power assignment schemes...?

Consider the following simple scheduling task Ψ:

Ψ: **Every node can send *one* message successfully?**

Nodes can choose receivers optimally! (e.g. nearest neighbor)

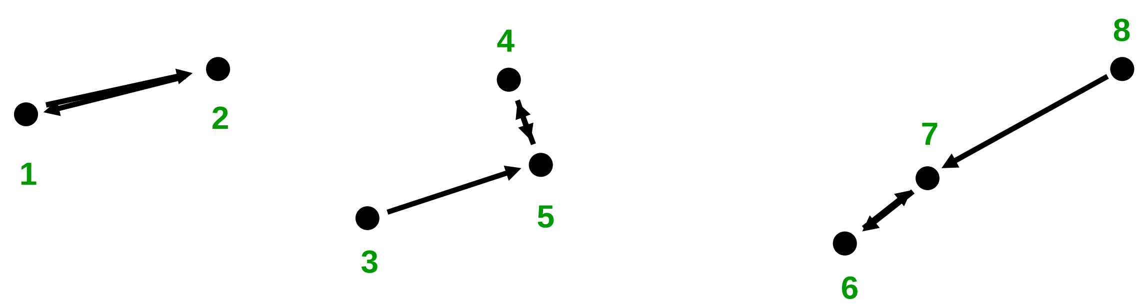
How many time-slots are required to schedule this task?

„The Scheduling Complexity in Wireless Networks“

A Simple Scheduling Problem - Example

1. How far from reality are graph models...?
2. How sub-optimal are common power assignment schemes...?

An example:



Time-Slot	**Senders:**
t_1:	v_1, v_4, v_7
t_2:	v_1, v_3, v_6
t_3:	v_5, v_8

This scheme uses 3 time slots!

→ **Scheduling complexity of Ψ is 3 in this example.**

A Simple Scheduling Problem

1. How far from reality are graph models...?
2. How sub-optimal are common power assignment schemes...?

- This is possibly the simplest possible scheduling problem!

> Define: Scheduling Complexity $S(\Psi)$ of Ψ
>
> The number of time-slots required until every node can transmit at least once!

Clearly, $S(\Psi) \leq n$

→ Problem describes a fundamental property of wireless networks.

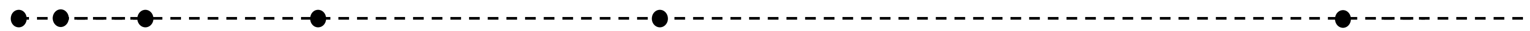
→ Because the problem is so simple...

1... standard MAC protocols are expected to perform reasonably well.

2... graph-based models are expected to be reasonably close to reality.

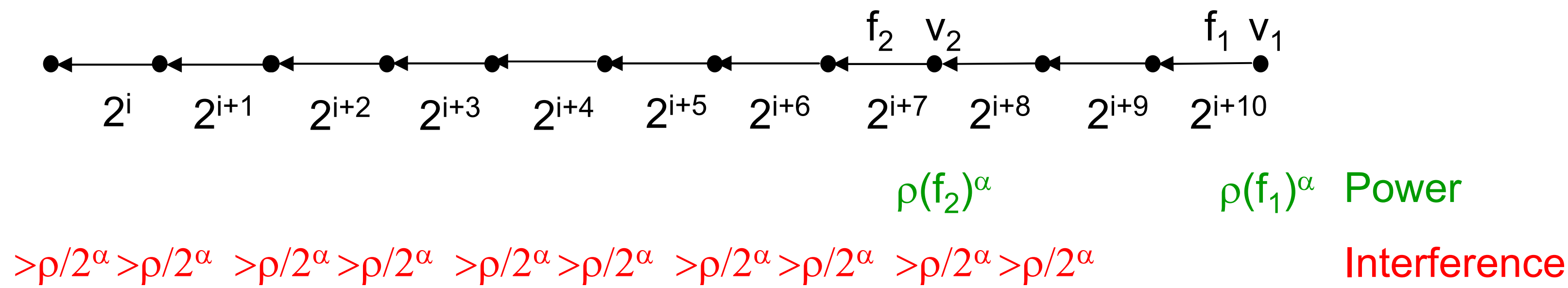
Lower Bound for $P \sim O(d^{\alpha})$ Power Assignment

- Consider again the exponential chain:



Lower Bound for $P \sim O(d^{\alpha})$ Power Assignment

- Consider again the exponential chain:



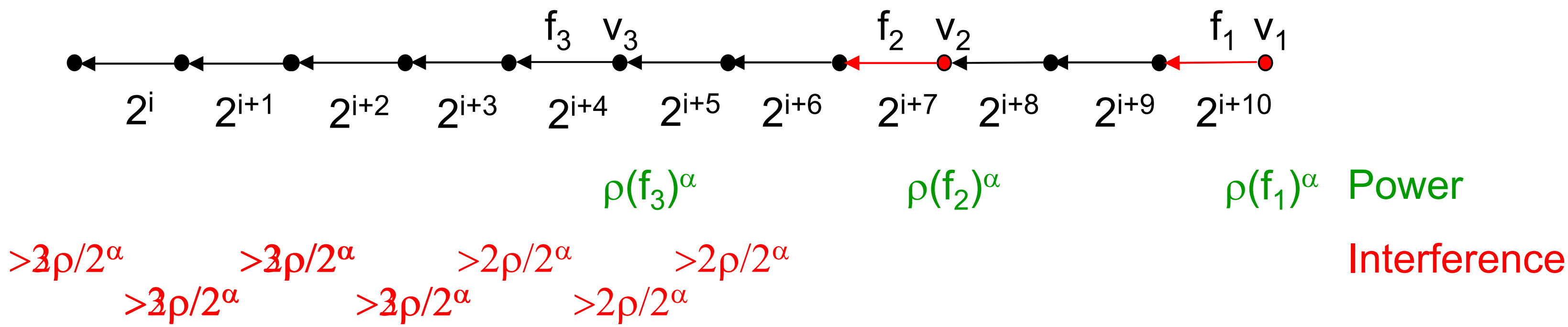
- How many links can we schedule simultaneously?
- Let us start with the first node v_1...
  → its power is $P_1 \geq \rho 2^{\alpha(i+10)}$ for some constant ρ

Why…???

- This creates interference of at least $\rho/2^{\alpha}$ at every other node!
- The second node v_2 also sends with power $P_2 = \rho 2^{\alpha(i+7)}$
- Again, this creates an additional interference of at least $\rho/2^{\alpha}$ at every other node!

Lower Bound for $P \sim O(d^\alpha)$ Power Assignment

- Consider again the exponential chain:



- How many links can we schedule simultaneously?
- Let us start with the first node v_1...
  → its power is $P_1 \geq \rho 2^{\alpha(i+10)}$ for some constant ρ
- This creates interference of at least $\rho/2^\alpha$ at every other node!
- The second node v_2 also sends with power $P_2 \geq \rho 2^{\alpha(i+7)}$
- Again, this creates an additional interference of at least $\rho/2^\alpha$ at every other node!

Why…???

And so on…

Lower Bound for $P \sim O(d^\alpha)$ Power Assignment

- Assume we can schedule *R* nodes in parallel.
- The left-most receiver x_r faces an interference of $R \cdot \rho/2^\alpha$

  → yet, x_r receives the message, say from x_s.
- How large can *R* be?
- The SINR at x_r must be at least β, and hence

$$\frac{\frac{\rho \cdot d(x_s, x_r)^\alpha}{d(x_s, x_r)^\alpha}}{N + R \cdot \frac{\rho}{2^\alpha}} \geq \frac{\rho 2^\alpha}{2^\alpha N + \rho R} \geq \beta$$

- From this, it follows that *R* is at most $2^\alpha/\beta$, and therefore...

  ... at least $n \cdot \min\{1, \beta/2^\alpha\}$ time slots are required for all links!

Any $P \sim O(d^\alpha)$ power assignment algorithm has scheduling complexity: **S(Ψ)∈ Ω(n)**

Lower Bounds and Lessons Learned…

- The trivial algorithm (scheduling each node individually) requires n time slots.

  $S(\Psi) \in O(n)$

- Any algorithm with $P \sim O(d^\alpha)$ power assignment requires Ω(n) time slots.

  $S(\Psi) \in \Omega(n)$

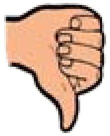
- Any algorithm with uniform power assignment requires Ω(n) time slots.

  $S(\Psi) \in \Omega(n)$

Hidden constants Are very small!

Observations:

- Theoretical performance of current MAC layer protocols almost as bad as scheduling every single node individually!
- Current MAC layer protocols have a severe scaling problem!
- Theoretically efficient MAC protocols must use non-trivial power levels!



Can we do better…?

- Can we break the $\Omega(n)$ barrier...?
- Observation: Scheduling a set of links of roughly the same length is easy...

  $S(\Psi) \in O(\#\text{of Length-classes})$

  → Partition the set of links in length-classes

  → Schedule each length-class independently one after the other...
- The problem is...

  e.g. exponential node-chain...

  → there may be many (up to n) different length-classes

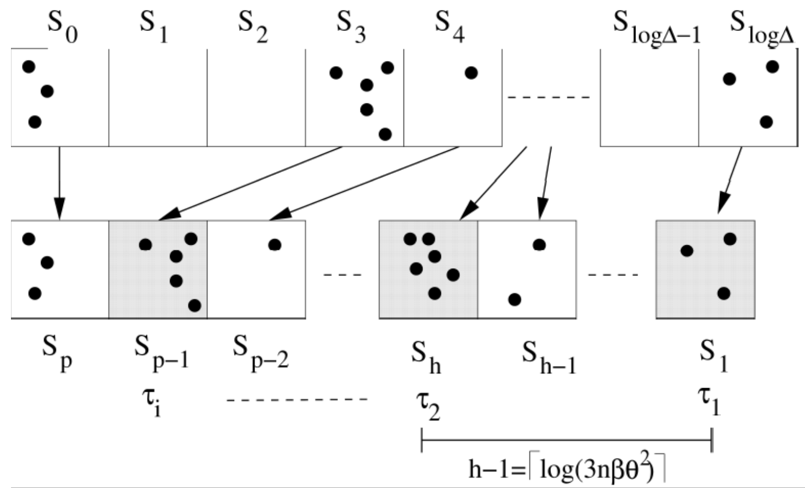
  → We must schedule links of different lengths simultaneously!

  e.g. uniform and $\sim d^{\alpha}$ examples before
- How can we assign powers to nodes?

  → Making the transmission power dependent on the length of link is bad!
- We must make the power assigned to simultaneous links dependent on their relative position of the length class!

Ooops, now it gets complicated...!

Can we do better…?



- A node v in length-class λ and a link of length d transmit roughly with a power of

$$P(v) \approx \beta^{\lambda} \cdot d^{\alpha}$$

Intuitively, nodes with small links must *overpower* their receivers!

- Unfortunately, it still does not work yet....
- ...we also need to carefully select the transmitting nodes!

Ooops, now it gets complicated...!

Can we do better…?

- Yes, we can... ... but it is somewhat complicated!

Compare to $\Omega(n)$

- Our results are [Moscibroda, Wattenhofer, INFOCOM 06]:

Problem Ψ can be scheduled in time: **$S(\Psi) \in O(\log^2 n)$**

What about scheduling more complex topologies than Ψ?

In any network, a strongly-connected topology can be scheduled in time: **$S(\text{Connected}) \in O(\log^3 n)$**

What about arbitrary set of requests?

Any topology can be scheduled in time:
$S(\text{Arbitrary}) \in O(I_{in} \cdot \log^2 n)$

The Three Witches (Talk Outline)

- Introduction
  - Why MAC is important
  - Orthodox MAC
- Witch #1: The Chicken-and-Egg Problem
- Witch #2: Power Control is Essential
- Witch #3: Models, Models, Models!

Let´s Talk about Models!

- Why models for sensor networks?
  - Allows precise evaluation and comparison of algorithms
  - Analysis of correctness and efficiency (proofs)

- Goal of model designer?
  - Simplifications and abstractions, … but not too simple.

- There are models for connectivity, interference, algorithm type, node distribution, energy consumption, etc.
  - Survey by Stefan Schmid, Roger Wattenhofer, WPDRTS 2006
  - This talk: A few examples for connectivity models

Example: Comparison of Two Algorithms for Dominating Set

Algorithm 1

- Algorithm co... S
- k^2+O(1)... sions/node
- O(Δ... approximation
- ...plex!
- ...nance OK

General Graph!
No Position Information!

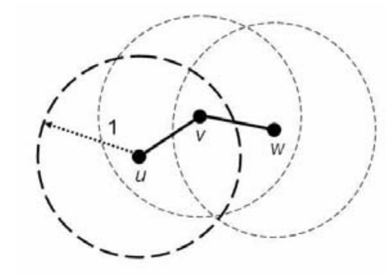
Algorithm 2

- Algorith... s DS
- 1 tr... n/node
- O... ximation
- ...formance great!

Unit Disk Graph Only!
Requires GPS Device!

The model determines the distributed complexity of a problem

Connectivity Models



General Graph

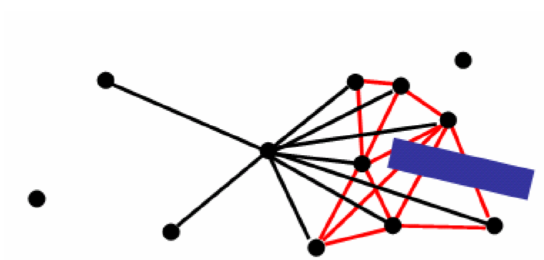
UDG

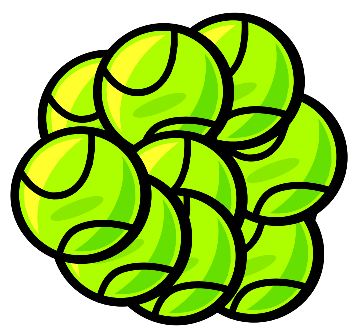
too pessimistic **too optimistic**

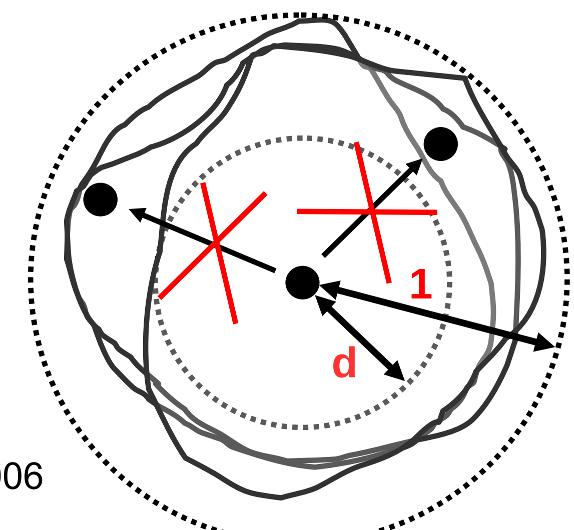
Bounded Independence

Unit Ball Graph

Quasi UDG

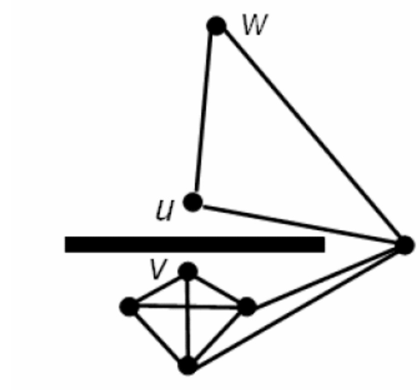






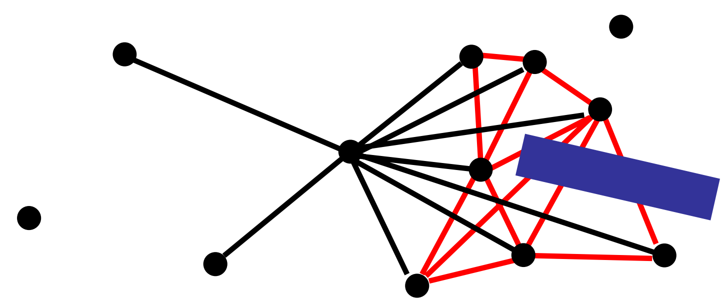
Connectivity: Bounded Independence Graph (BIG)

- How realistic is QUDG?
  - u and v can be close but not adjacent
  - model requires very small d in obstructed environments (walls)



- However: in practice, neighbors are often also neighboring

- Solution: BIG Model
  - Bounded independence graph
  - Size of any independent set grows polynomially with hop distance *r*
  - e.g. $O(r^2)$ or $O(r^3)$



Connectivity: Unit Ball Graph (UBG)

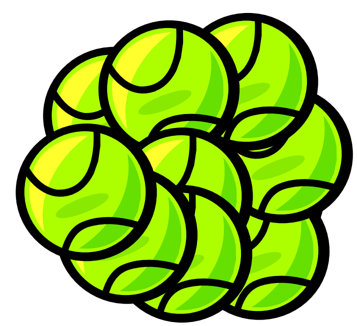
- $\exists$ metric (V,d) describing distances between nodes $u,v \in V$

such that:

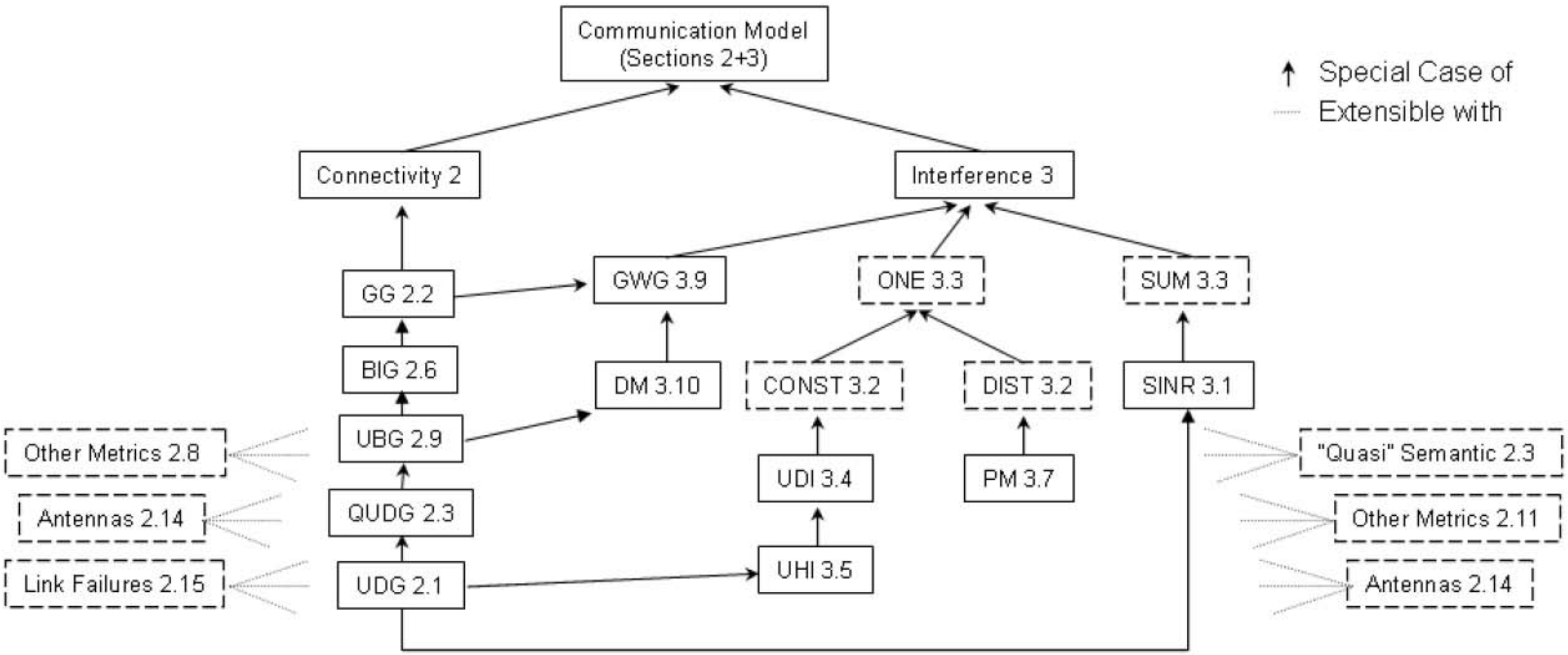
$$\boxed{\begin{array}{l} d(u,v) \leq 1 : (u,v) \in E \\ d(u,v) > 1 : (u,v) \notin E \end{array}}$$

- Assume that doubling dimension of metric is constant
  - Doubling dimension: log(#balls of radius r/2 to cover ball of radius r)

UBG based on underlying doubling metric.

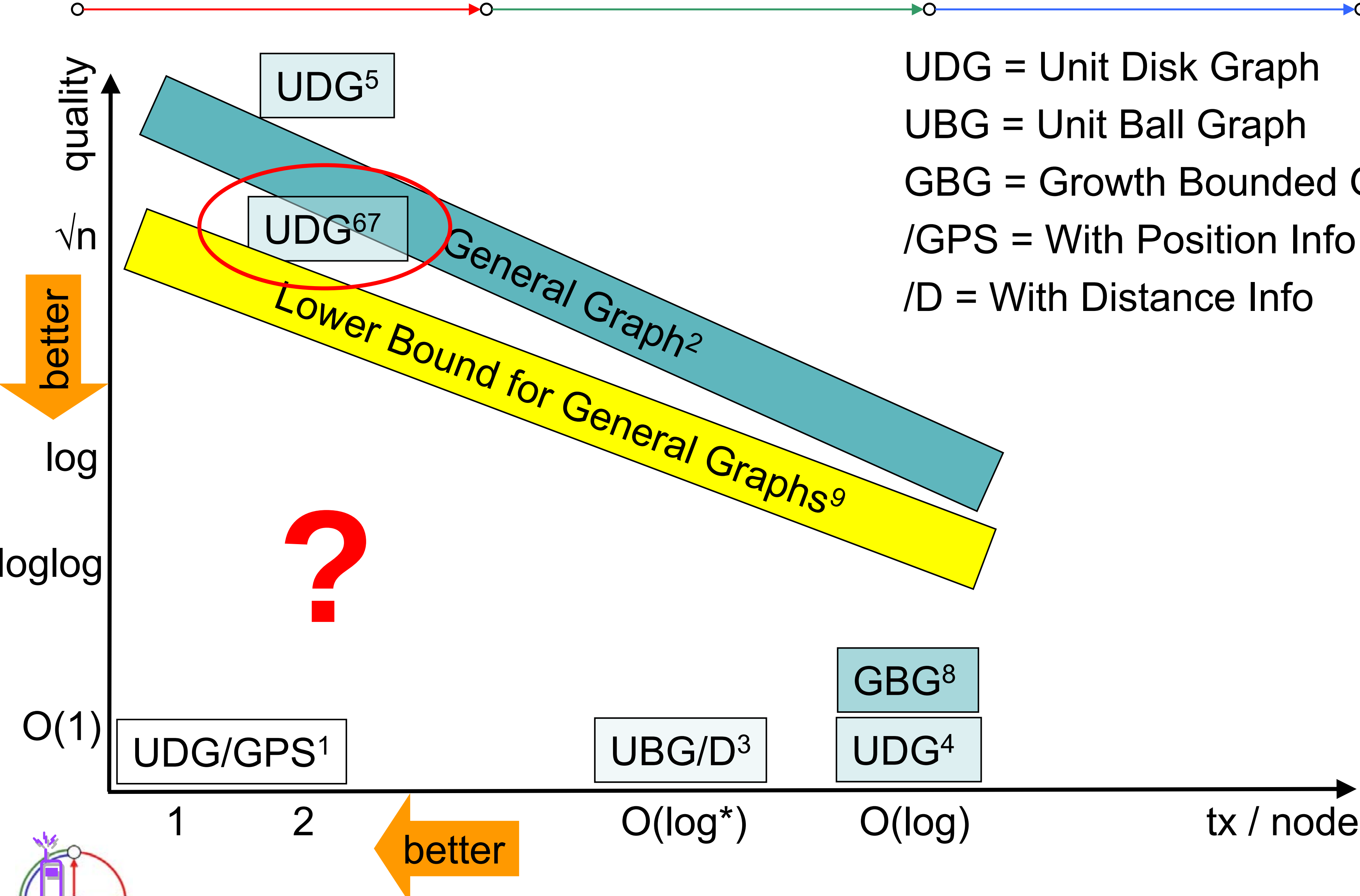


Models can be put in relation



- Try to proof correctness in an as “high” as possible model
- For efficiency, a more optimistic (“lower”) model might be fine

The model determines the complexity



UDG = Unit Disk Graph

UBG = Unit Ball Graph

GBG = Growth Bounded G.

/GPS = With Position Info

/D = With Distance Info

References

1. Folk theorem, e.g. Kuhn, Wattenhofer, Zhang, Zollinger, PODC 2003
2. Kuhn, Wattenhofer, PODC 2003
   - Improved: Kuhn, Moscibroda, Wattenhofer, SODA 2006
   - CDS by Dubhashi et al, SODA 2003
3. Kuhn, Moscibroda, Wattenhofer, PODC 2005
4. Alzoubi, Wan, Frieder, MobiHoc 2002
5. Wu and Li, DIALM 1999
6. Gao, Guibas, Hershberger, Zhang, Zhu, SCG 2001
7. Wattenhofer, MedHocNet 2005 talk, Improving on Wu and Li
8. Kuhn, Moscibroda, Nieberg, Wattenhofer, DISC 2005
9. Kuhn, Moscibroda, Wattenhofer, PODC 2004

My Own Private View on Networking Research

Class	Analysis	Communi cation model	Node distribution	Other drawbacks	Popu larity
Imple-mentation	Testbed	Reality	Reality(?)	“Too specific”	5%
Heuristic	Simulation	UDG to SINR	Random, and more	Many…! (no benchmarks)	80%
Scaling law	Theorem/ proof	SINR, and more	Random	Existential (no protocols)	10%
Algorithm	Theorem/ proof	UDG, and more	Any (worst-case)	Worst-case unusual	5%

Conclusions

- MAC Layer is important
  - Not much (theoretical) work done
  - There are issues
    - chicken-egg
    - power control
    - models
- It seems that the algorithms/foundations community is striving for new, more realistic models
  - I showed parts of the connectivity hierarchy
  - But there is much more, everything in flux
- Thanks to Thomas Moscibroda, Fabian Kuhn, Stefan Schmid, and more of my students for their work.


Thank You!
Questions?
Remarks?
Roger Wattenhofer
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

BACKUP

Lower Bound for $P \sim O(d^\alpha)$ Power Assignment

- Assume we can schedule *R* nodes in parallel.
- The left-most receiver x_r faces an interference of $R \cdot \rho/2^\alpha$

  → yet, x_r receives the message, say from x_s.
- How large can *R* be?
- The SINR at x_r must be at least β, and hence

$$\frac{\frac{\rho \cdot d(x_s,x_r)^\alpha}{d(x_s,x_r)^\alpha}}{N + R \cdot \frac{\rho}{2^\alpha}} \geq \frac{\rho 2^\alpha}{2^\alpha N + \rho R} \geq \beta$$

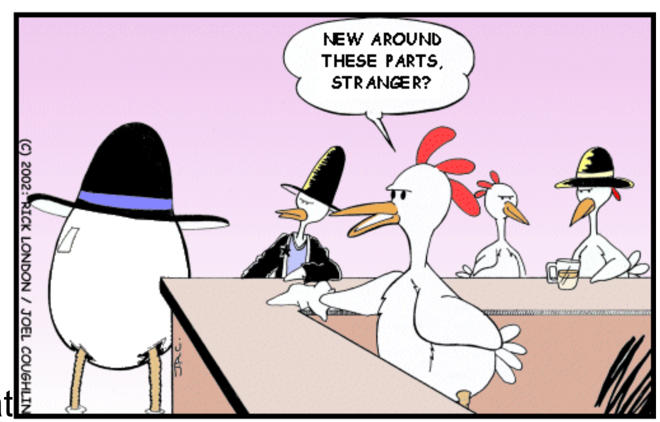
- From this, it follows that *R* is at most $2^\alpha/\beta$, and therefore....

  .... at least $n \cdot \min\{1, \beta/2^\alpha\}$ time slots are required for all links!

Any $P \sim O(d^\alpha)$ power assignment algorithm has scheduling complexity: **$S(\Psi) \in \Omega(n)$**

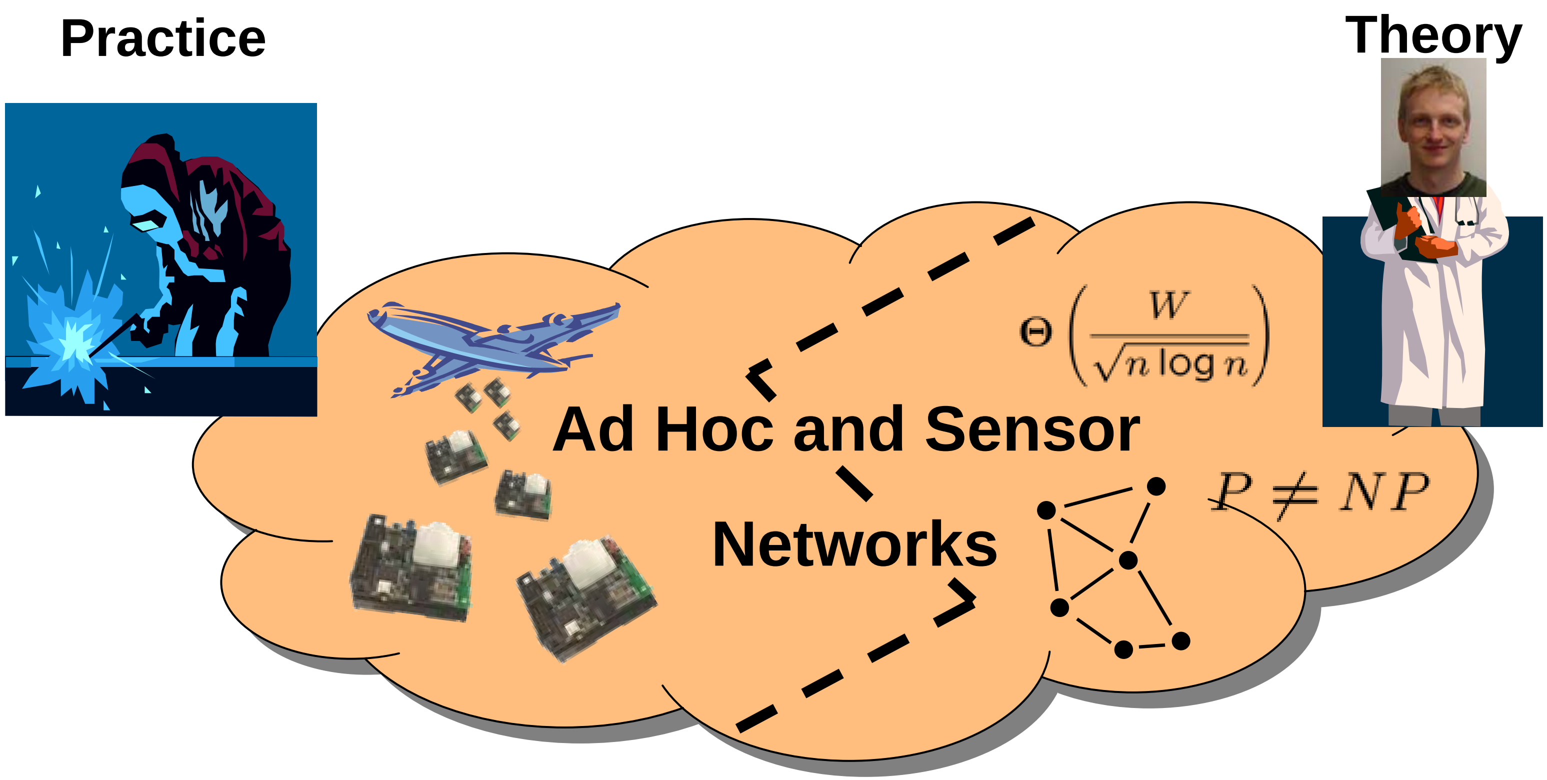
(((Notes Page)))

- Witch #1: The Chicken-and-Egg Problem
  - Dynamics…
- Witch #2: Power Control is Essential
  - UDG stimmt nicht…
- Witch #3: Network Models
- More material
  - Reading list on www.dcg.ethz.ch



AND YET THE QUESTION REMAINED:
"WHO CAME FIRST?"

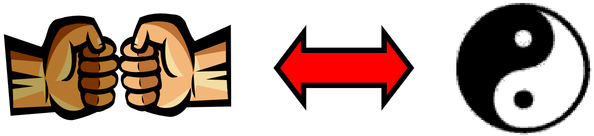
Of Theory and Practice...



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

Of Theory and Practice...

- What is the reason for this chasm...?



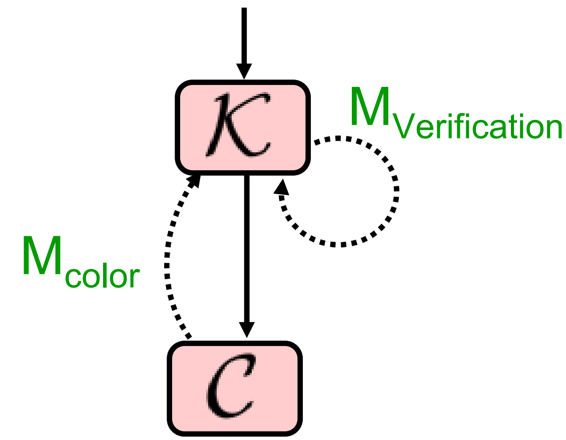
- Theoreticians try to understand the fundamentals
- Need to abstract away a few technicalities...

What are technicalities...???

- Abstracting away too many „technicalities“ renders theory useless for practice!

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

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

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

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

Cascading resets..?

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

Avoid Starvation - Idea

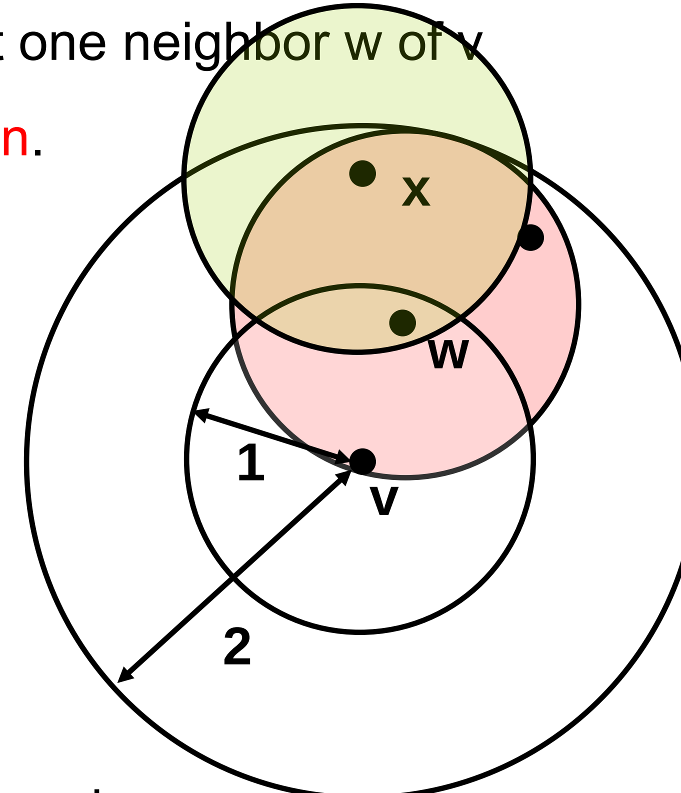
- Consider a node v entering state $\mathcal{K}$ at time t_v and verifying color c
- We show that by time $t_v+O(\Delta \log n)$, at least one neighbor w of v has transmitted (broadcast!) without collision.
- w has counter at least $\gamma\Delta \log n+1$
- All neighbors of w verifying c

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

Simulation

- The hidden constants in the big-O notation are quite big.
- Simulation shows that this is an artefact of „worst-case“ analysis.
- In reality, it is sufficient to set $\alpha := 10$.

  → Running time is at most $\boxed{t < 10 \cdot \log^2 n}$

With current hardware: BTnodes, Scatterweb, Mica2, etc.

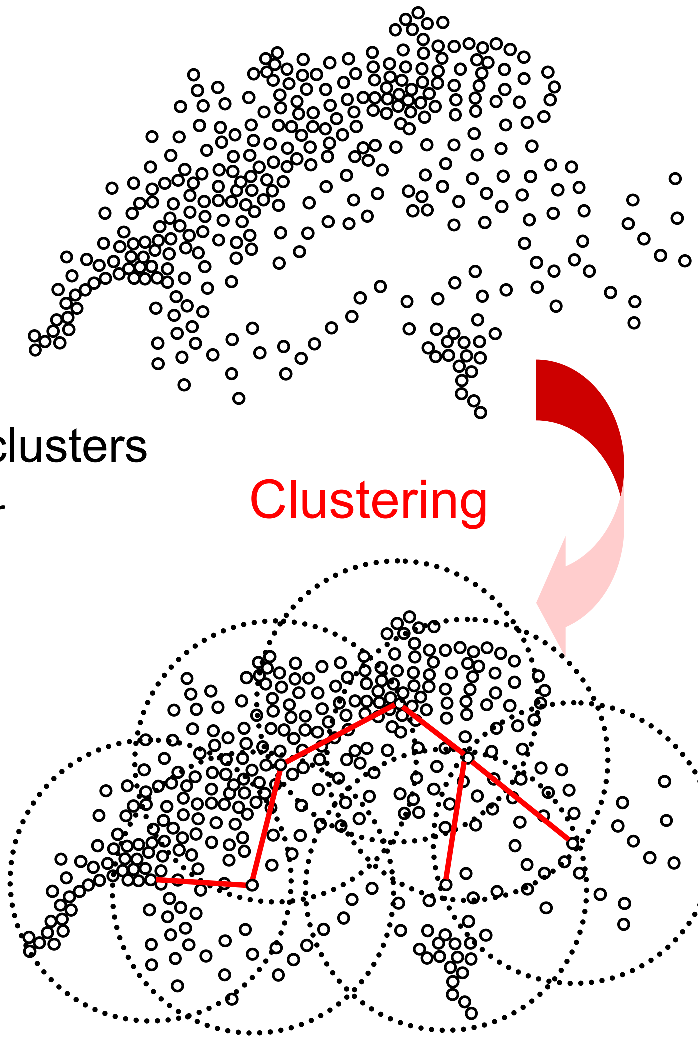
Raw transmission rate:	~ 115 kb/s
Switch time trans → recv:	~ 20 μs
Switch time recv → trans:	~ 12 μs
Paketsize of algorithm:	~20 Byte

→ Lenght of one time-slot is < 3 ms

Initializing 1000 nodes takes time < 3 seconds!

The Importance of Being Clustered...

- Clustering
  - Virtual Backbone for efficient routing
    → *Connected Dominating Set*
  - Improves usage of sparse resources
    → *Bandwidth, Energy, ...*
  - Spatial multiplexing in non-overlapping clusters
    → *Important step towards a MAC Layer*



Clustering helps in bringing structure into Chaos!

Dominating Set

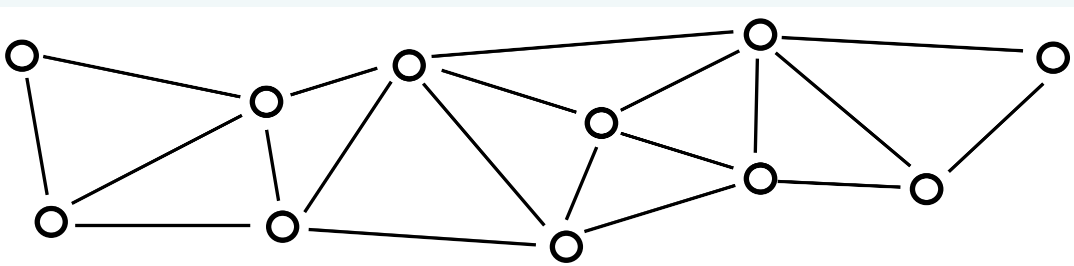
- Clustering:
  - Choose clusterhead such that:

    Each node is either a clusterhead or has a clusterhead in its communication range.

- When modeling the network as a graph G=(V,E), this leads to the well-known Dominating Set problem.

Dominating Set:

- A Dominating Set DS is a subset of nodes such that each node is either in DS or has a neighbor in DS.
- Minimum Dominating Set MDS is a DS of minimal cardinality.



Yet Another Dominating Set Algorithm...???

- There are many existing DS algorithms
  - [Kutten, Peleg, Journal of Algorithms 1998]
  - [Gao, et al., SCG 2001]
  - [Jia, Rajaraman, Suel, PODC 2001]
  - [Wan, Alzoubi, Frieder, INFOCOM 2002 & MOBIHOC 2002]
  - [Chen, Liestman, MOBIHOC 2002]
  - [Kuhn, Wattenhofer, PODC 2003]
  - .....
- Q: Why yet another clustering algorithm ?
- A: Other algorithms - with theoretical worst-case bounds - make too strong assumptions! (see previous slides...)

  → Not valid during initialization phase!

Overview

- Motivation
  Model

- Algorithm
  Analysis

- Conclusion
  Outlook

Clustering Algorithm - Results

- With three communication channels

> In expectation, our algorithm computes a $O\left(\frac{1}{d^2}\right)$ approximation for MDS in time
>
> $$O\left(\frac{\log N}{d^2}\left(\log \Delta + \frac{\log N}{\log \log N}\right)\right)$$

- Measurements suggest that 0.5 < d < 1.
  → Constant approximation!
- The time-complexity thus reduces to

$$O\left(\frac{\log^2 N}{\log \log N}\right) \quad \text{for} \quad 1 \leq \Delta \leq N^{1/\log \log N}$$

$$O(\log N \log \Delta) \quad \text{for} \quad N^{1/\log \log N} \leq \Delta \leq N$$

N : Upper bound on number of nodes in the network

Δ : Upper bound on number of nodes in a neighborhood (max. degree)

d : Quasi unit disk graph parameter

Clustering Algorithm – Basic Idea

- Use 3 independent communication channels Γ_1, Γ_2, and Γ_3.
  → Then, simulate these channels with a single channel.

- For the analysis: Assume time to be slotted
  → Algorithm does not rely on this assumption
  → Slotted analysis only a constant factor better than unslotted
  (similar to ALOHA)

Clustering Algorithm – Basic Structure

Upon wake-up do:

1) Listen for $\alpha \cdot \log^2 N/(d^2 \log\log N)$ time-slots on all channels

   upon receiving message → become dominated

   → stop competing to become dominator

2) For j=log Δ downto 0 do

   for $\alpha \cdot \log N/d^2$ slots, send with prob. $p_1 = \eta d^2 2^{-\log \Delta + j}$

   upon sending → become dominator

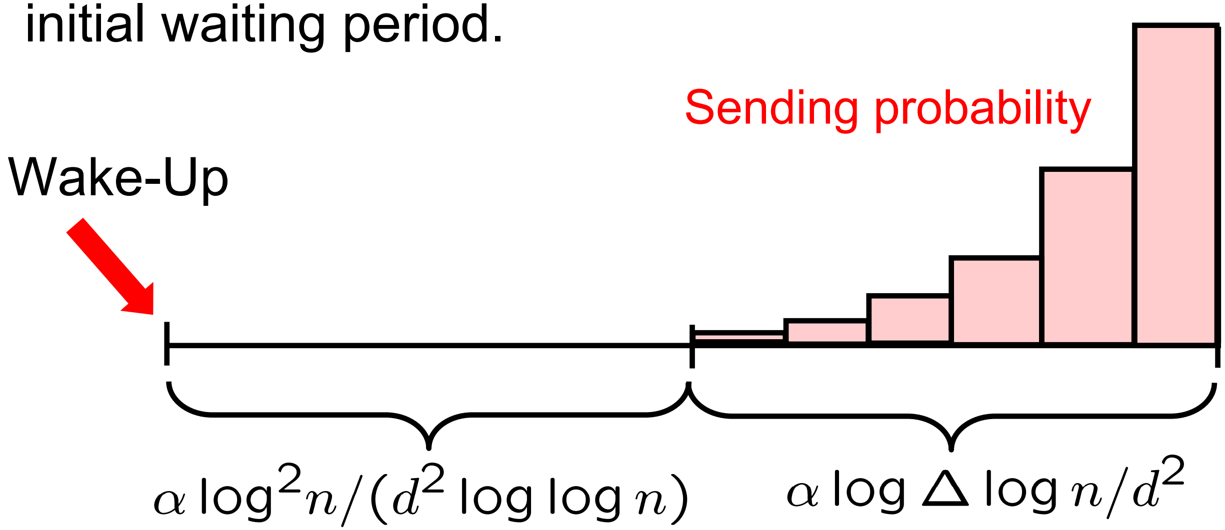
   upon receiving message → become dominated

   → stop competing to become dominator

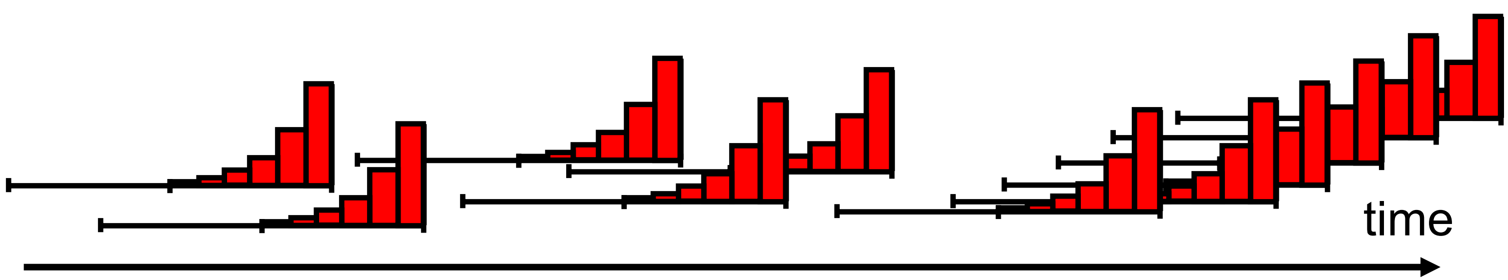
3) Additionally, dominators send on Γ_2 and Γ_3 with prob.
   $p_2 = \eta d^2 \log\log N/\log N$ and $p_3 = \eta d^2 \log\log N/\log^2 N$.

Clustering Algorithm – Basic Structure

- Each node's sending probability increases exponentially after an initial waiting period.

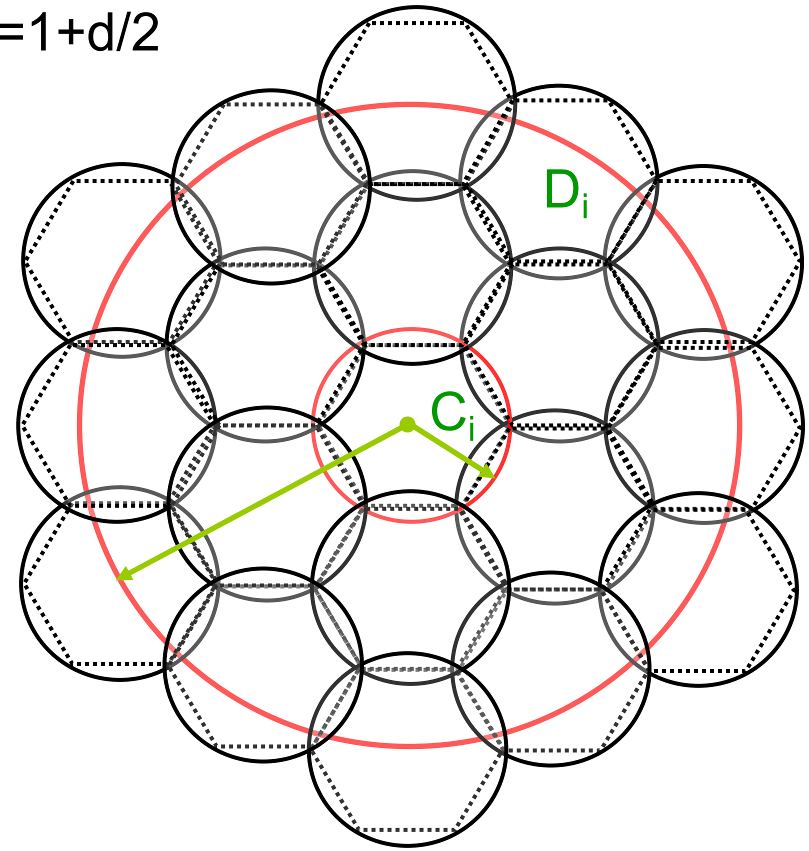


- Sequences are arbitrarily shifted in time (asynchronous wake-up)



Analysis - Outline

- Cover the plane with (imaginary) circles C_i of radius r=d/2
- Let D_i be the circle with radius R=1+d/2



- A node in C_i can hear all nodes in C_i
- Nodes outside of D_i cannot interfere with nodes in C_i

- We show: Algorithm has O(1) dominators in each C_i
- Optimum needs at least 1 dominator in D_i

Constant Approximation for constant d

Analysis - Outline

1. Bound the *sum of sending probabilities* in a circle C_i

   Remember: Due to asynchronous wake-up, every node may have a different sending probability

2. Bound the number of collisions in C_i before C_i becomes cleared

3. Bound the number of sending nodes per collision

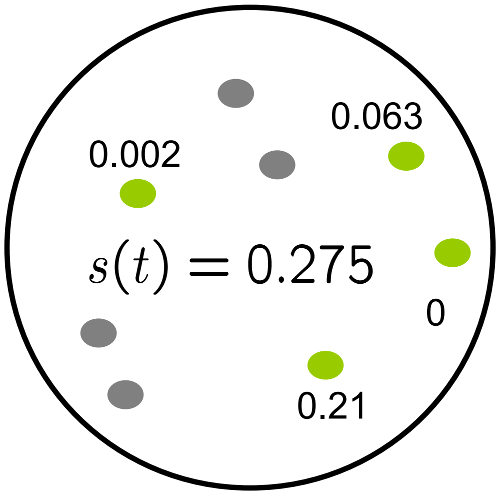
4. Newly awakened, already covered nodes will not become dominator

Analysis

Lemma 1: *Bound sum of sending probabilities in C_i*

- *Def:* Let $s(t)$ be the sum of sending probabilities of nodes in a circle C_i at time t, i.e.,

$$s(t) := \sum_{k \in C_i} p_k(t)$$

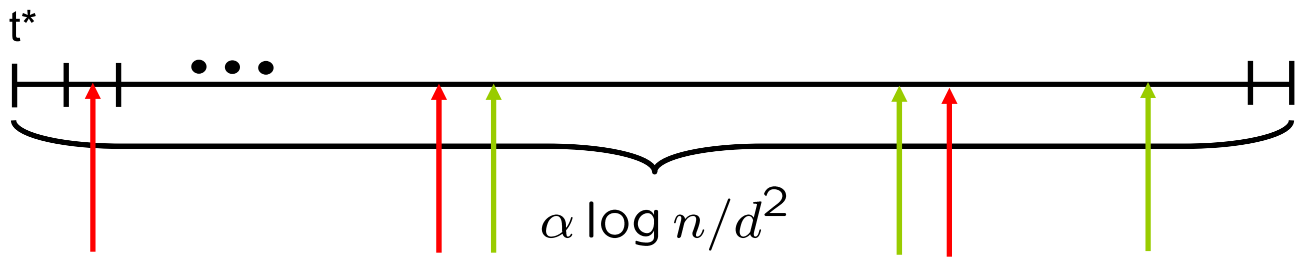


For all circles C_i and all times t, it holds that $s(t) \leq 3\eta d^2$ w.h.p.

Analysis

- Proof of Lemma 1:

- Induction over all time-slots when (for the first time) $s(t) > \eta d^2$ in a circle C_i. (Induction over multi-hop network!)
- Let t* be such a time-slot
- Consider interval $[t^*, \ldots, t^* + \alpha \log n/d^2 - 1]$

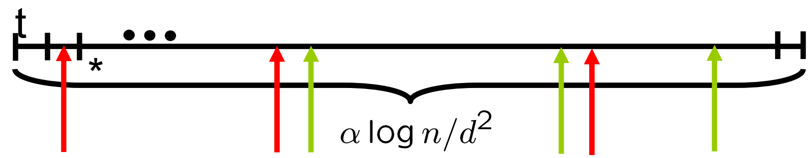


Nodes double their sending probability

New nodes start competing with initial sending probability

Analysis

- Proof of Lemma 1 (cont)
- Existing nodes can at most double
- New nodes send with very small probability



$$\boxed{s(t + \alpha \log n/d^2 - 1) \leq 3\eta d^2}$$

→ Next, we show in the paper that i[$t^*, \ldots, t^* + \alpha \log n/d^2 - 1$]

there will be at least one time-slot in which no node in $D_i \setminus C_i$,

and exactly one node in C_i sends.

→ After this time-slot, C_i is *cleared,* i.e., all (currently awake) nodes are decided.

→ Sum of sending probabilities does not exceed $3\eta d^2$

Analysis - Results

- For each circle C_i holds:
  - Number of dominators before a clearance in O(1) in expectation
  - Number of dominators after a clearance in O(1) w.h.p
  - → Number of dominators in C_i in O(1) in expectation

- Optimum has to place at least one dominator in D_i.

In expectation, the algorithm compute a $O(1/d^2)$ approximation.

- Reasonable values of d are constant → Constant approximation!

Three Channels → Single Channel

- Three independent communication channels not always feasible
- Simulation with a single channel is possible within O(polylog(n)).
- Idea:
  - Each node simulates each of its multi-channel time-slots with O(polylog(n)) single-channel time-slots.
  - It can be shown that result remains the same.

Algorithm compute a $O(1/d^2)$ approximation for MDS in polylogarithmic time even with a single communication channel.

Random Node Distribution

- Theoreticians often assume that, ...

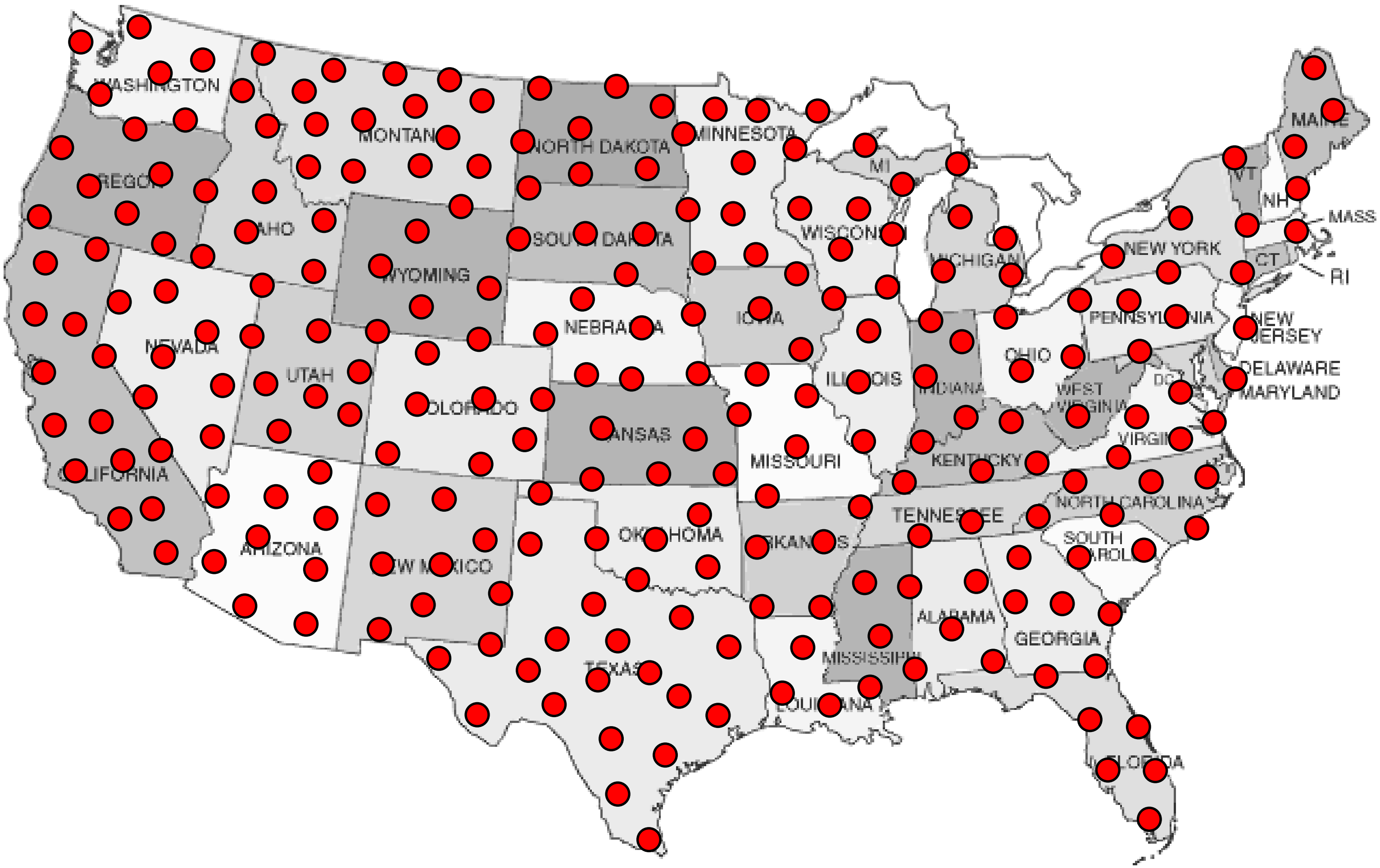
nodes are randomly, uniformly distributed in the plane.

This assumption allows for nice formulas

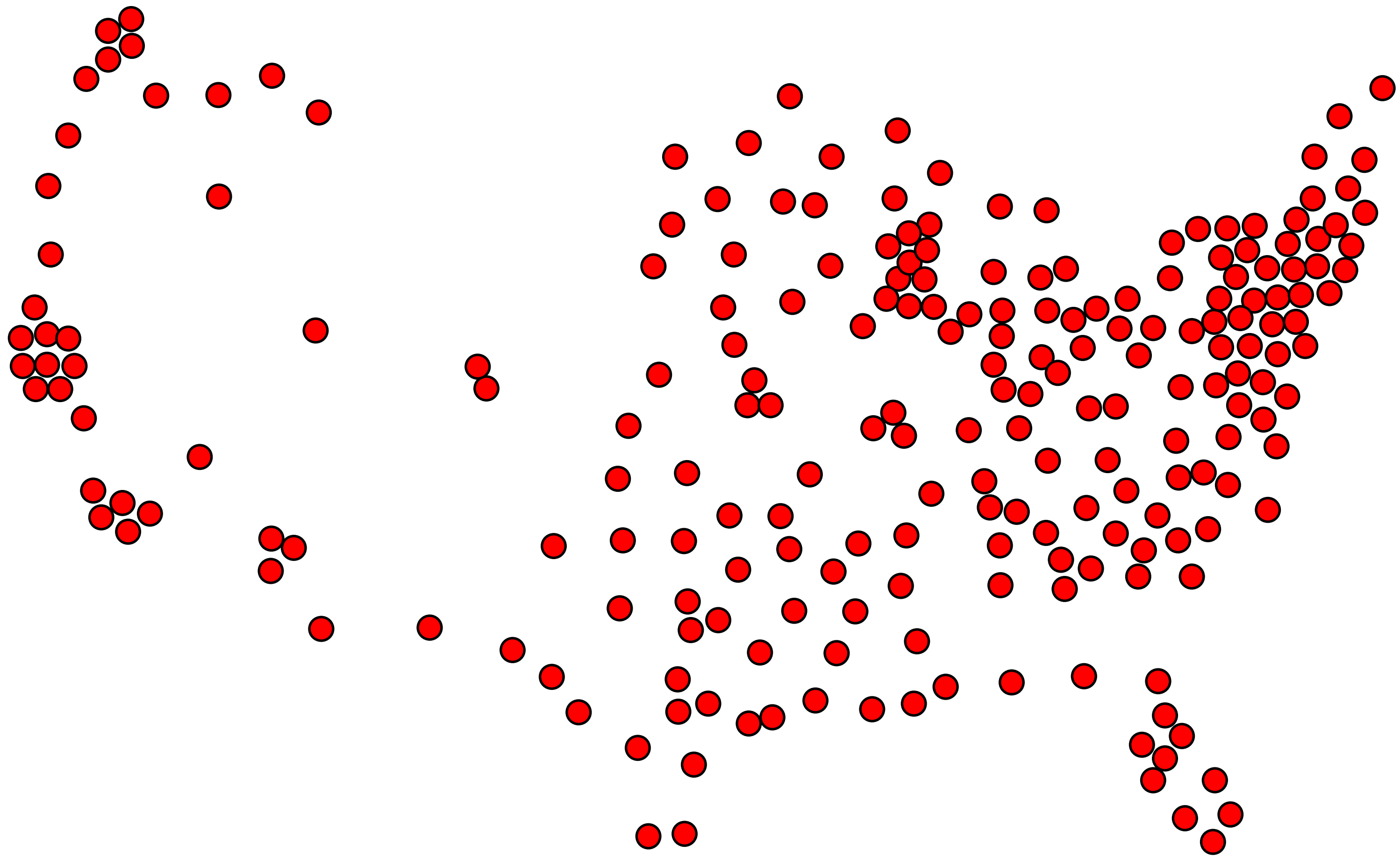
But is this really a „technicality“...?

How do real networks look like...?

Like this?


NORTH DAKOTA
WYOMING
UTAH
ARIZONA
IOWA
MI
NEW YORK
VT
NH
MASS
CT
RI
DELAWARE
MARYLAND
DC
GEORGIA

Or rather like this?



Random Node Distribution

- In theory, it is often assumed that, ...

nodes are randomly, uniformly distributed in the plane.

This assumption allows for nice formulas

Most small- and large-scale networks feature highly heterogenous node densities.

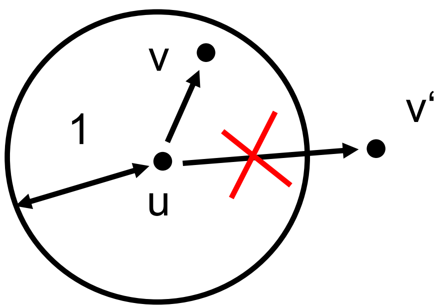
At high node density, assuming uniformity renders many practical problems trivial.

→ Not a technicality!

Unit Disk Graph Model

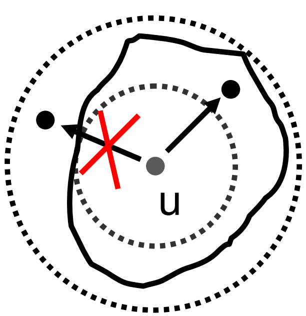
- In theory, it is often assumed that, ...

nodes form a unit disk graph!



Two nodes can communicate if they are within Euclidean distance 1.

This assumption allows for nice results



Signal propagation of real antennas not clear-cut disk!

Algorithms designed for unit disk graph model may not work well in reality. → Not a technicality!

Some complicated algorithm to compute not-quite-coloring

LP Approximation
Algorithm for Primal Node $v_i^{(p)}$:

$x_i := 0$;
for $e_p := k_p - 2$ **to** $-f - 1$ **by** -1 **do**
  for 1 **to** h **do**
    $(*\ \gamma_i := \frac{c_{\max}}{c_i} \sum_j a_{ji} r_j\ *)$
    for $e_d := k_d - 1$ **to** 0 **by** -1 **do**
      $\tilde{\gamma}_i := \frac{c_{\max}}{c_i} \sum_j a_{ji} \tilde{r}_j$;
      if $\tilde{\gamma}_i \geq 1/\Gamma_p^{e_p/k_p}$ **then**
        $x_i^+ := 1/\Gamma_d^{e_d/k_d}$; $x_i := x_i + x_i^+$;
      fi;
      send $x_i^+, \tilde{\gamma}_i$ to dual neighbors;
      receive $\tilde{r}_j$ from dual neighbors
    od;
    receive r_j from dual neighbors
  od
od;
$x_i := x_i / \min_{j \in N_i^{(p)}} \sum_\ell a_{j\ell} x_\ell$

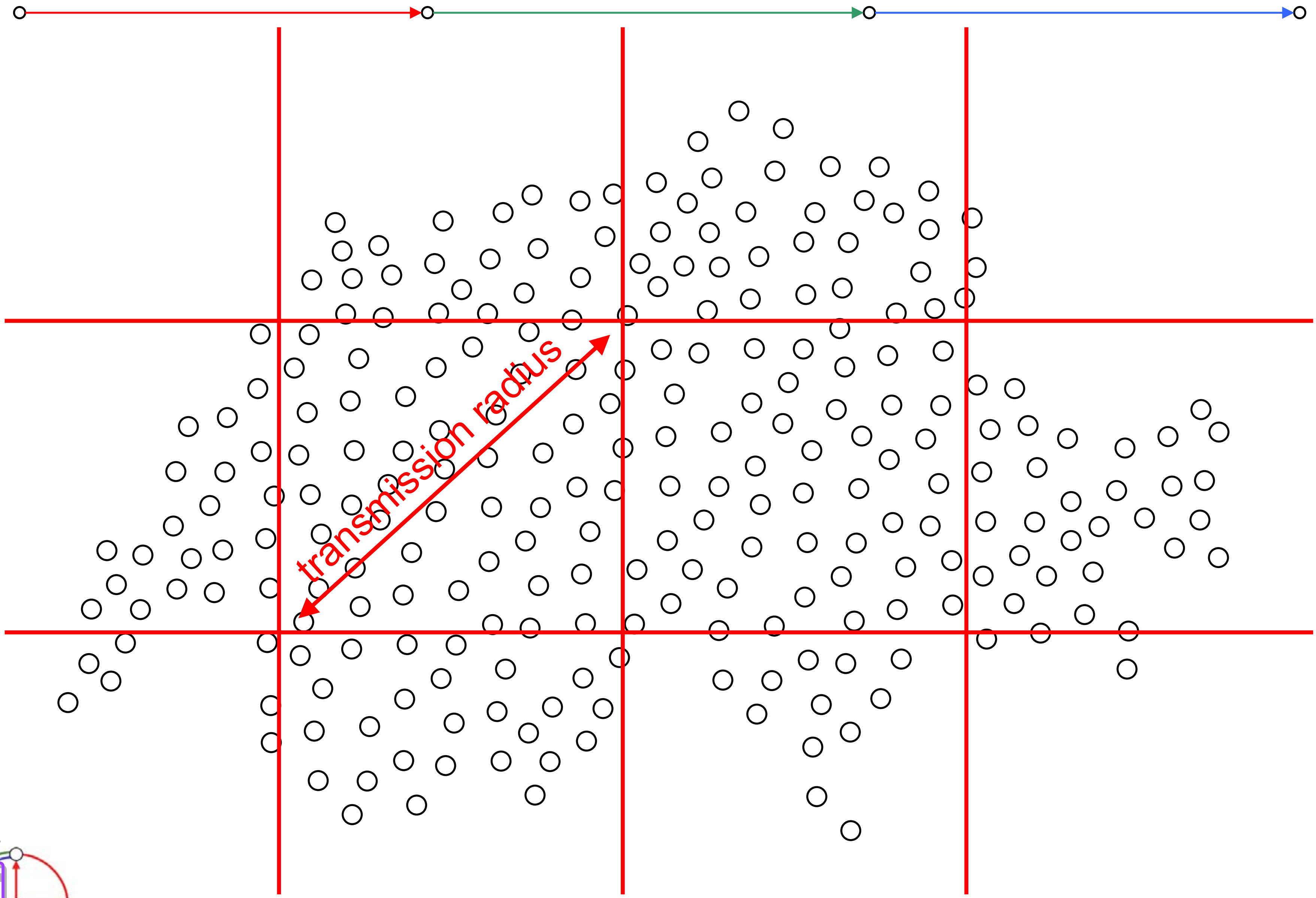
LP Approximation
Algorithm for Dual Node $v_i^{(d)}$:

$y_i := y_i^+ := w_i := f_i := 0$; $r_i := 1$;
for $e_p := k_p - 2$ **to** $-f - 1$ **by** -1 **do**
  for 1 **to** h **do**
    $\tilde{r}_i := r_i$;
    for $e_d := k_d - 1$ **to** 0 **by** -1 **do**
      receive $x_j^+, \tilde{\gamma}_j$ fro
      $y_i^+ := y_i^+ + \tilde{r}_i \sum_j$
      $w_i^+ := \sum_j a_{ij} x_j^+$;
      $w_i := w_i + w_i^+$; f_i
      if $w_i \geq 1$ **then** $\tilde{r}_i$:
      send $\tilde{r}_i$ to primal n
    od;
    increase_duals();
    send r_i to primal nei
  od
od;
$y_i := y_i / \max_{j \in N_i^{(d)}} \frac{1}{c_j} \sum$

procedure increase_duals():

if $w_i \geq 1$ **then**
  if $f_i \geq f$ **then**
    $y_i := y_i + y_i^+$; $y_i^+ := 0$;
    $r_i := 0$; $w_i := 0$
  else if $w_i \geq 2$ **then**
    $y_i := y_i + y_i^+$; $y_i^+ := 0$;
    $r_i := r_i / \Gamma_p^{\lfloor w_i \rfloor / k_p}$
  else
    $\lambda := \max\{\Gamma_d^{1/k_d}, \Gamma_p^{1/k_p}\}$;
    $y_i := y_i + \min\{y_i^+, r_i \lambda / \Gamma_p^{e_p/k_p}\}$;
    $y_i^+ := y_i^+ - \min\{y_i^+, r_i \lambda / \Gamma_p^{e_p/k_p}\}$;
    $r_i := r_i / \Gamma_p^{1/k_p}$
  fi;
  $w_i := w_i - \lfloor w_i \rfloor$
fi

A much simpler algorithm to compute 2-hop-coloring



Algorithm 2 TODO!

1. Each cell, depending on position, has a unique predefined number between 0 and 15.

2. Fetch a not-yet-taken small integer in your cell

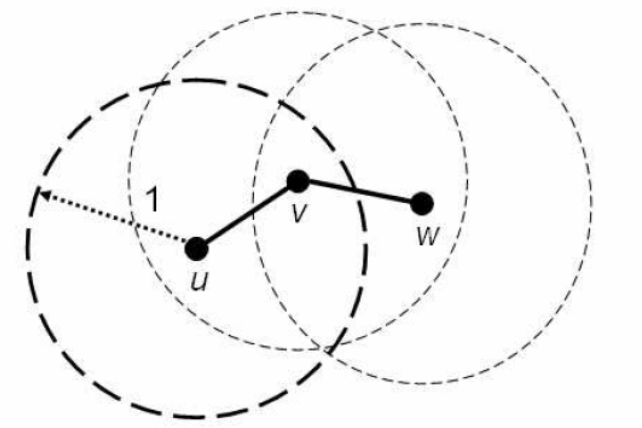
3. Your color is your number plus

4. That's it.

Connectivity (1)

- Which nodes are adjacent to a given node *v*?

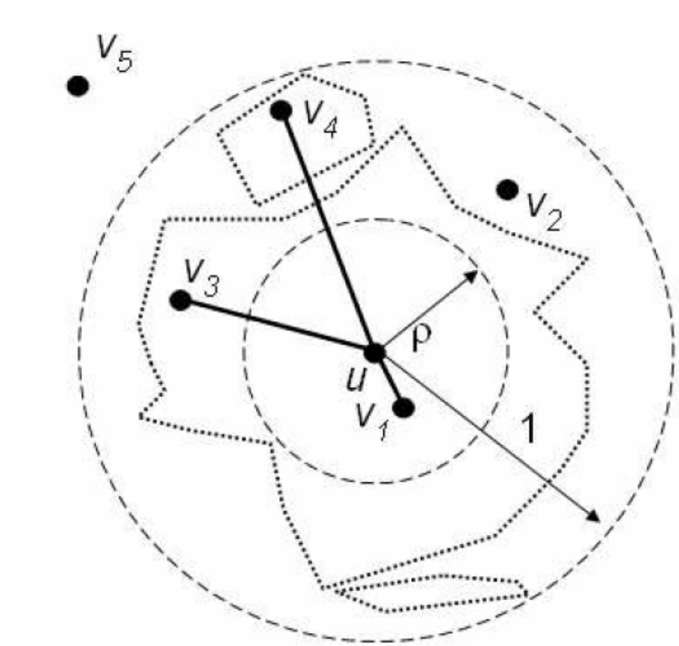
- Example: Unit Disk Graph
  - Classic Model from computational geometry
  - $\{u,v\} \in E \Leftrightarrow |u,v| \leq 1$



- Pro
  - Very simple
  - Analytically tractable
  - Realistic in unobstructed environments
- Contra
  - Too simple
  - Not realistic in inner-city networks with many buildings etc.

Connectivity (2)

- More realistic: the *Quasi* UDG (QUDG)
  - $\{u,v\} \in E \Leftrightarrow |u,v| \leq \rho$
  - $\{u,v\} \notin E \Leftrightarrow |u,v| > 1$
  - otherwise: It depends!

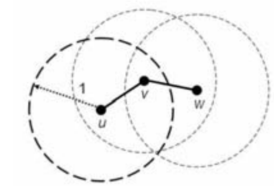


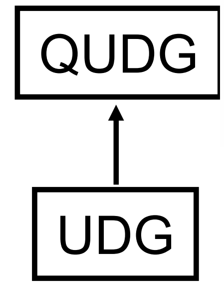
- It depends…
  - … on an adversary,
  - … on probabilistic model,
  - etc.!

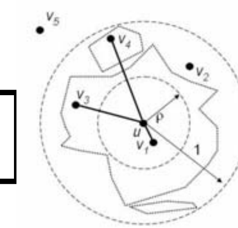
- Advantage: Accounts for a certain flexibility

Connectivity Put into Perspective (1)

- Fact: UDG is a QUDG
  - $\rho = 1$

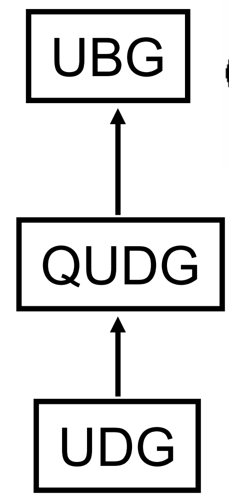






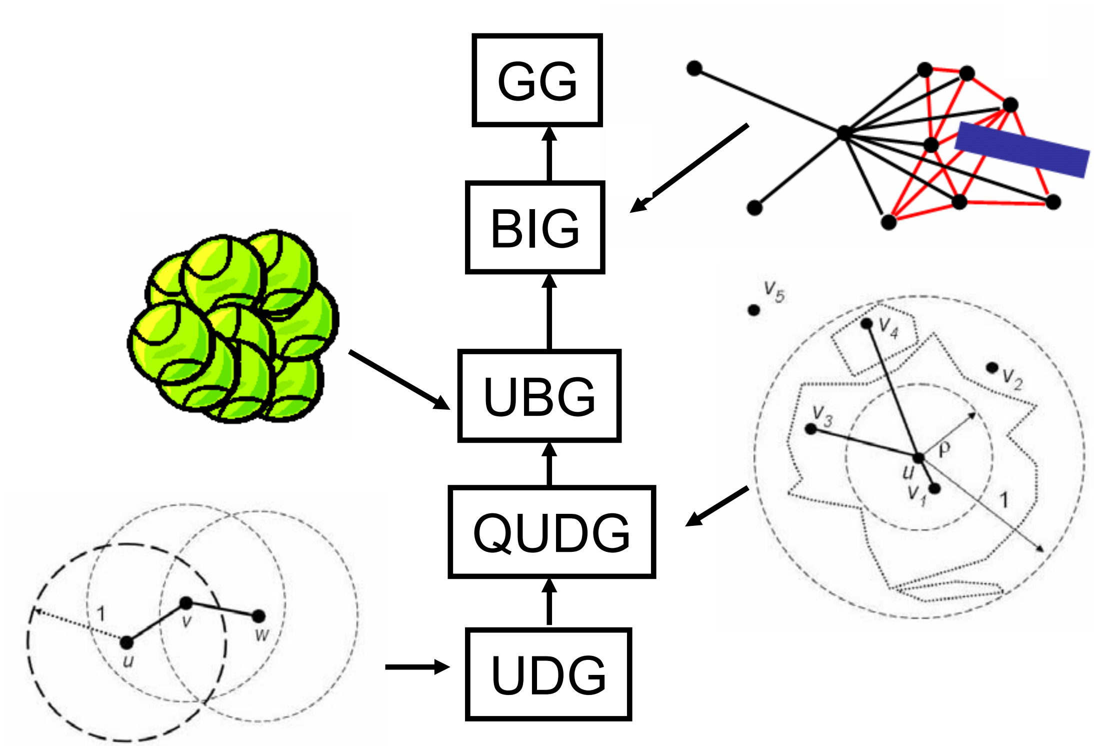
- Fact: However, in the QUDG with constant ρ, the set of nodes in radius *r* can always be covered by a constant number of balls of radius *r/2* and hence:

- Fact: QUDG is a UBG



Connectivity Put into Perspective (2)

- Fact: The size of the independent sets of any UBG is polynomially bounded, i.e., the UBG is a BIG.
- Finally, a BIG is of course a special kind of a general graph (GG).



MACbeth

The Three Witches of Media Access Theory

Roger Wattenhofer

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich