Algorithms for Sensor Networks
…What Is It Good For?!

Absolutely nothing?!?

Hypothesis: Impact(Theory) → ϵ

Scoring for Theory

- "Theory is important, even if it sometimes does not have impact"
  - sometimes decades later, e.g., number theory for cryptography

- Packet switching (very important for sensor networks) was promoted by theory guys in the early 60s:
  - Paul Baran, Donald Davies, Leonard Kleinrock, et al.
  - Later followed by Lawrence Roberts, Robert Kahn, Vinton Cerf, et al.

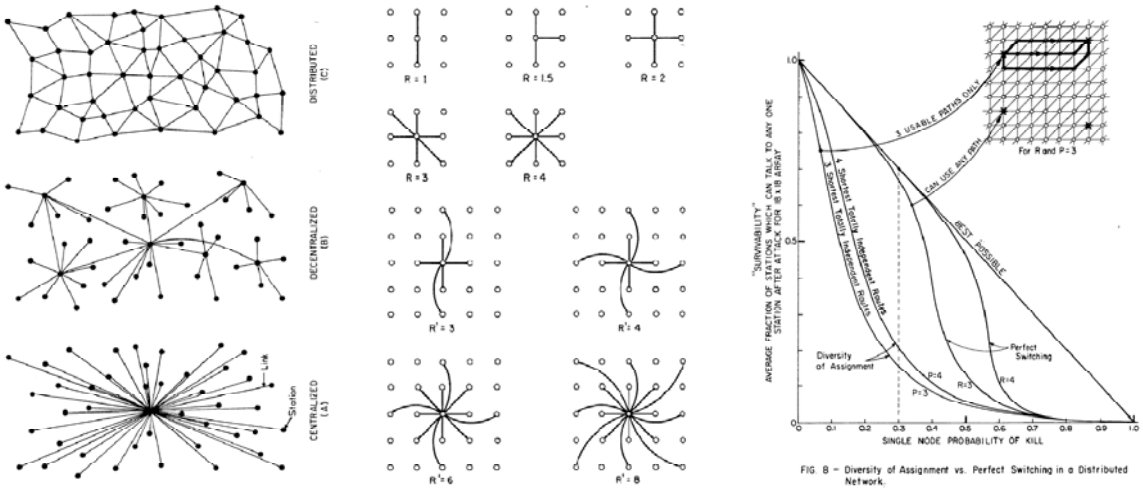


FIG. 8 – Diversity of Assignment vs. Perfect Switching in a Distributed Network.

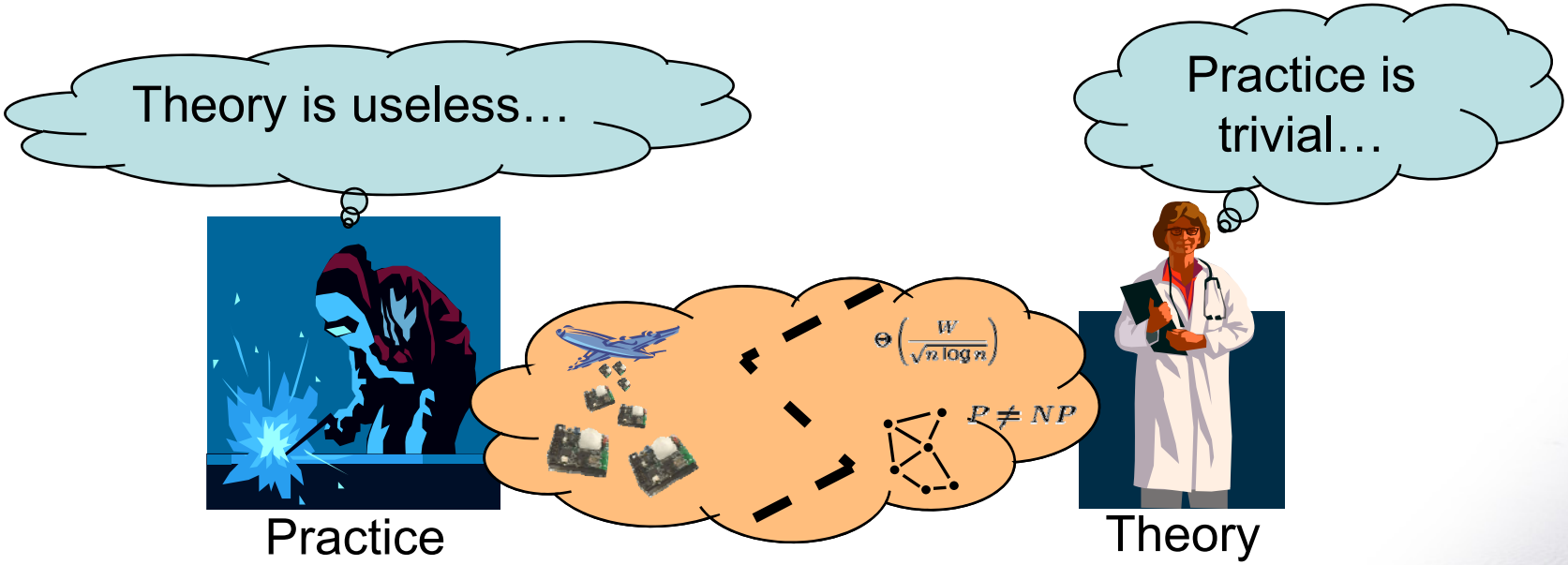
Scoring for Systems

- Baran et al. was almost 50 years ago
- Systems people get it “right” quite often…
- Many important difficult problems are “not really theoretical”…

Impact(Recent Theory) → ϵ!

Why? (More theory whining)

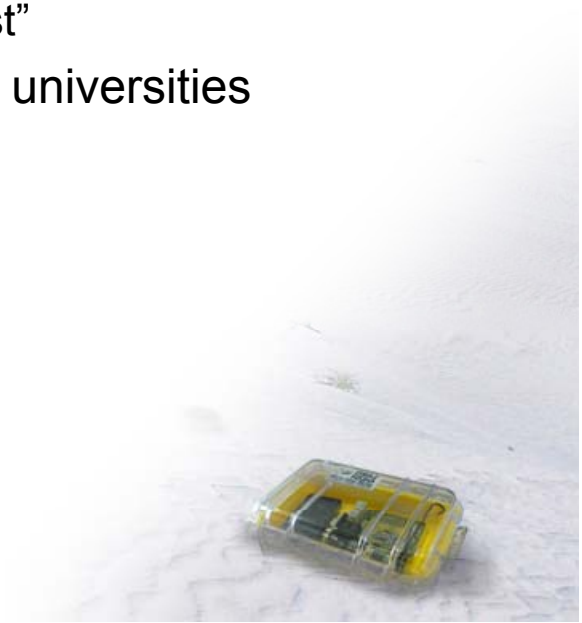
- Why does theory not have impact on practical systems?



Systems people don't read theory papers

- Sometimes for good reasons…
  - unreadable
  - don't matter that much (only getting out the last %)
  - wrong models
  - theory is lagging behind
  - bad theory merchandising/branding
    - systems papers provide easy to remember acronyms
    - "On the Locality of Bounded Growth" vs. "Smart Dust"
  - good theory also comes from outside the top 5 US universities
    - having hundreds of workshops does not help, is just a good excuse for not following up research

- … do I sound embittered?!? :-)



Why recent theory does not have impact on real systems...

1) Systems people don't read theory papers

2) Theory people don't build systems

Maybe theory people should build systems themselves?!?

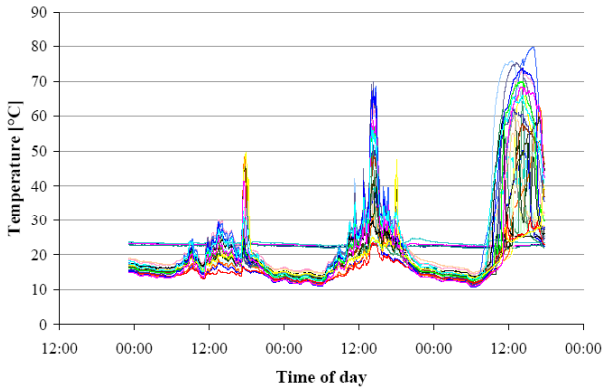
3) Ergo, theory does not have practical impact

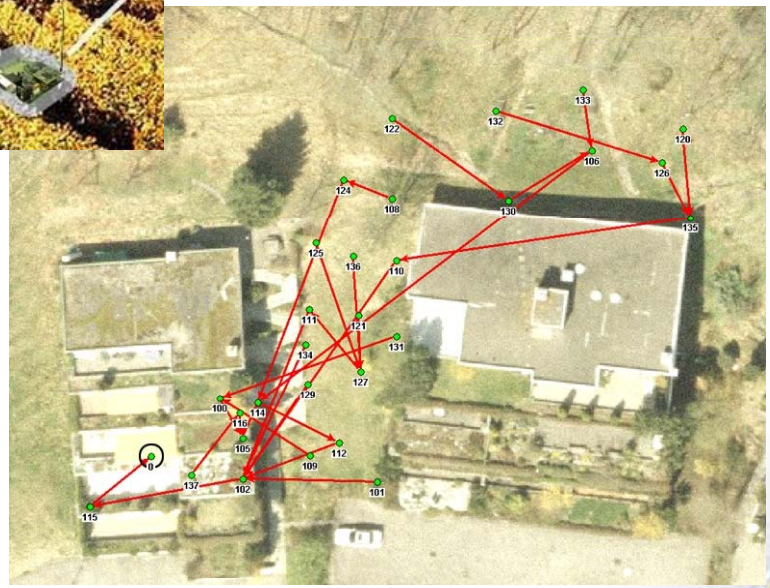
Systems Perspective: Dozer

Example: Dozer

[Burri, von Rickenbach, W, IPSN 2007]

- Up to 10 years of network life-time
- Mean energy consumption: 0.066 mW
- Operational network in use > 2 years
- High availability, reliability (99.999%)





Is Dozer a theory-meets-systems success story?

- Good news
  - Theory people can develop good systems!
  - Dozer is to the best of my knowledge more energy-efficient and reliable than all other published systems protocols…
  - For more than 2 years already!
- Bad news
  - Dozer does not have an awful lot of theory inside
- Ugly news
  - Dozer v2 has even less theory than Dozer v1
- Hope
  - Despite not being aware still subliminal theory ideas in system?

Energy-Efficient Protocol Design

- Communication subsystem is the main energy consumer
  - Power down radio as much as possible

TinyNode	Power Consumption
uC sleep, radio off	0.015 mW
Radio idle, RX, TX	30 – 40 mW

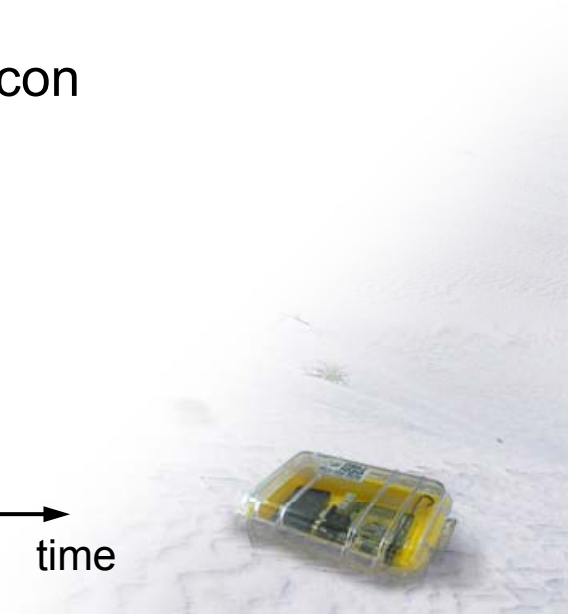
- Issue is tackled at various layers
  - MAC
  - Topology control / clustering
  - Routing

➡ Orchestration of the whole network stack to achieve duty cycles of ~1‰

Dozer System

- Tree based routing towards data sink
  - No energy wastage due to multiple paths
  - Current strategy: SPT
- TDMA based link scheduling
  - Each node has two independent schedules
  - No global time synchronization

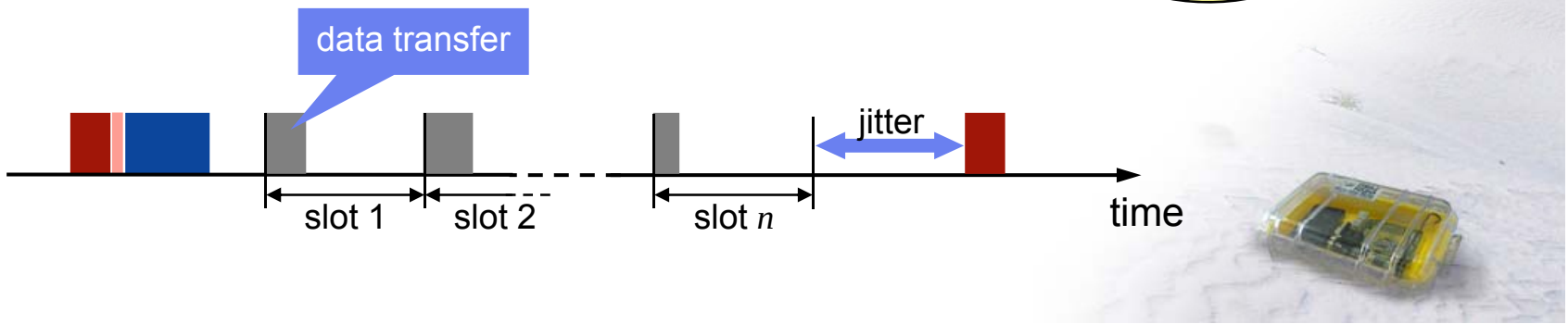
- The parent initiates each TDMA round with a beacon
  - Enables integration of disconnected nodes
  - Children tune in to their parent's schedule



Dozer System

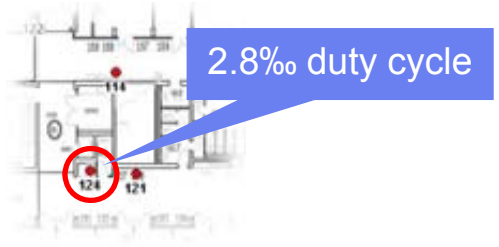
- Parent decides on its children data upload times
  - Each interval is divided into upload slots of equal length
  - Upon connecting each child gets its own slot
  - Data transmissions are always ack'ed

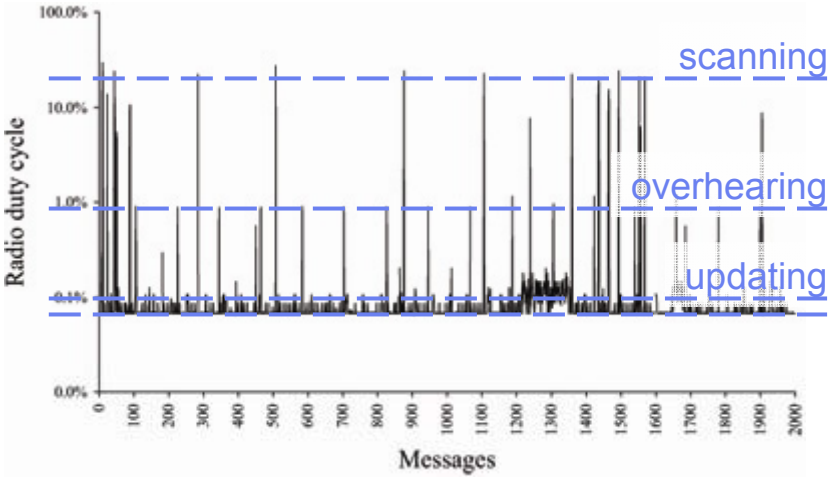
- No traditional MAC layer
  - Transmissions happen at exactly predetermined point in time
  - Collisions are explicitly accepted
  - Random jitter resolves schedule collisions



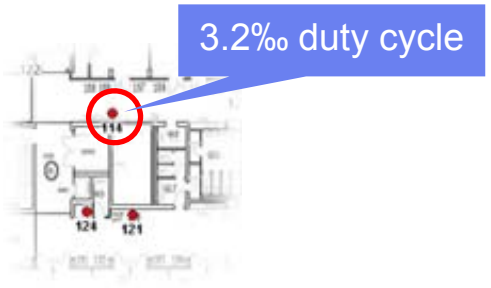
Dozer in Action

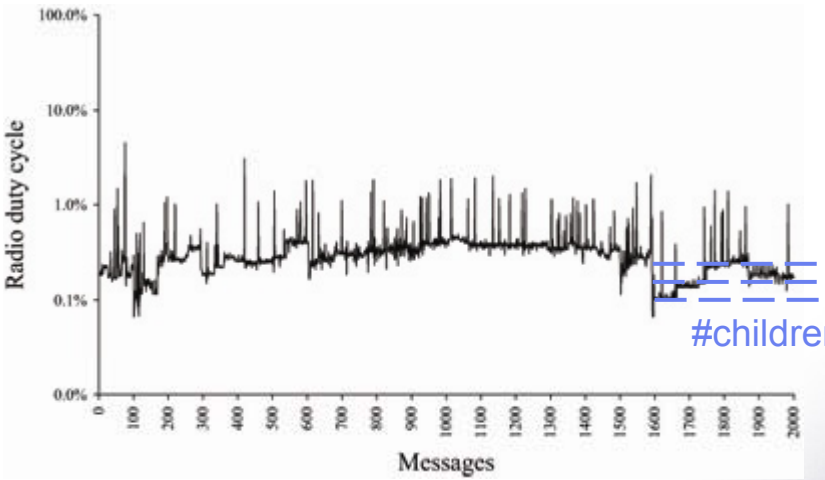
Energy Consumption



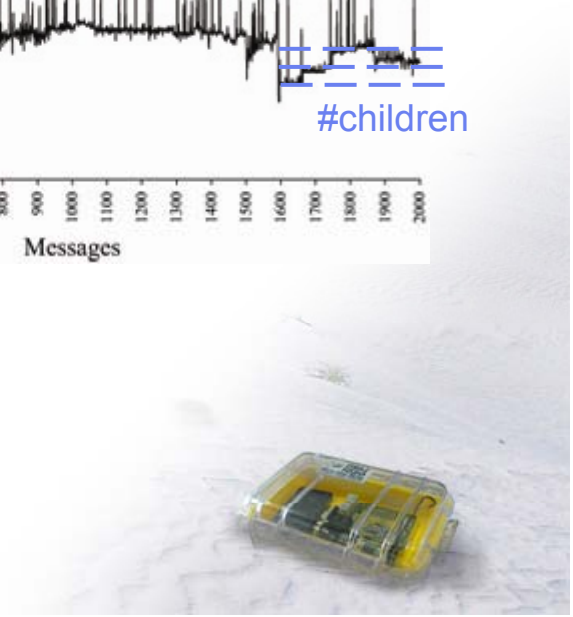


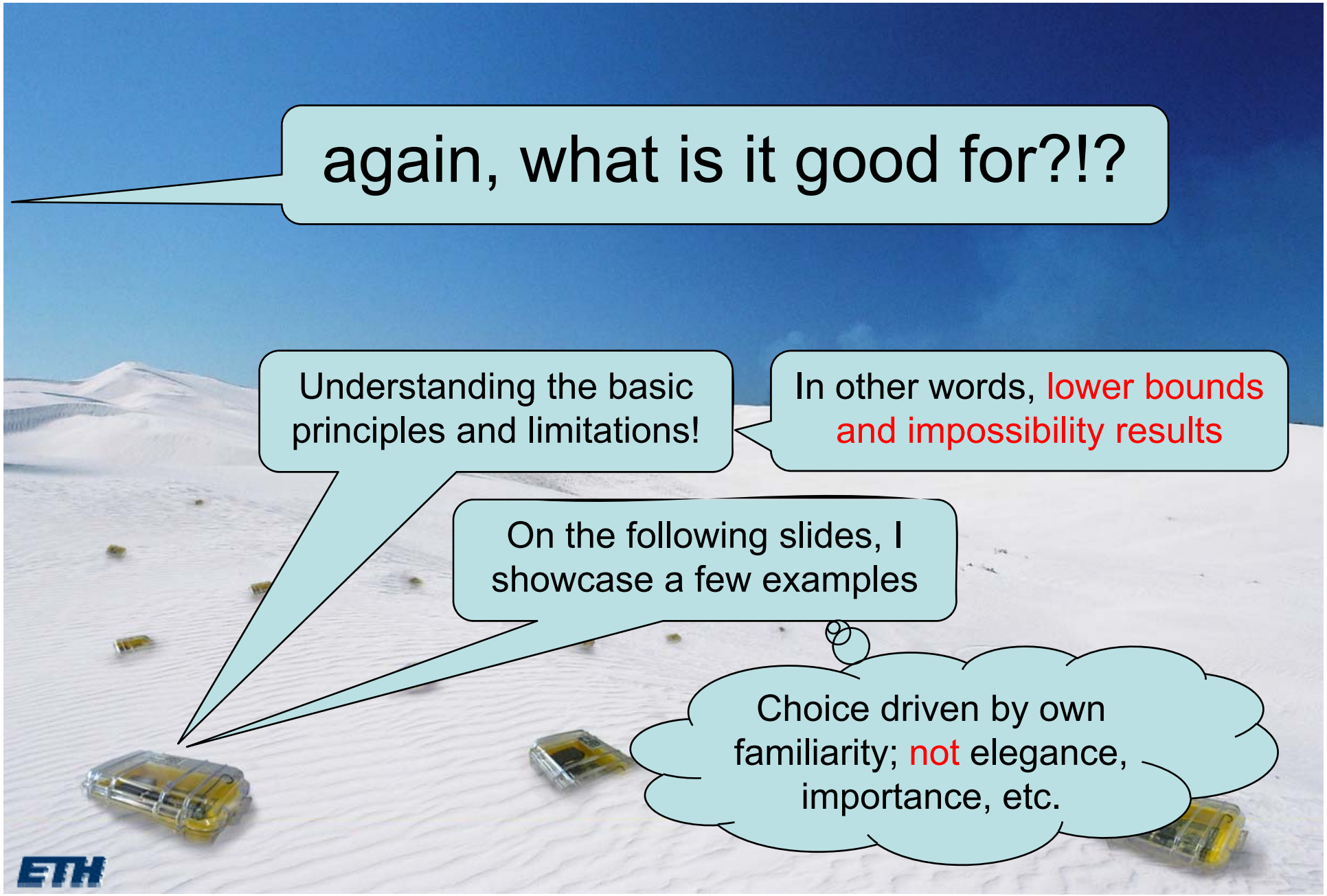
- Leaf node
- Few neighbors
- Short disruptions





- Relay node
- No scanning



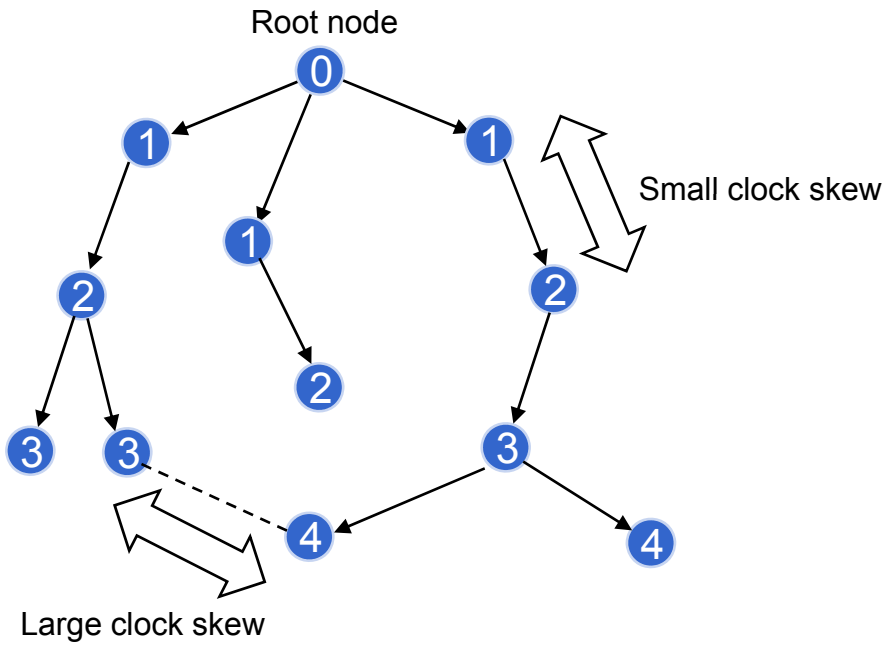

again, what is it good for?!?
Understanding the basic principles and limitations!
In other words, lower bounds and impossibility results
On the following slides, I showcase a few examples
Choice driven by own familiarity; not elegance, importance, etc.

Time Synchronization

Clock Synchronization

- Problem: Clocks have offset/drift, messages have variable delays

1. Global property: Minimize clock skew between any two nodes
2. Local ("gradient") property: Small clock skew between two nodes if the distance between the nodes is small.
3. Clock should not be allowed to jump backwards
   - You don't want new events to be registered earlier than older events.



Trivial Solution: Let t = 0 at all nodes and times

- Problem: Clocks have offset/drift, messages have variable delays

1. Global property: Minimize clock skew between any two nodes
2. Local (gradient) property: Small clock skew between two nodes if the distance between the nodes is small
3. Clock should not be allowed to jump backwards

- To prevent trivial solution, we need a fourth constraint:

4. Clock should always to move forward.
   - Sometimes faster, sometimes slower is OK.
   - But there should be a minimum and a maximum speed.

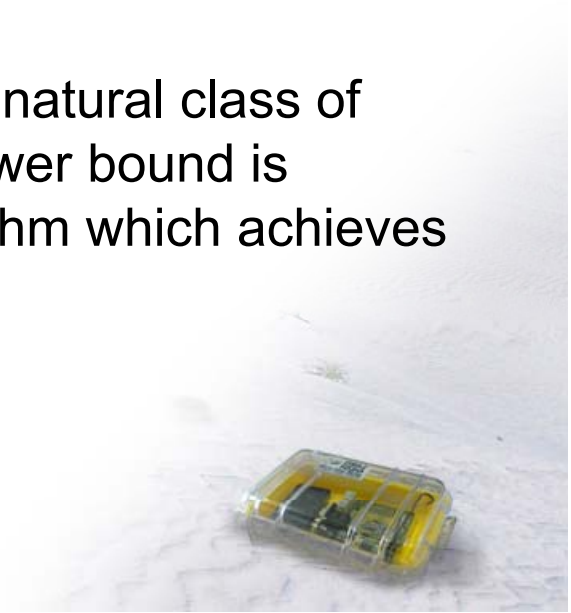
Results

- All natural/proposed clock synchronization algorithms seem to fail horribly, having at least linear skew between neighbor nodes.

- Indeed [Fan, Lynch, PODC 2004] show that when logical clocks need to obey minimum/maximum speed rules, the skew of two neighboring clocks can be up to $\Omega(\log D / \log \log D)$, where D is the diameter of the network; updated by [Meier, Thiele, PODC 2005]

- Later [Locher, W, DISC 2006] show that a for the natural class of oblivious clock synchronization algorithms, the lower bound is $\Omega(\sqrt{D})$. Also they present a new (oblivious) algorithm which achieves $O(\sqrt{D})$.

- Nice open problem…?
  [Lenzen, Locher, W, FOCS 2008]



Data Gathering

Distributed Aggregation

Growing interest in distributed aggregation!

→ Sensor networks, distributed databases...

Aggregation functions?
→ *Distributive* (max, min, sum, count)
→ *Algebraic* (plus, minus, average)
→ *Holistic* (median, k^{th} smallest/largest value)

Combinations of these functions enable *complex queries*!
→ „What is the average of the 10% largest values?“

What cannot be computed using these functions?

Aggregation Model

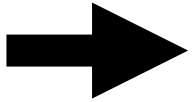
How difficult is it to compute these aggregation primitives?

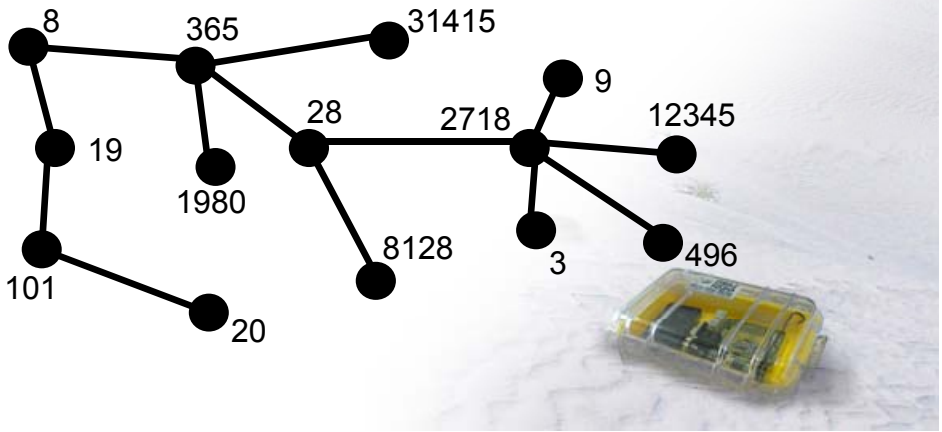
Simple breadth-first construction!

Can easily be generalized to an arbitrary number of elements!

Model:

- Connected graph G = (V,E) of diameter D_G, |V| = n.
- Nodes v_i and v_j can communicate directly if $(v_i,v_j) \in E$.
- A spanning tree is available (diameter $D \leq 2D_G$)°
- Asynchronous model of communication.
- All nodes hold a single element. °
- Messages can contain only a constant number of elements.





Distributive & Algebraic Functions

How difficult is it to compute these aggregation primitives?

→ We are interested in the time complexity!

Worst-case for every legal input and every execution scenario!

Slowest message arrives after 1 time unit!

→ *Distributive* (sum, count...) and *algebraic* (plus, minus...) functions are easy to compute:

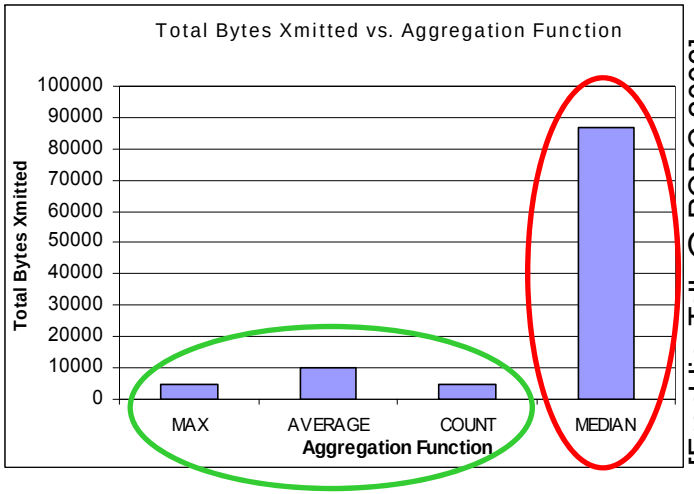
Use a simple *flooding-echo* procedure → ***convergecast***!

Time complexity: $\Theta(D)$

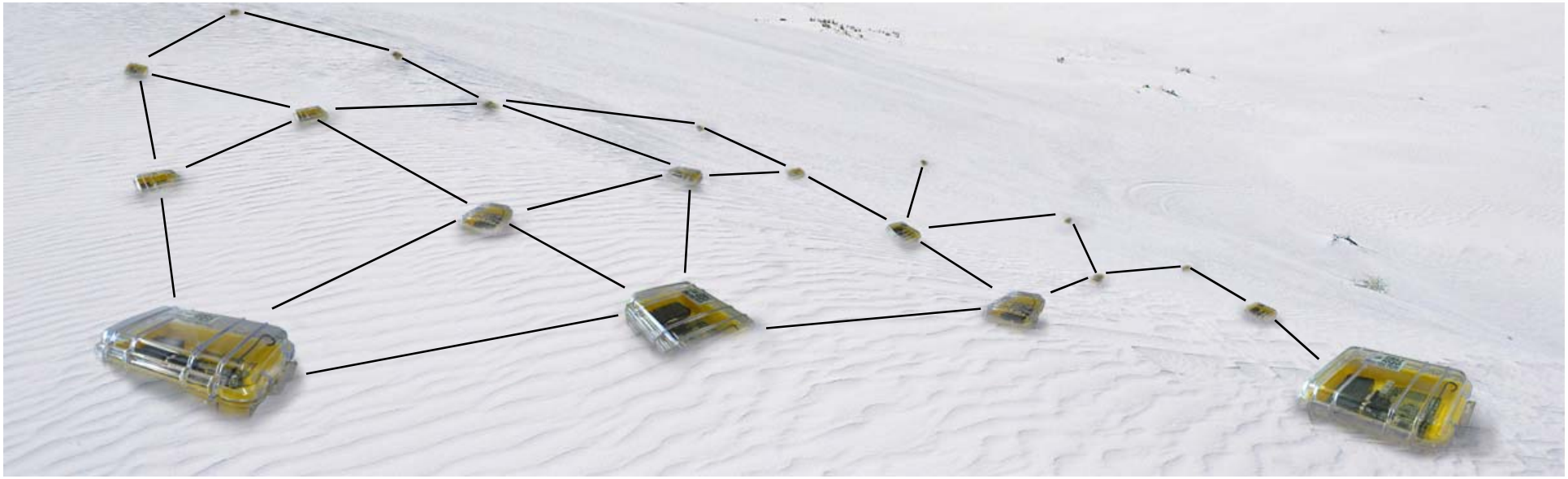
What about holistic functions (such as k-selection)???
Is it (really) harder...?
Impossible to perform in-network aggregation?

Distributed Selection

- Database requests („SELECT ...“) consist of combinations of functions such as MAX, AVG, COUNT, k^{th} largest, etc.
- In a (sensor) network, most functions are trivially computable in diameter time.
- Only selection (median, k^{th} largest, 90% smallest values, etc.) is considered to be impossible (or at least difficult).



[Franklin, Talk @ PODC 2003]

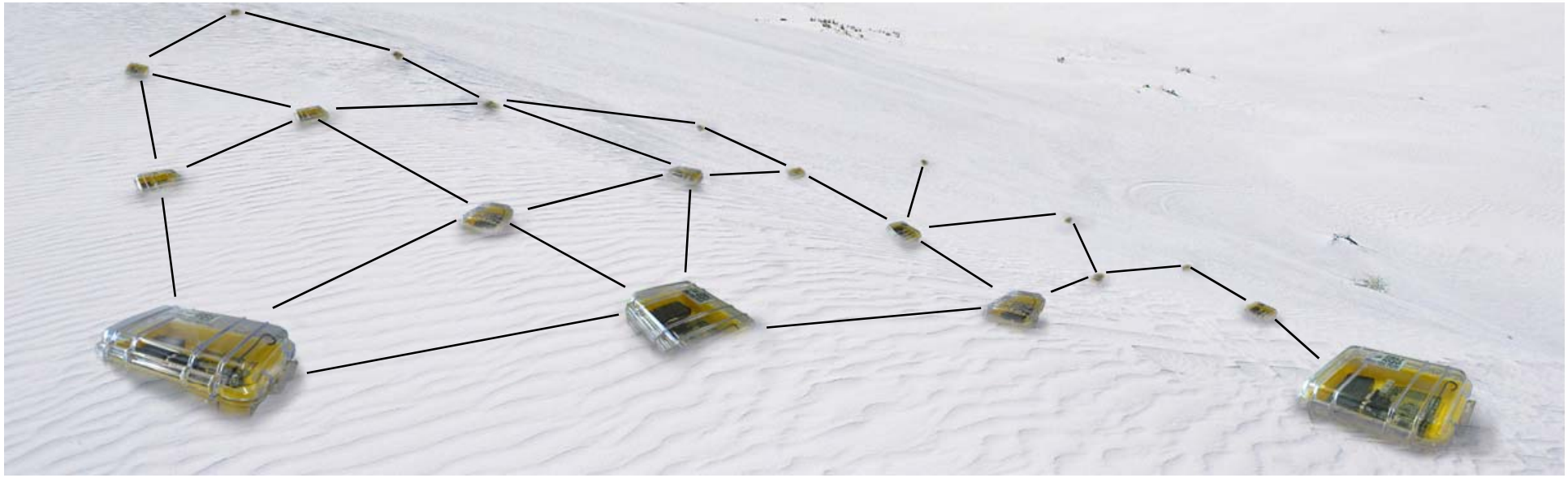


Results

- [Locher, Kuhn, W, SPAA 2007] showed that

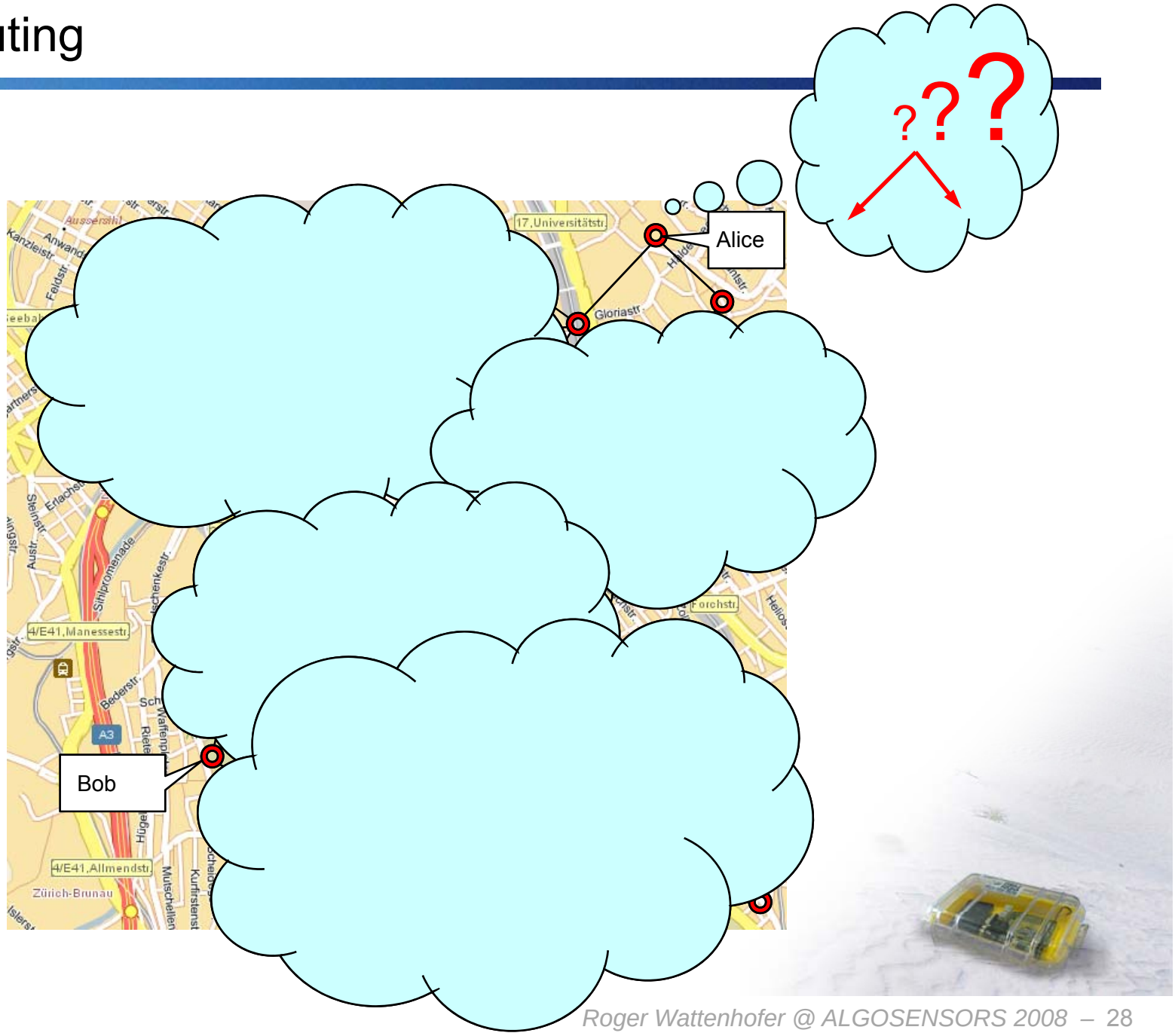
Selection can be done in time $O(D \cdot \log_D n)$. This is asymptotically optimal as there is a matching $\Omega(D \cdot \log_D n)$ lower bound. For deterministic algorithms: $O(D \cdot \log^2_D n)$.

D = diameter
n = # of nodes

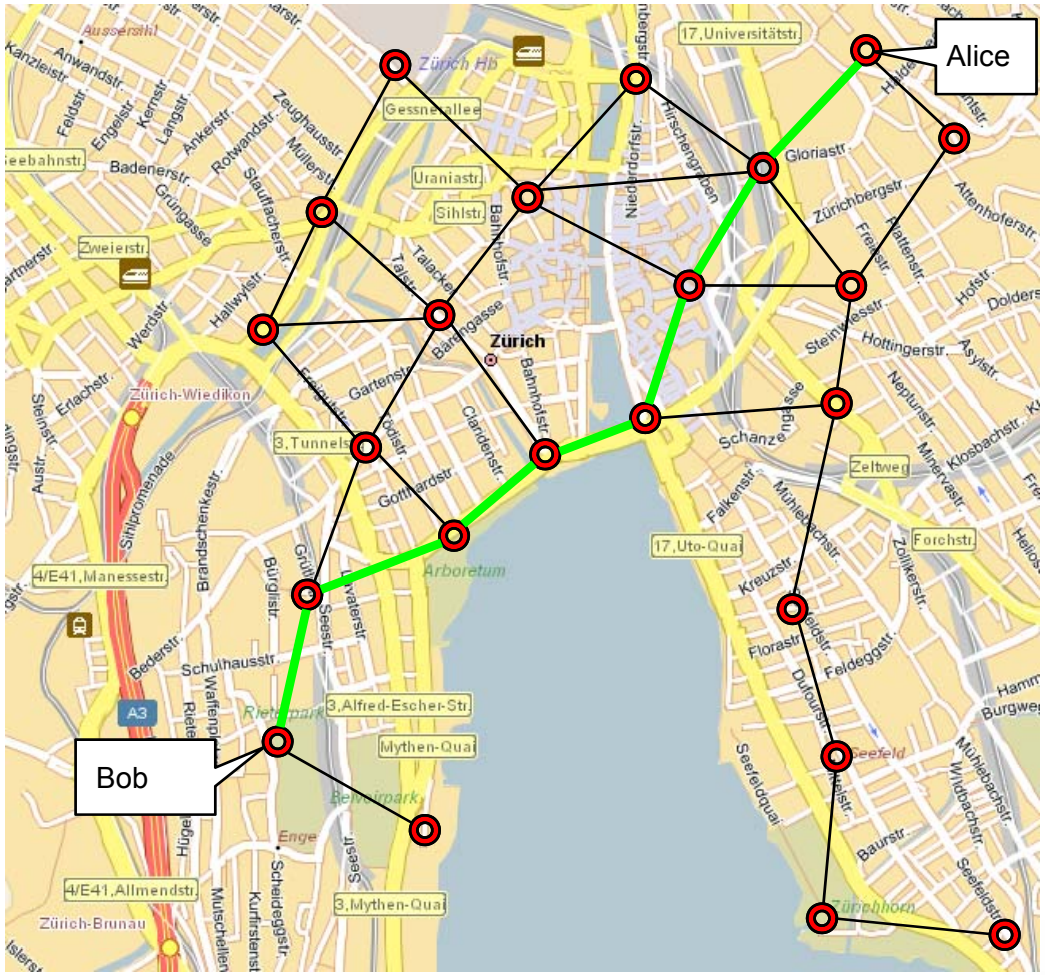



Geo-Routing

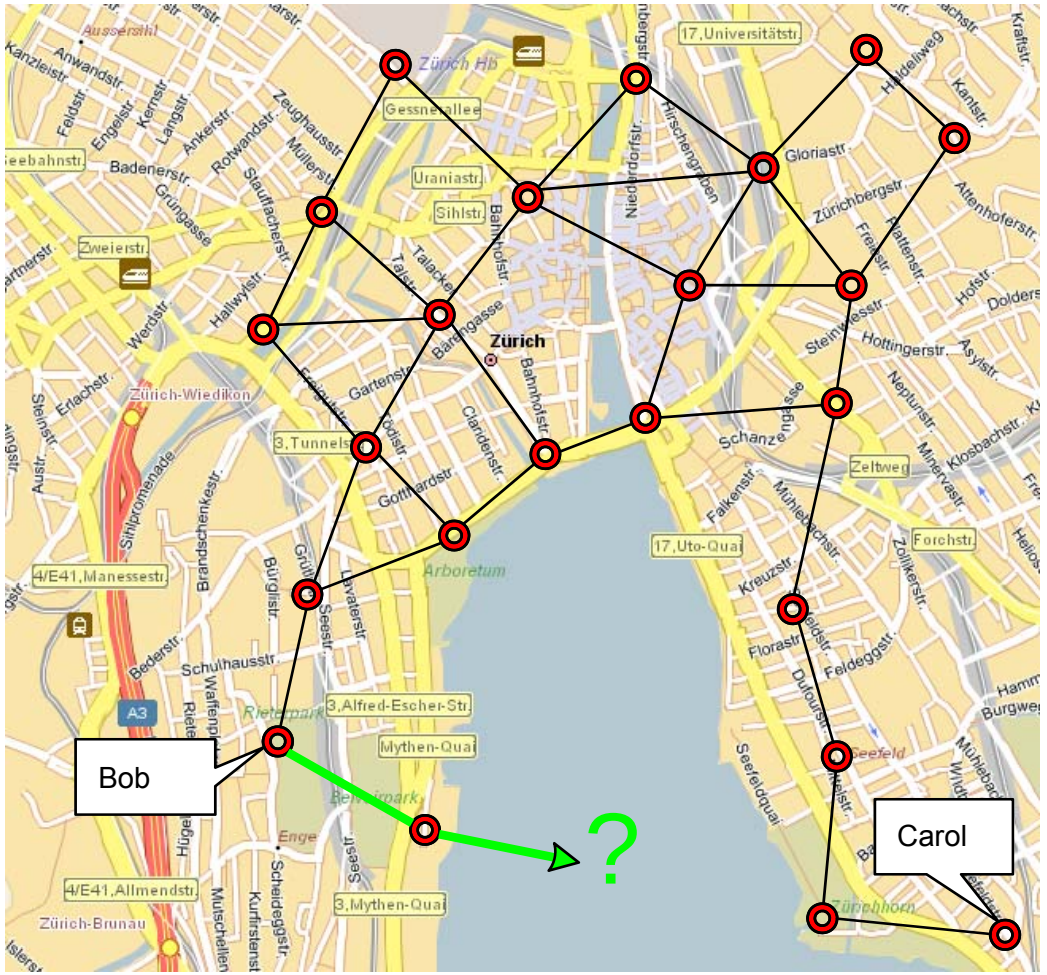
Geo-Routing


???
Alice
Bob
17,Universitätstr.
Gloriastr.
4/E41,Manessestr.
A3
4/E41,Allmendstr.
Zürich-Brunau

Greedy Geo-Routing?



Greedy Geo-Routing?

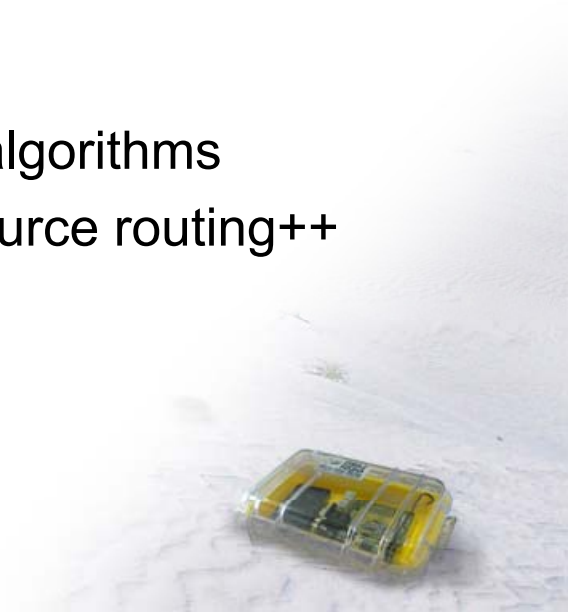


What is Geographic Routing?

- A.k.a. geometric, location-based, position-based, etc.

- Each node knows its own position and position of neighbors
- Source knows the position of the destination
- No routing tables stored in nodes!

- Geographic routing makes sense
  - Own position: GPS/Galileo, local positioning algorithms
  - Destination: Geocasting, location services, source routing++
  - Learn about ad-hoc routing in general

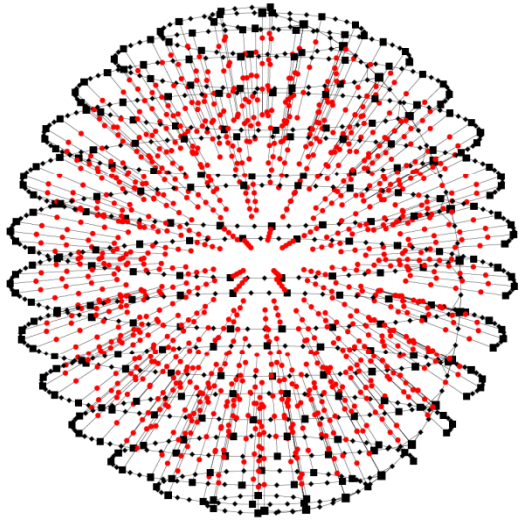


Geo-Routing Results

- Can be done ("face routing")
  - [Kranakis, Singh, Urrutia, CCCG 1999]
  - [Bose, Morin, Stojmenovic, Urrutia, DIALM 1999]
  - later: others… "GPSR"

- At what cost?
  - Geo-routing cost (hops) is quadratic to optimal route
    [Kuhn, W, Zollinger, DIALM 2002]

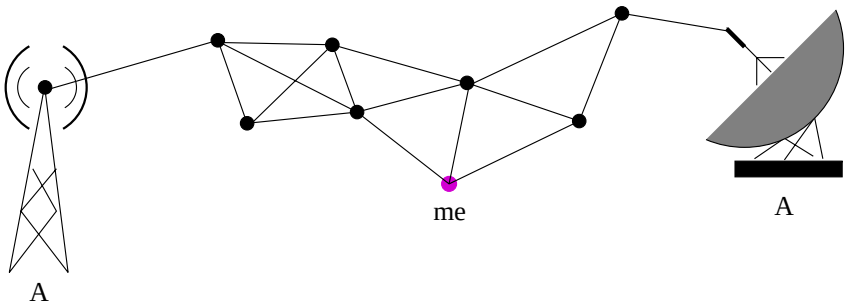
- Can it be done in 3D?!?
  - Does a technique like face routing exist for 3D?
  - No! There is no deterministc 3D geo-routing algo
    [Durocher, Kirkpatrick, Naranyanan, ICDCN 2008]
  - … unless you use randomization
    [Flury, W, Infocom 2008]




Positioning
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Positioning

- Why positioning?
  - Sensor nodes without position information is often meaningless
  - Geo-routing



- Why not GPS (or Galileo)?
  - ~~Heavy, large, and expensive~~
  - ~~Battery drain~~
  - Not indoors
  - Accuracy?

- Idea: equip small fraction with GPS (anchors)

Is Multi-Hop Positioning Possible…?

- … let's assume to have perfect hardware?
  - we can measure distances between nodes perfectly
  - we can measure relative angles between nodes perfectly

- No!
  - NP-hard: [Breu, Kirkpatrick, CG 1998]
  - … even if we have exact distance information [Aspnes, Goldberg, Yang, Algosensors 2004]
  - … even if we have exact angle information [Bruck, Gao, Jiang, Mobihoc 2004]
  - APX-hard: [Kuhn, Moscribroda, W, DIALM 2004]

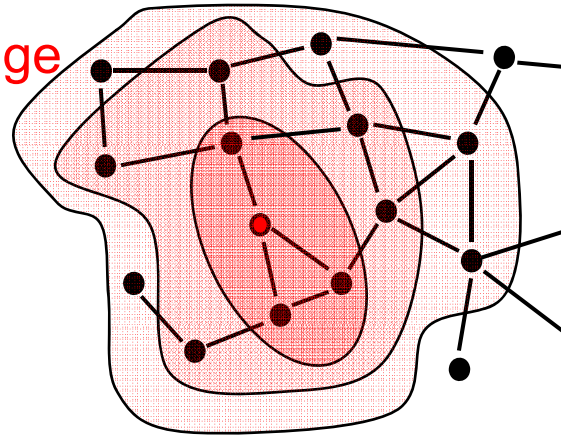
  - Best algorithm: $O(\log^{2.5} n)$ approximation [Pemmaraju, Pirwani, ESA 2006]

Local Algorithms

Global Optimization with Local Information?

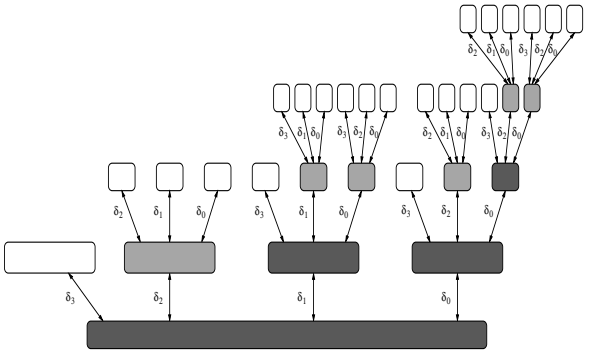
- Towards a theory for understanding large-scale networks/systems

- Nodes in network/system only have local knowledge
  - nodes must make decision based on their local information only



- We proved the first upper and lower bounds for traditional network optimization problems
  - now we have a much better understanding what is (in)feasible
  - basis for understanding self-organization & dynamic systems

- [Linial, SIAM JoC 1992]
- [Kuhn, Moscibroda, W, PODC 2004]
- [Schneider, W, PODC 2008]
- [Lenzen, W, DISC 2008]

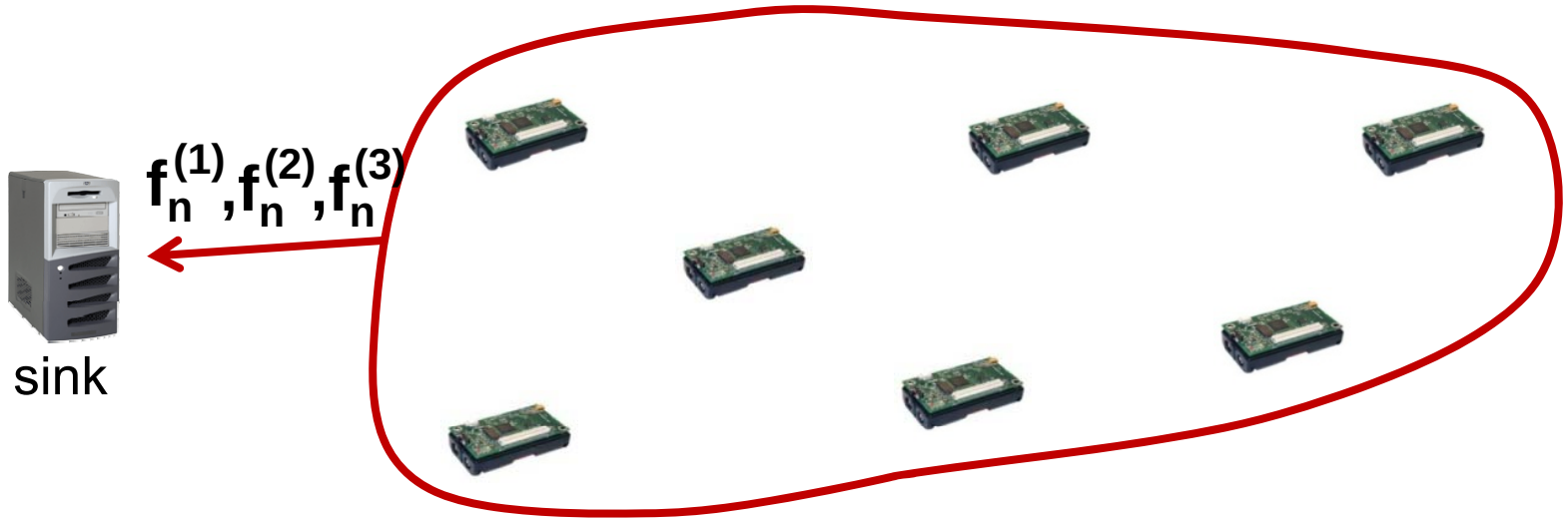



Capacity

Data Gathering in Wireless Sensor Networks

- Data gathering & aggregation
  - Classic application of sensor networks
  - Sensor nodes periodically sense environment
  - Relevant information needs to be transmitted to sink

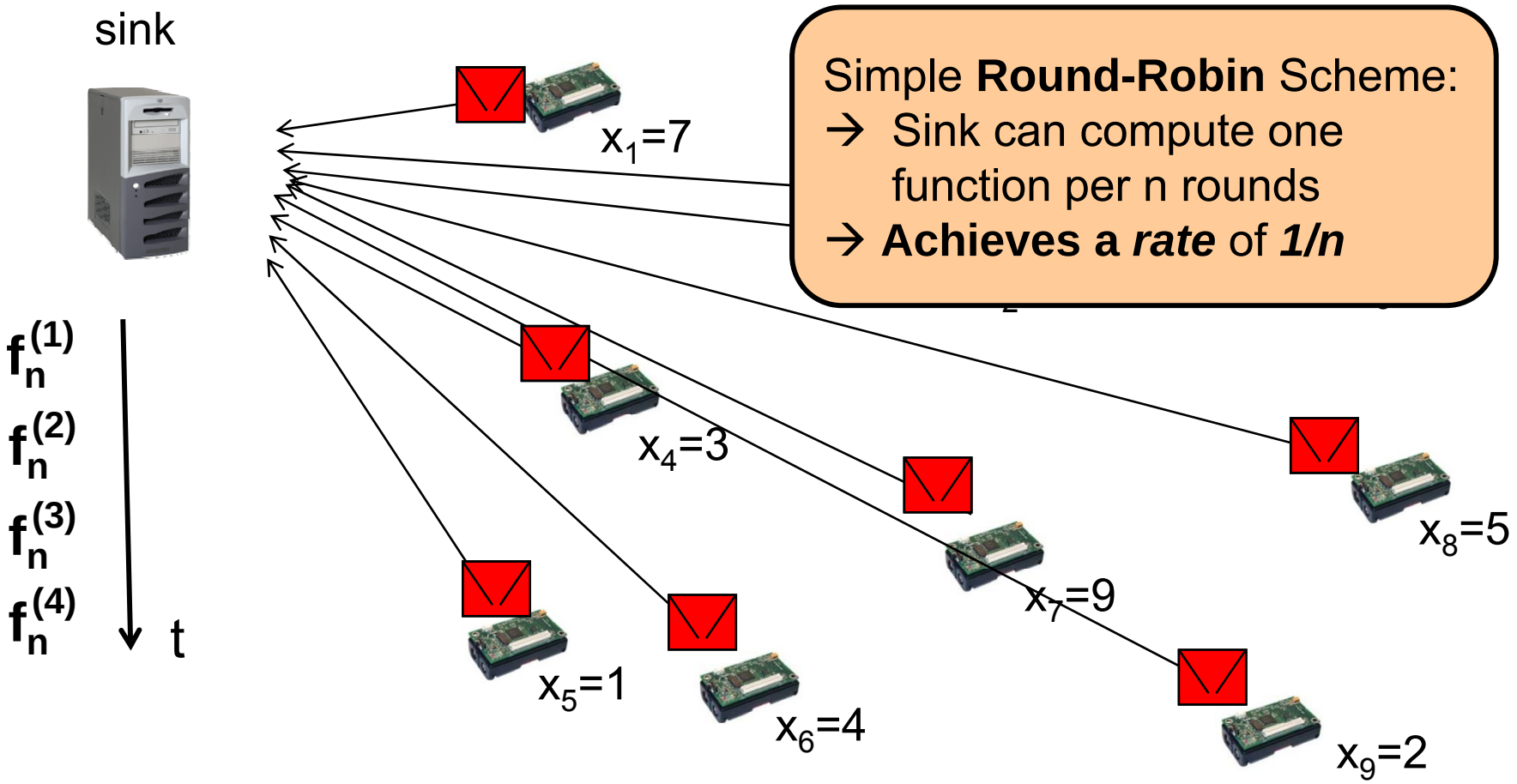
- Functional Capacity of Sensor Networks
  - Sink peridically wants to compute a function f_n of sensor data
  - At what rate can this function be computed?



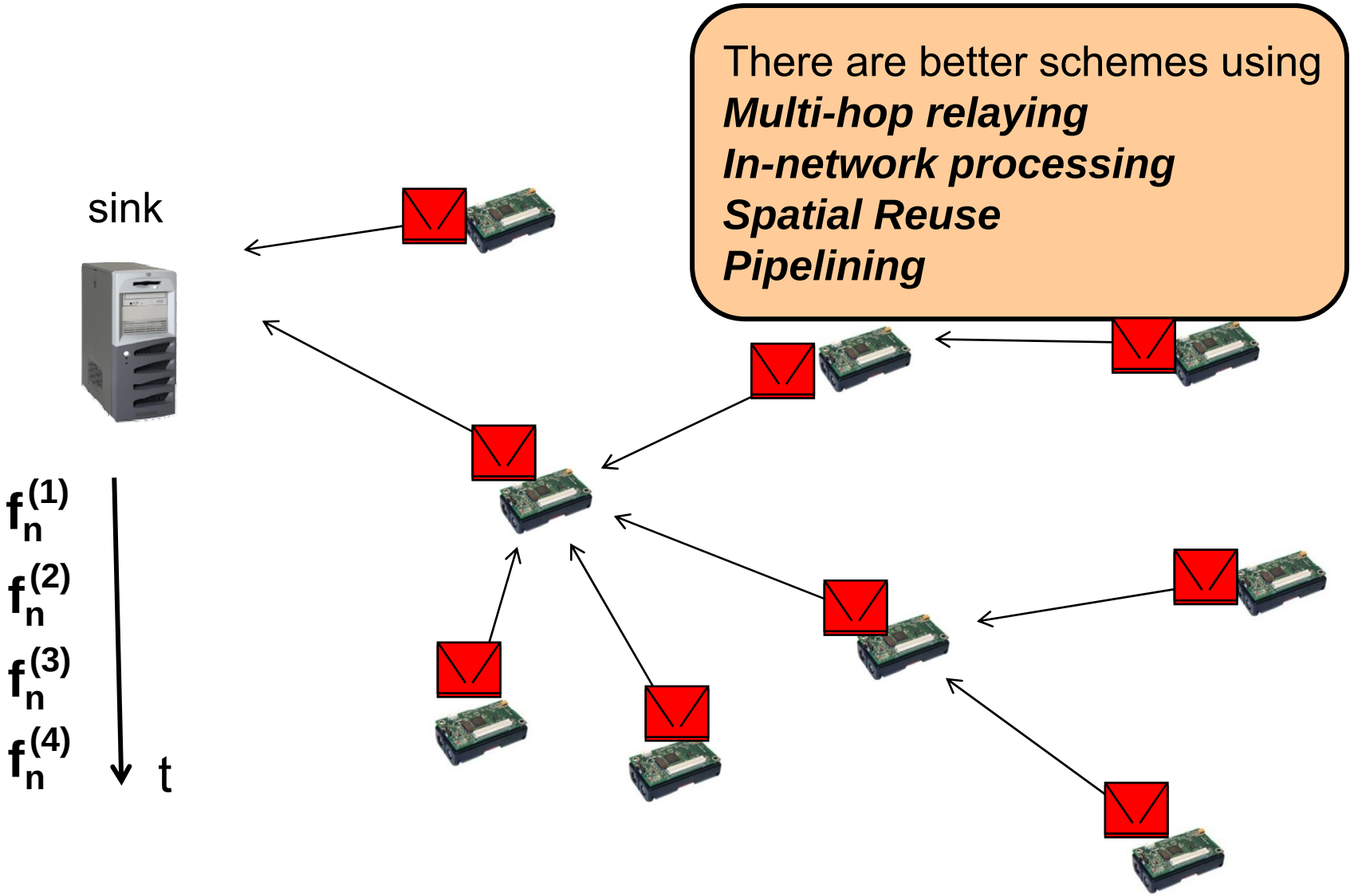
Data Gathering in Wireless Sensor Networks

Example: simple round-robin scheme

→ Each sensor reports its results directly to the root one after another



Data Gathering in Wireless Sensor Networks



Capacity in Wireless Sensor Networks

At what ***rate*** can sensors transmit data to the sink?
Scaling-laws → how does rate decrease as n increases…?

$\Theta(1/n)$ $\Theta(1/\sqrt{n})$ $\Theta(1/\log n)$ $\Theta(1)$

Answer depends on:
- Function to be computed
- Coding techniques
- Network topology

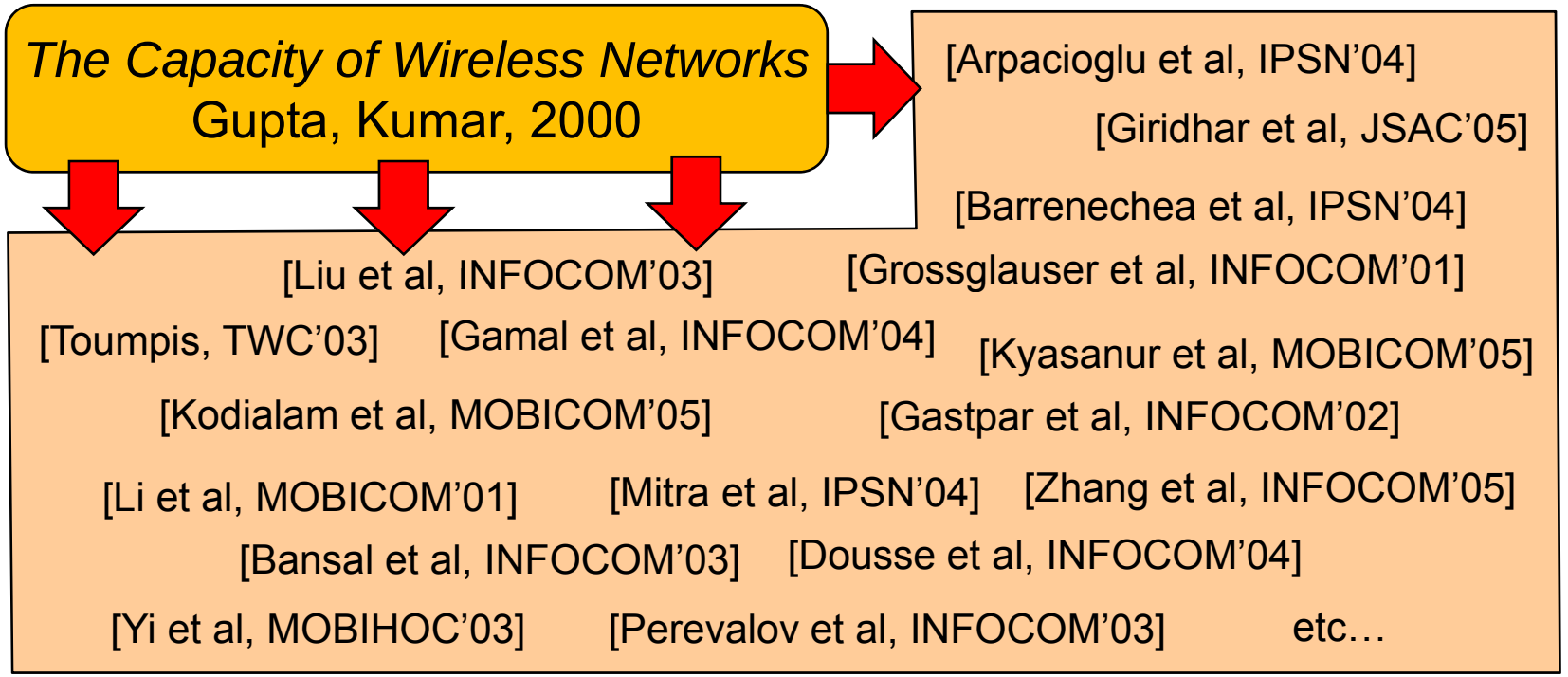
→ Only perfectly compressible functions (max, min, avg,…)

→ No fancy coding techniques

Practical relevance?

- Efficient data gathering!
- Efficient MAC layer!

- This (and related) problem is also studied theoretically:

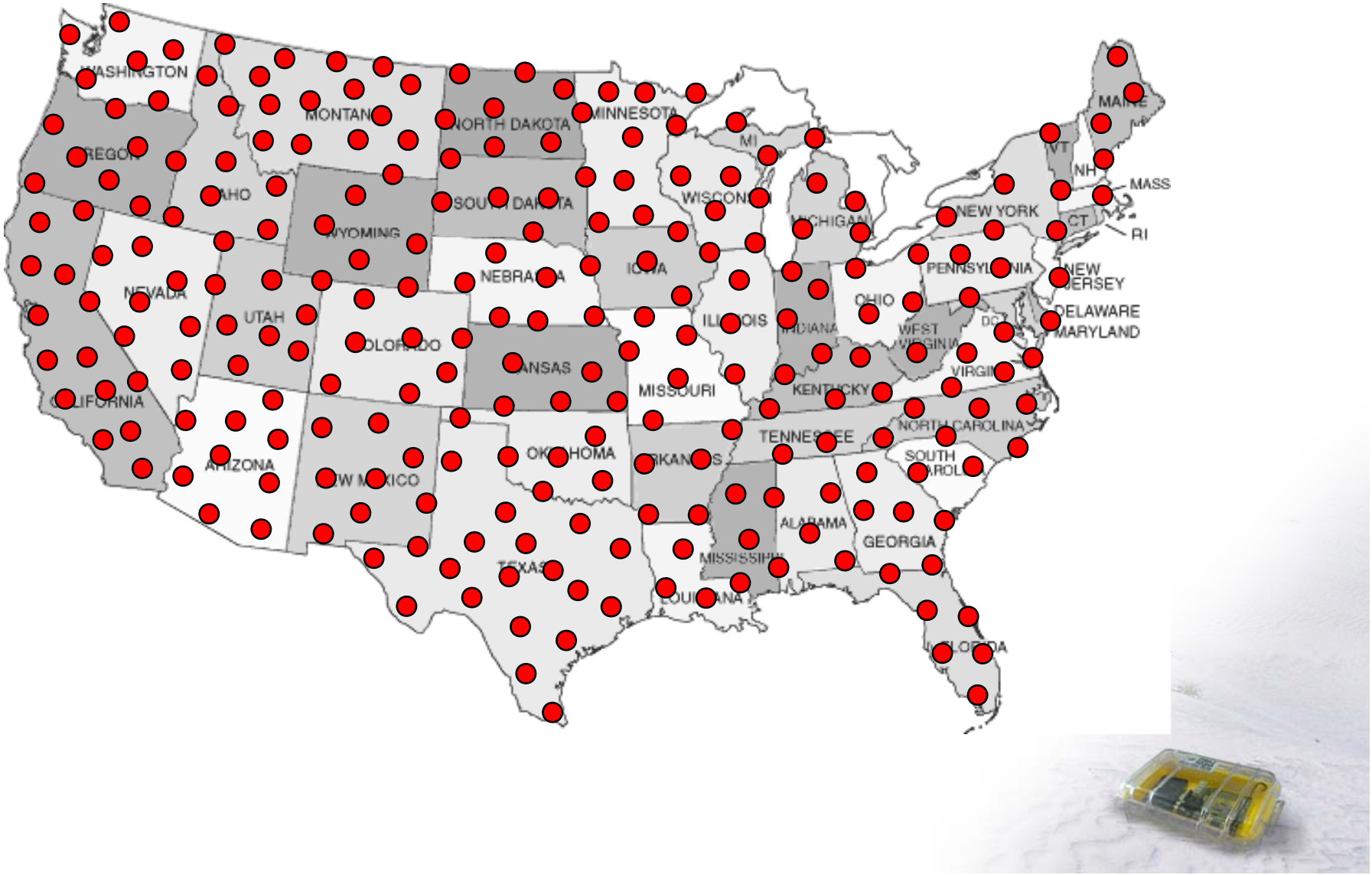


Worst-Case Capacity

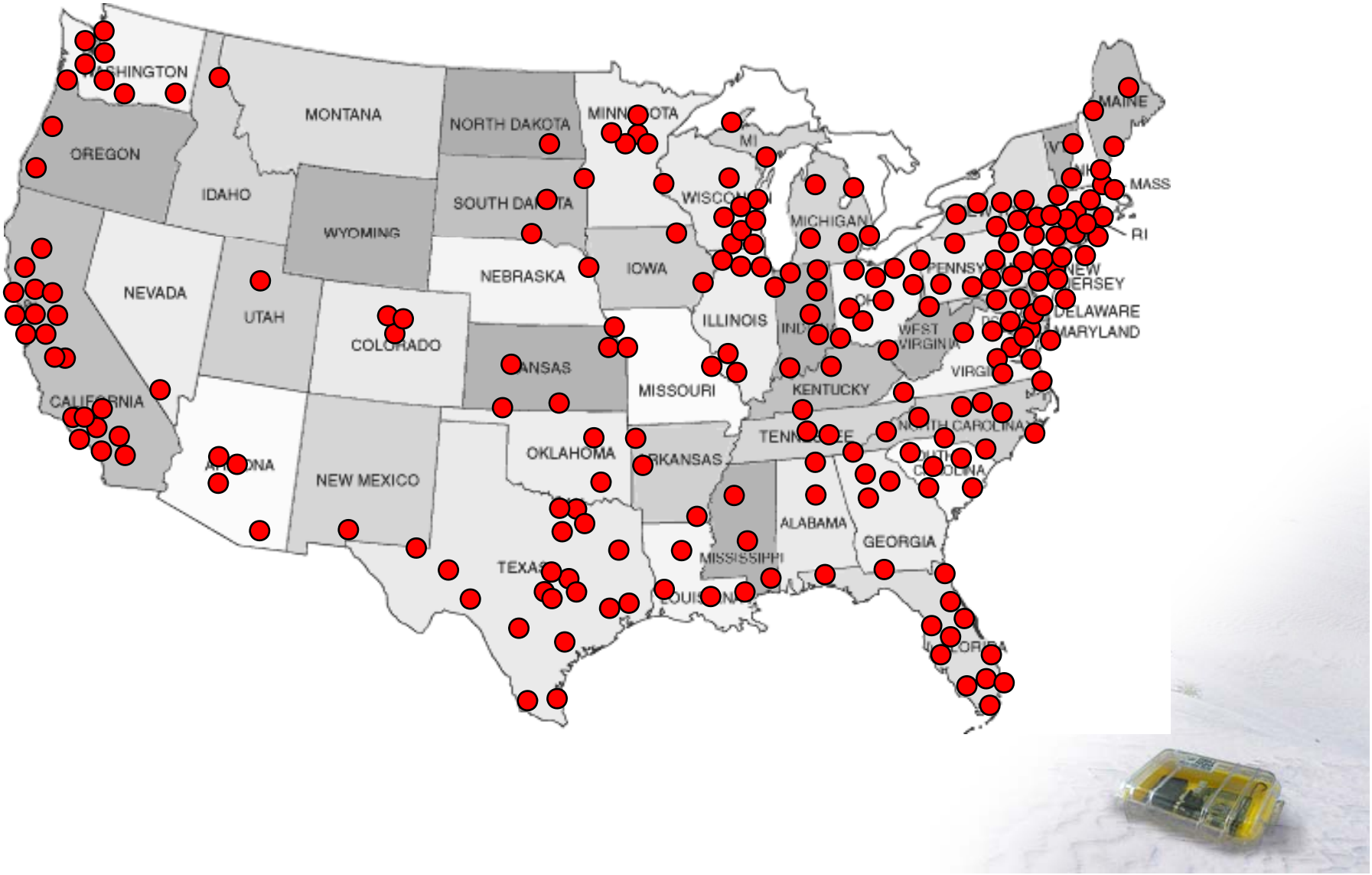
- Capacity studies so far make very strong assumptions on node deployment, topologies
  - randomly, uniformly distributed nodes
  - nodes placed on a grid
  - etc...

What if a network looks differently…?

Like this?


MONTANA
NORTH DAKOTA
MINNESOTA
WISCONSIN
MICHIGAN
NEW YORK
MAINE
VT
NH
MASS
CT
RI
NEW JERSEY
DELAWARE
MARYLAND
PENNSYLVANIA
OHIO
INDIANA
WEST VIRGINIA
KENTUCKY
TENNESSEE
NORTH CAROLINA
SOUTH CAROLINA
GEORGIA
ALABAMA
MISSISSIPPI
LOUISIANA
TEXAS
OKLAHOMA
KANSAS
MISSOURI
NEBRASKA
COLORADO
WYOMING
UTAH
NEVADA
ARIZONA
NEW MEXICO
CALIFORNIA
OREGON
WASHINGTON

Or rather like this?



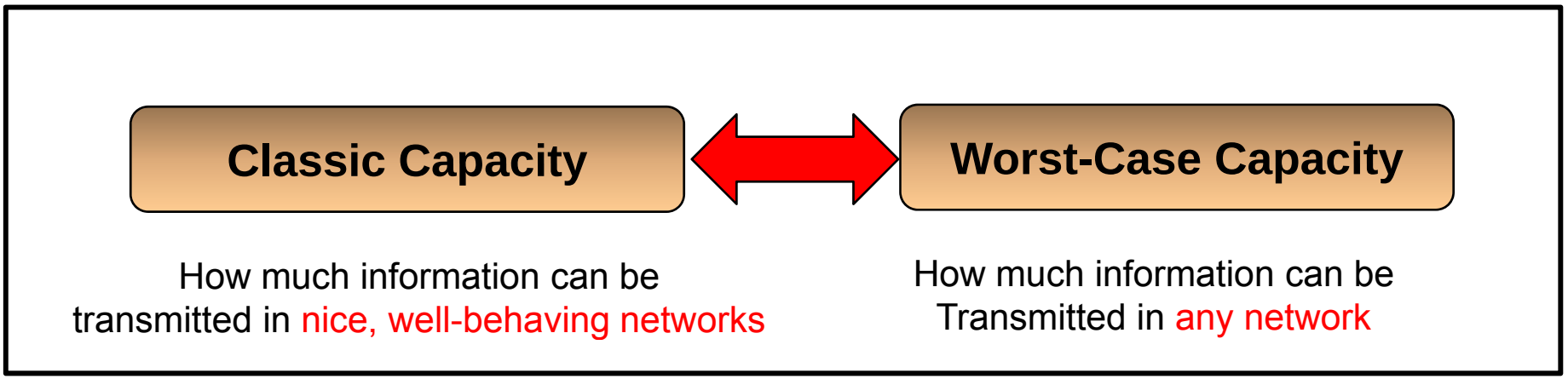
Worst-Case Capacity

- Capacity studies so far have made very strong assumptions on node deployment, topologies
  - randomly, uniformly distributed nodes
  - nodes placed on a grid
  - etc...

What if a network looks differently…?

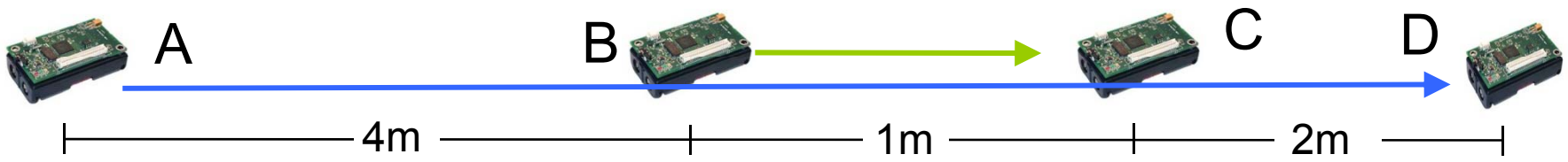
We assume arbitrary node distribution

worst-case topologies

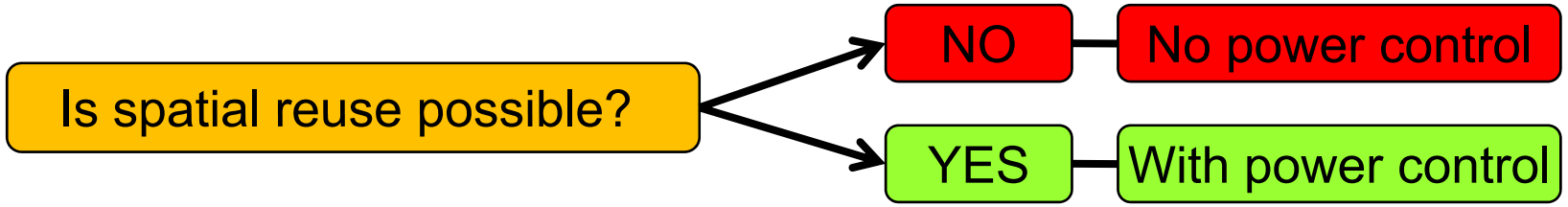


Example: Protocol vs. Physical Model

A sends to D, B sends to C



Assume a single frequency (and no fancy decoding techniques!)



Let α=3, β=3, and N=10nW

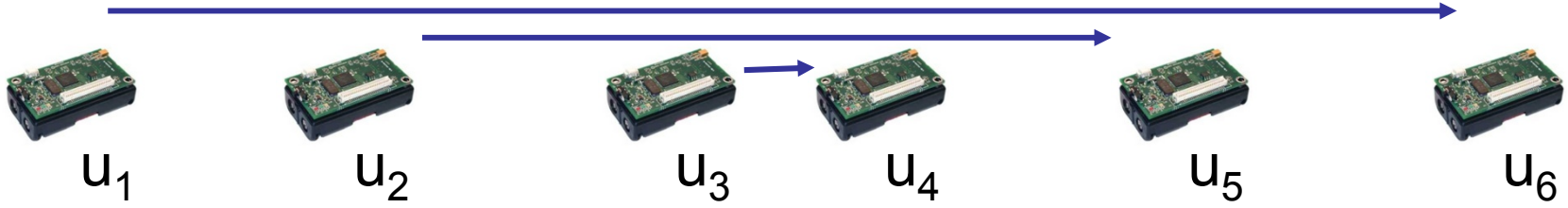
Transmission powers: P_B= -15 dBm and P_A= 1 dBm

SINR of A at D: $\frac{1.26mW/(7m)^3}{0.01\mu W + 31.6\mu W/(3m)^3} \approx 3.11 \geq \beta$

SINR of B at C: $\frac{31.6\mu W/(1m)^3}{0.01\mu W + 1.26mW/(5m)^3} \approx 3.13 \geq \beta$

This works in practice!

- We did measurements using standard mica2 nodes!
- Replaced standard MAC protocol by a (tailor-made) „SINR-MAC“
- Measured for instance the following deployment...



- Time for successfully transmitting 20‘000 packets:

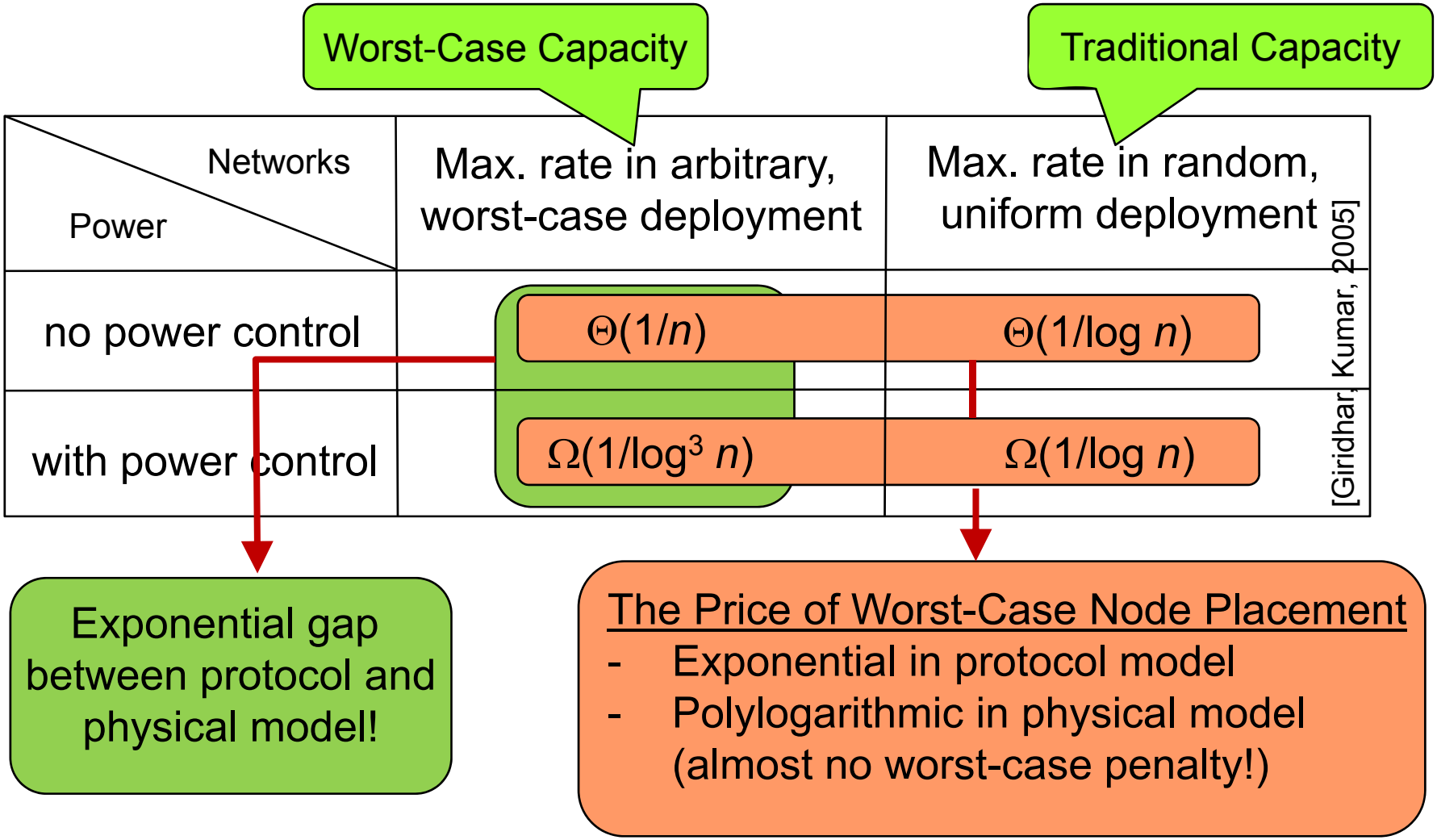
	Time required	
	standard MAC	“SINR-MAC”
Node u_1	$721s$	$267s$
Node u_2	$778s$	$268s$
Node u_3	$780s$	$270s$

	Messages received	
	standard MAC	“SINR-MAC”
Node u_4	19999	19773
Node u_5	18784	18488
Node u_6	16519	19498

Speed-up is almost a factor 3

[Moscibroda, W, Weber, Hotnets’06]

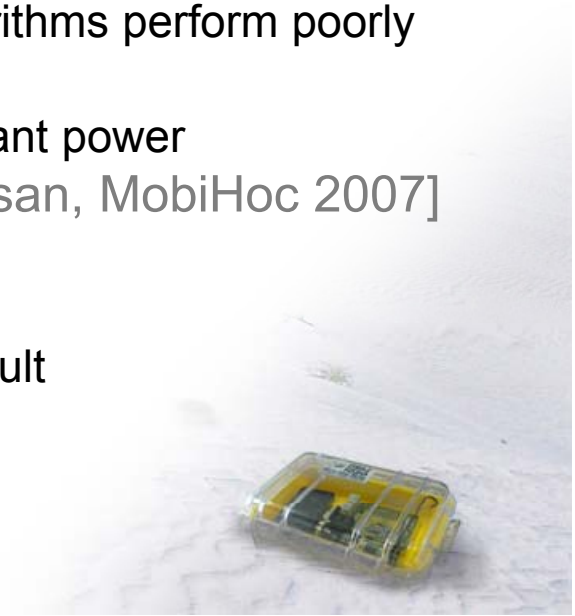
Worst-Case Capacity in Wireless Networks



Overview of results so far

- [Moscibroda, W, Infocom 2006]
  - First paper in this area, $O(\log^3 n)$ bound for connectivity, and more
  - This is essentially the paper I presented on the previous slides

- [Moscibroda, W, Zollinger, MobiHoc 2006]
  - First results beyond connectivity, namely in the topology control domain
- [Moscibroda, W, Weber, HotNets 2006]
  - Practical experiments, ideas for capacity-improving protocol
- [Moscibroda, Oswald, W, Infocom 2007]
  - Generalizion of Infocom 2006, proof that known algorithms perform poorly
- [Goussevskaia, Oswald, W, MobiHoc 2007]
  - Hardness results & constant approximation for constant power
- [Chafekar, Kumar, Marathe, Parthasarathy, Srinivasan, MobiHoc 2007]
  - Cross layer analysis for scheduling and routing
- [Moscibroda, IPSN 2007]
  - Connection to data gathering, improved $O(\log^2 n)$ result
- [Locher, von Rickenbach, W, ICDCN 2008]
  - Still some major open problems



Summary

- Lower Bounds and Impossibility Results
  - Clock Synchronization
  - Distributed Selection / Median
  - Geo-Routing
  - Positioning
  - Local Algorithms
  - Capacity

Theory for sensor networks, what is it good for?!

How many lines of pseudo code //
Can you implement on a sensor node?


[Ali G]

The best algorithm is often complex //
And will not do what one expects.

Theory models made lots of progress //
Reality, however, they still don't address.

My advice: invest your research £££s //
in ... impossibility results and lower bounds!


Thank You!
Questions & Comments?
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich