# Corso di Reti Mobili

# Reti Ad Hoc e Reti di Sensori

**Paolo Santi**

Istituto di Informatica e Telematica del CNR, Pisa, ITALY

*paolo.santi@iit.cnr.it*

Istituto di Informatica e Telematica

# Topology Control

## in Wireless Ad Hoc and Sensor Networks

Istituto di Informatica e Telematica

# Summary of TC

- Introduction
  - Motivation: the need for Topology Control (TC) in ad hoc and sensor networks
  - A network model: radio signal propagation, energy consumption, and interference
  - An informal definition of TC
  - Topology Control: a taxonomy
  - TC in the protocol stack
- The Critical Transmitting Range for connectivity
- Topology Optimization Problems:
  - the Range Assignment Problem
  - Energy efficient topologies for unicast/broadcast

# Summary of TC (2)

- Distributed Topology control
  - "Ideal" properties of a distributed TC protocol
  - Examples of distributed TC protocols: location-based, direction-based, neighborhood based
- Dealing with node mobility
- Towards and implementation of TC: level-based TC

# Motivations for topology control

- Energy and capacity are limited resources in ad hoc/sensor networks
- In case of sensor networks, energy consumption is especially critical
- The network designer should strive for reducing node energy consumption and providing sufficient network capacity
- *The answer*: Topology Control (TC) — maintain a topology with certain properties (e.g., connectivity) while *reducing* energy consumption and/or *increasing* network capacity

# TC and energy consumption

- Wireless channel model: (no interference)
  - $P_i$: power used by node $i$ to send the message
  - $P_{r,j}$: intensity of the received signal at node $j$
  - Node $j$ can correctly receive the message sent by $i$ if

    $$P_{r,j} \geq \beta ,$$

    where $\beta$ is a threshold value which depends on the requested communication quality
  - $P_{r,j}$ is determined by the path loss between nodes $i$ and $j$

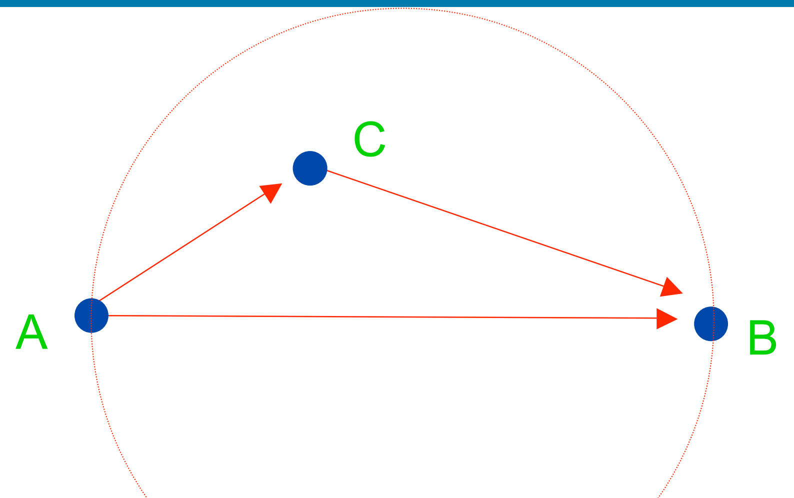
    $$P_{r,j} = P_i / PL(i,j)$$

  - Typical assumption:

    $$PL(i,j) \propto \text{dist}(i,j)^{\alpha} ,$$

    where $\alpha$ is in the range [2,6] (depending on environmental conditions)

# TC and energy consumption (2)




A wants to send a packet to B

For energy efficiency, is it better to use the long link AB, or two shorter links AC-CB?

$P_{XY}$ = min power needed to send a packet from X to Y

One long link: $P_{AB} = dist(A,B)^{\alpha}$

Two short links: $P_{AC} + P_{CB} = dist(A,C)^{\alpha} + dist(C,B)^{\alpha}$

Example ($\alpha = 2$): $dist(A,B)^2 = dist(A,C)^2 + dist(C,B)^2 - \cos(ACB)$

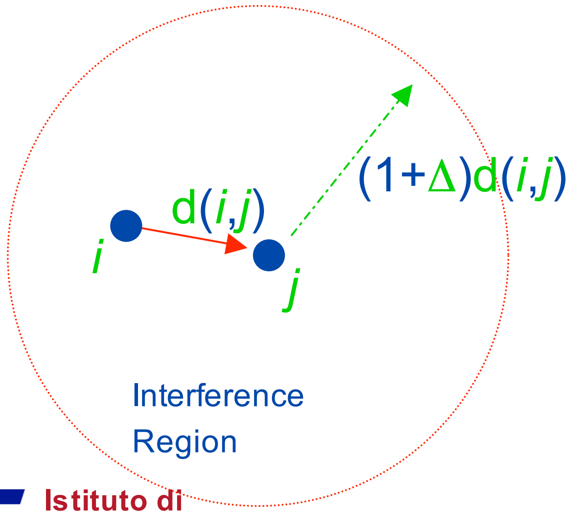
Conclusion: two short links are preferable whenever C is in the dashed circle

# TC and network capacity

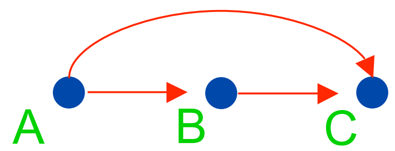
- Protocol Interference model:
  - *Assumption*: all the nodes use the same transmit power
  - A packet sent by node *i* is correctly received by node *j* (within range) if and only if

$$\mathrm{dist}(j,w) \geq (1+\Delta)\,\mathrm{dist}(i,j)\,,$$

for any other node *w* that is transmitting simultaneously, where $\Delta$ is a constant that depends on the features of the radio



# TC and network capacity (2)




A wants to send a packet to C

For network capacity, is it better to use the long link AC, or two shorter links AB-BC?

Compare the size of the Interference Regions

One long link: $\pi$ dist(A,C)$^2$ $(1+\Delta)^2$

Two short links: $\pi$ dist(A,B)$^2$ $(1+\Delta)^2$ + $\pi$ dist(B,C)$^2$ $(1+\Delta)^2$ ,

where dist(A,C) = dist(A,B) + dist(B,C)

Conclusion: (by Holder inequality) two short links are preferable

# Topology control: an informal definition

**Topology control:**

*the art of coordinating nodes' decisions regarding their transmitting ranges, in order to generate a network with the desired properties*

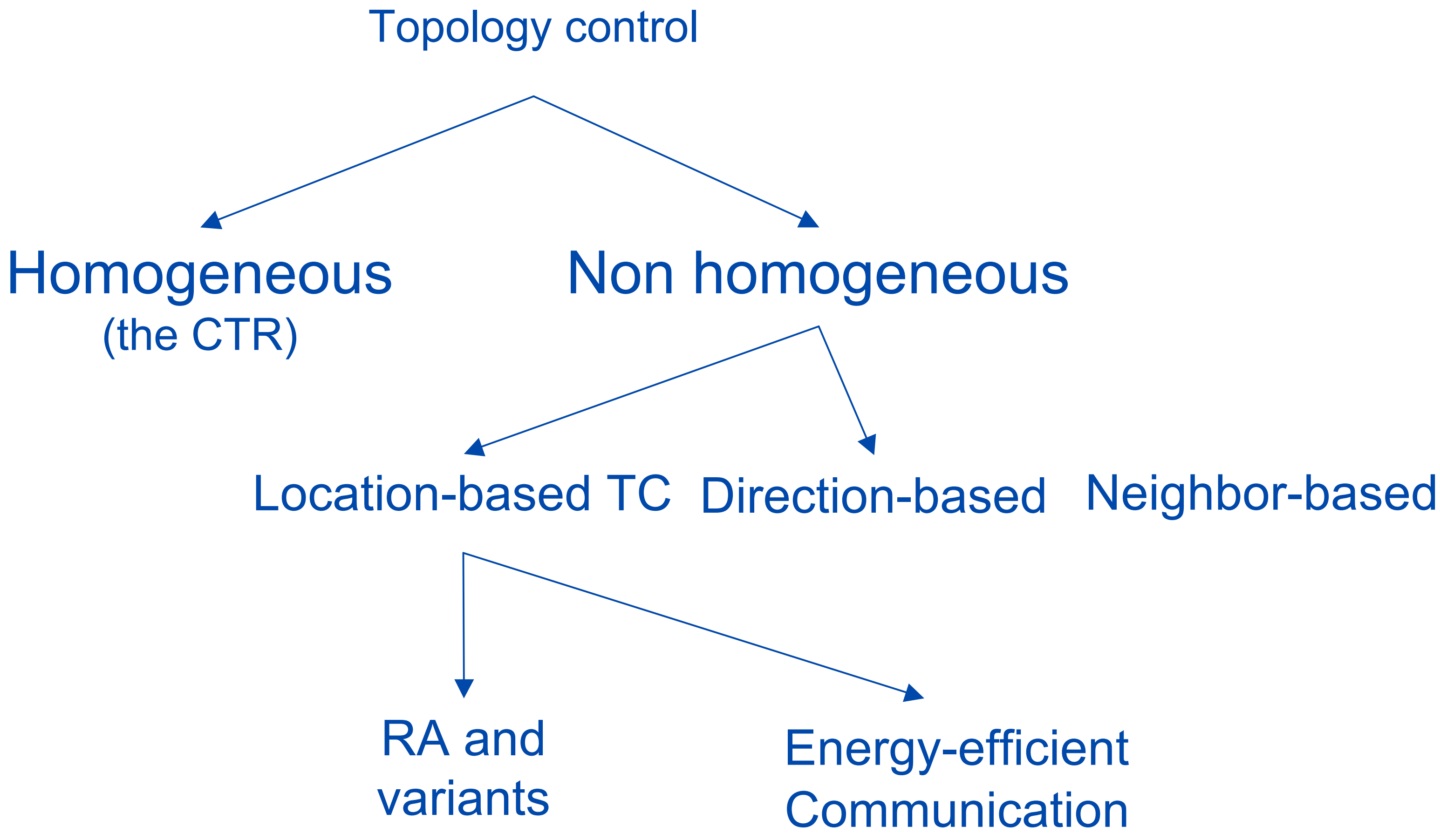
Other forms of "topology control":

- *Clustering techniques*: a way of "organizing" the network topology

Differences between TC and clustering:

- In clustering, nodes do not change the transmit power; instead, nodes are assigned with different roles in the network
- In TC, nodes change their transmit power dynamically; all the nodes have the same role

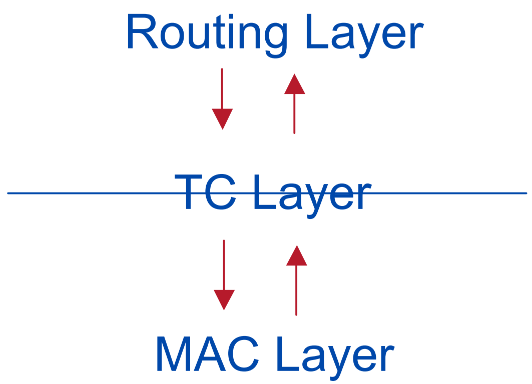
# Topology control: a taxonomy



# TC in the protocol stack

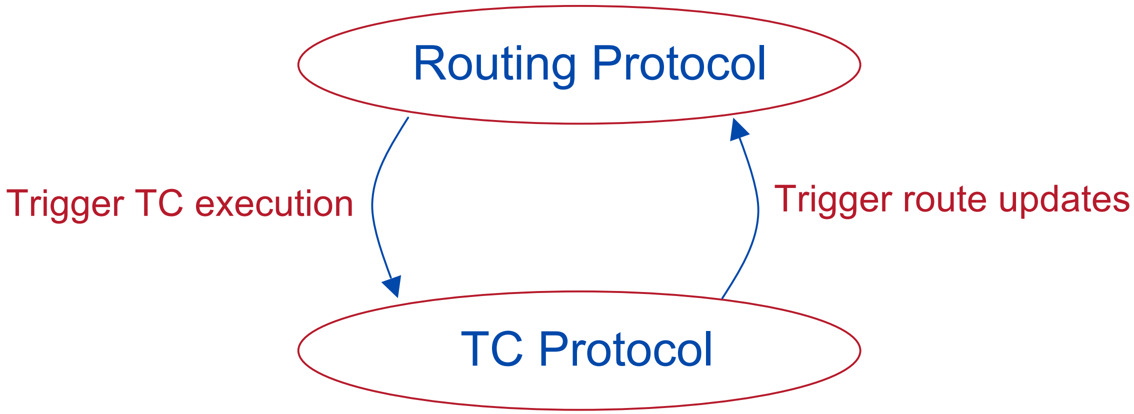
- Where should TC be positioned in the protocol stack?
- No clear answer in the literature

One possible solution:



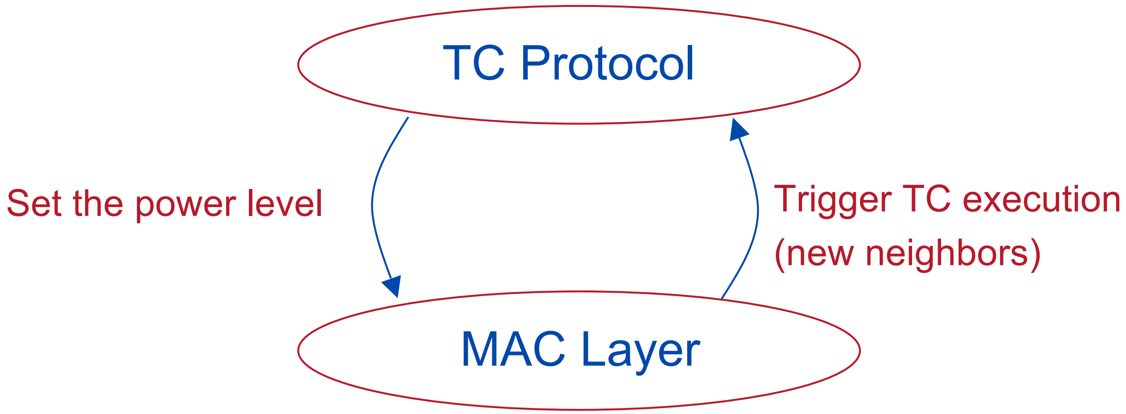
# TC and Routing

One possible view:



# TC and MAC

One possible view:



# TC and MAC (2)

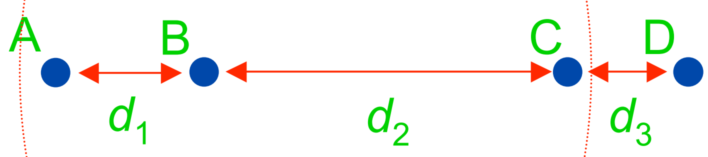
Using different transmit powers can:

- Introduce additional opportunities for collisions in some cases (BAD)

as well as

- Reduce interference (i.e., increase network capacity) in other cases (GOOD)

# TC and MAC: Example




$d_1 < d_2$ and $d_3 < d_2$

*Hp*:802.11 MAC with RTS/CTS exchange

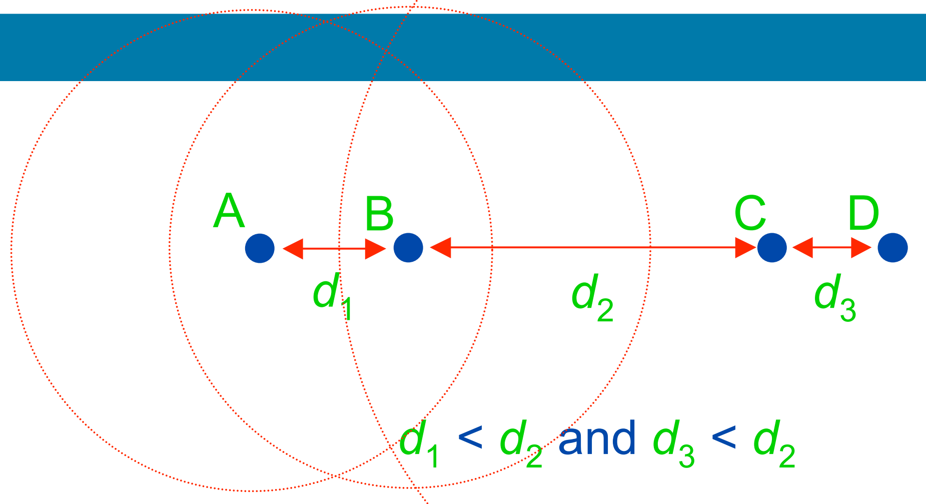
A wants to send a packet to B, and C wants to send a packet to D

No transmit power control:

all the nodes have the same range $r$, with $r > d_2 + \max\{d_1, d_3\}$

A and C are within each other transmit range: the first that accesses the channel transmits, the other must wait

# TC and MAC: Example(2)




*Hp*:802.11 MAC with RTS/CTS exchange

A wants to send a packet to B, and C wants to send a packet to D

Bad tx power control:

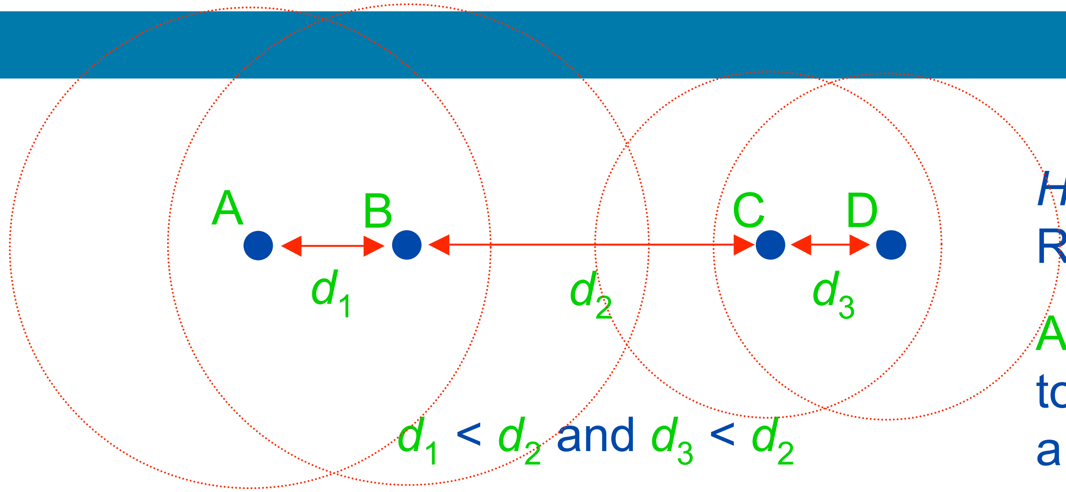
A and B have tx range $r_1$ with $d_1 < r_1 < d_2$

C and D have tx range $r_2$ with $r_2 > d_2$

C cannot hear RTS/CTS exchange between A and B; C starts its own transmission, causing a collision at node B

Istituto di Informatica e Telematica

# TC and MAC: Example(3)




*Hp*:802.11 MAC with RTS/CTS exchange

A wants to send a packet to B, and C wants to send a packet to D

Good tx power control:

A and B have tx range $r_1$ with $d_1 < r_1 < d_2$

C and D have tx range $r_3$ with $d_3 < r_3 < d_2$

Transmissions A → B and C → D can occur simultaneously without interference; network capacity is doubled!

# The Critical Transmitting Range

# The Critical Transmitting Range (CTR)

Assumption: all the nodes have the same transmitting range *r*

**The CTR problem:**

Assume *n* nodes are placed in a given region *R*; what is the minimum value of *r* such that the resulting network is (strongly) connected?

- This minimum value of *r* is called the *critical transmitting range* for connectivity

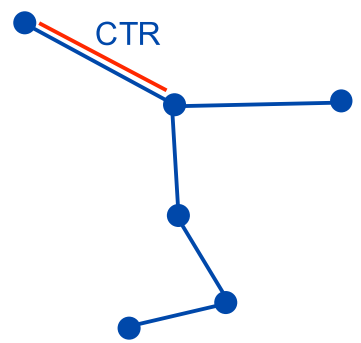
# CTR: motivations

- Why studying the CTR problem is important?
  - In many situations, dynamically adjusting the node transmitting range is not feasible – for instance, because the wireless transceiver does not allow the transmit power to be adjusted

- Characterizing the CTR helps the network designer to answer fundamental questions, such as:
  - Given *n*, which is the minimum value of the transmitting range that ensures connectivity?
  - Given a transmitter technology (i.e., *r*), how many nodes must be distributed in order to obtain a connected network?

# The longest MST edge

- Solving the CTR problem is easy if node positions are know: the CTR is the longest edge of the Euclidean MST built on the nodes




Istituto di Informatica e Telematica

# CTR: probabilistic approaches

- In many realistic scenarios, node positions *are not* known in advance (for instance, sensors spread from a moving vehicle)

- *Probabilistic approaches*: nodes are distributed in *R* according to some distribution; which is the value of *r* which guarantees connectivity with high probability (w.h.p.)?

**Remark:** In this context, w.h.p. means that the probability of connectivity converges to 1 as *n* grows to infinity

# Geometric Random Graphs

- **GRG:** a set of $n$ points are distributed in a $d$-dimensional region $R$ according to some distribution, and some property of the resulting node placement is investigated

  *Example:*

  - length of the longest nearest neighbor link
  - length of the longest MST edge (CTR)
  - total cost of the MST

- These results have been used to prove the following result:
  - If nodes are distributed uniformly at random in $[0,1]^2$, the CTR for connectivity w.h.p. is $r = \sqrt{\frac{\log n}{n}}$

# Critical ranges

- The following result holds for one-dimensional networks (line):

$$r = \log n / n$$

- The following result holds for three-dimensional networks (cube):

$$r = \sqrt[3]{\frac{\log n - \log\log n}{\pi n} + \frac{3}{2} \cdot \frac{1.41 + g(n)}{\pi n}}$$

where $g(n)$ is an arbitrary function which grows to infinity with $n$

# CTR: more practical results

- Besides analytical characterization, the CTR for connectivity has been estimated through simulation

| *n* | CTR |
|---|---|
| 10 | 0,6566 |
| 25 | 0,4415 |
| 50 | 0,3258 |
| 75 | 0,2720 |
| 100 | 0,2353 |
| 250 | 0,1533 |
| 500 | 0,1082 |
| 750 | 0,0894 |
| 1000 | 0,0765 |

**Table 1.**
Values of the CTR when $n$ nodes are distributed uniformly in $R = [0,1]^2$. Here, the CTR is defined as the minimum transmitting range that generates at least 99% of connected graphs

# The COMPOW protocol

- COMPOW is a distributed protocol that attempts to determine the CTR for connectivity
- Nodes can transmit using a predefined number of power levels (6)
- All the nodes use the same power levels
- Nodes maintain a routing table for each power level, and set as the common transmit power the minimum level such that the corresponding routing table contains all the nodes in the network

# The COMPOW protocol (2)

- Setting the power to this minimum level achieves the three goals of:
  - maximizing network capacity,
  - reducing contention to access the wireless link
  - extending network lifetime

  with respect to the case of no TC

- Drawbacks of the COMPOW protocol:
  - Considerable message overhead
  - Requires global knowledge (routing table)

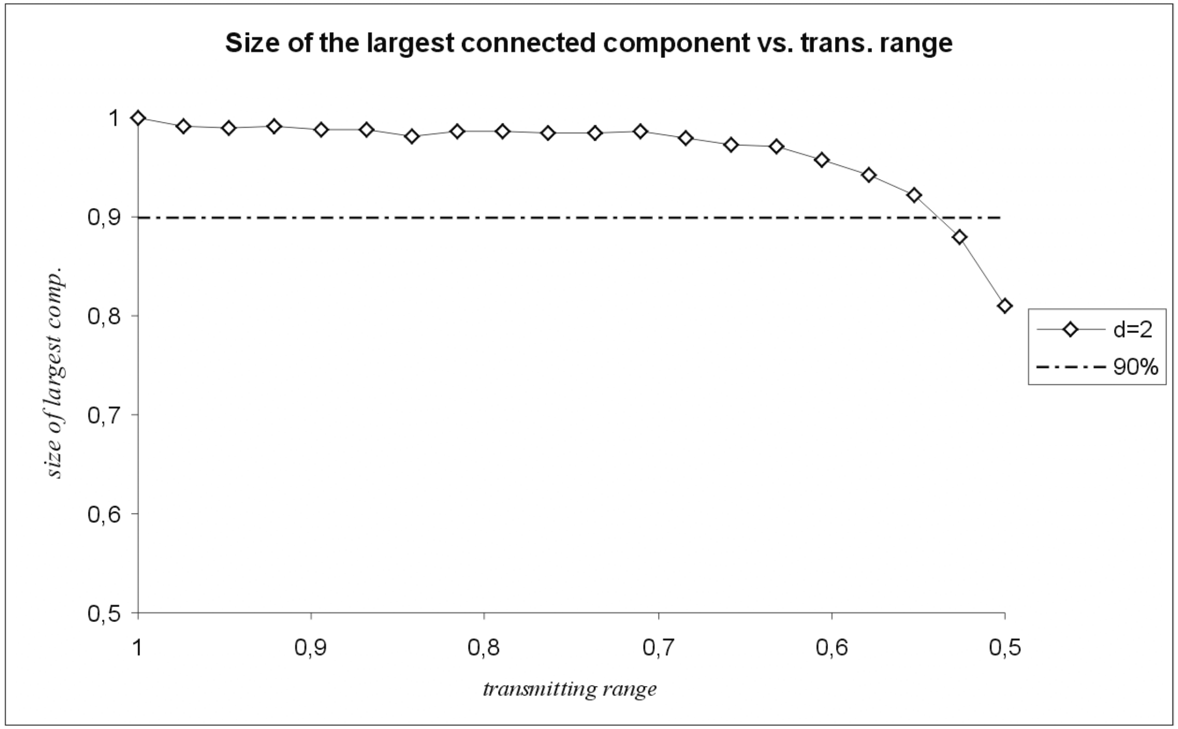
# The giant component

- Suppose all the nodes set their transmit power to 0, and start increasing their power simultaneously
- W.h.p., *connectivity occurs when the last isolated node disappears from the graph*
- In other words, a *giant component* is formed soon, and the remaining increase in the transmit power is needed to connect few isolated nodes

⇓

- Thus, **a lot of power is used to connect relatively few nodes**
- Giant component phenomenon supported by experimental data:
  - reducing the transmitting range of about 40% with respect to CTR yields a graph in which 90% of the nodes are connected

# The giant component (2)




Size of the largest connected component in the communication graph vs. transmitting range (1= CTR). The network is composed by $n$ = 128 nodes

## Corso Reti Mobili

# The Range Assignment Problem

# The communication graph

- Range assignment *RA*: function that assigns a transmit range *RA*(*u*) to each node *u* in the network

- Given node positions and a range assignment *RA*, the *communication graph* contains a directed edge (*u*,*v*) if and only if *v* is within *u*'s transmitting range, i.e. $RA(u) \geq dist(u,v)$

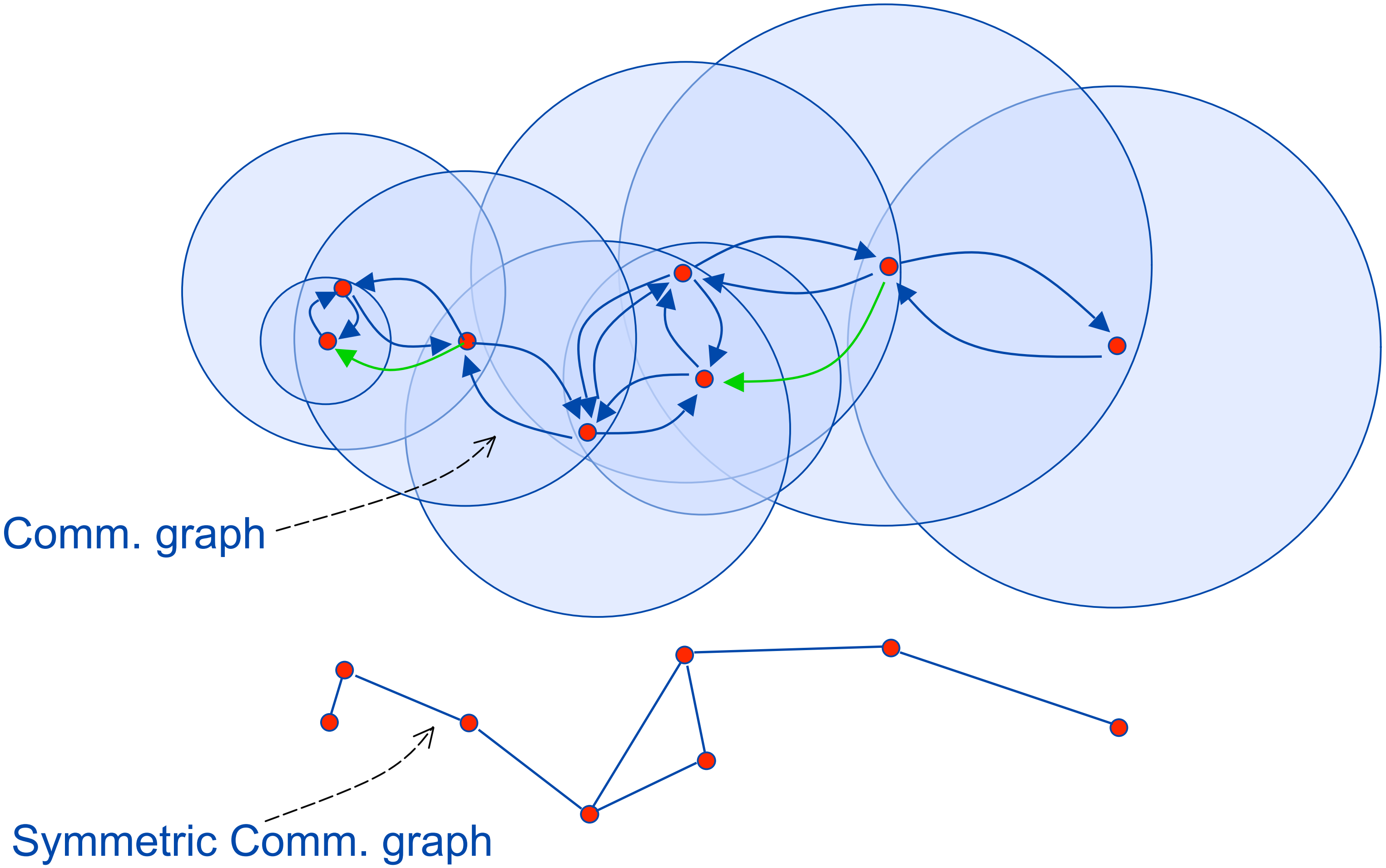
- A range assignment is said to be *connecting* if it generates a strongly connected communication graph

# The symmetric communication graph

- Often, we are only interested in bi-directional (symmetric) links

- The *symmetric communication graph* is obtained from the communication graph by deleting unidirectional wireless links

# An example (Disk Graph)



# The Range Assignment problem

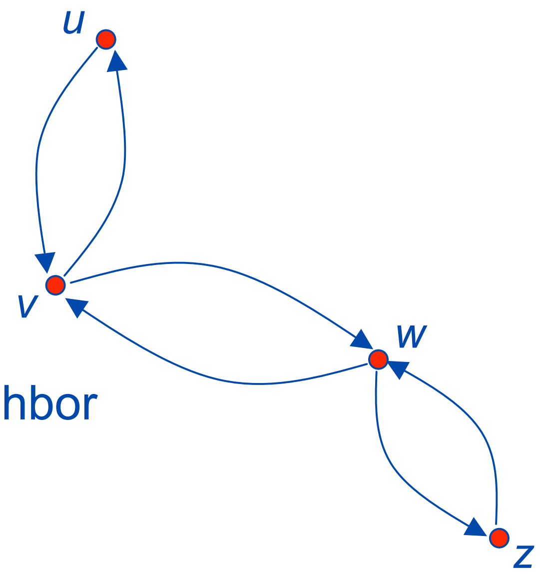
- In the CTR problem, all the nodes have the same transmitting range. What happens in the more general case in which nodes may have different ranges?

- First observation: *unidirectional* links may occur

**The RA problem:**

Consider a set of $n$ points in a $d$-dimensional region $R$, denoting the node positions. Determine a connecting range assignment $RA$ of minimum energy cost, i.e. such that $\sum_u (RA(u))^\alpha$ is minimum

# The Range Assignment problem (2)




*Finding the optimal RA*:

- ✓ Connect each node to the closest neighbor
- ✓ Then what?

In this case is easy:

connect *v* to *w* and *w* to *v*

*But in general?*

# The Range Assignment problem (3)

- The RA problem can be solved in polynomial time if *d* = 1 (nodes along a line), while it is NP-hard if *d* = 2,3
- However, a 2-approximation of the optimal range assignment can be calculated in polynomial time using the MST

# The symmetric RA problem

- The implementation of unidirectional wireless links is "expensive"
- Are unidirectional links really useful?
  - Recent experimental as well as theoretical results seem to say: *no*
- Having a connected backbone of *symmetric* links would ease the integration of TC with existing protocols

# The WSRA problem

**The WSRA problem:**

Consider a set of *n* points in a *d*-dimensional region *R*, denoting the node positions, and let $G_S$ be the symmetric subgraph of the communication graph. Determine a range assignment *RA* such that $G_S$ is connected and the energy cost is minimum

- Solving the WSRA problem remains NP-hard for two and three-dimensional networks

- It has been proven that the additional energy cost necessary to obtain a connected backbone of symmetric edges in the communication graph is asymptotically negligible

# Energy-efficient communication

- Another branch of research focused on computing topologies which have energy-efficient paths between source-destination pairs

- Given a connected communication graph *G*, the problem is to determine a certain subgraph *G'* of *G* (the routing graph) which can be used for routing messages between nodes in an energy-efficient way

- Why use the routing graph *G'* instead of *G*?
  - Because *G'* is *sparse*, thus the task of finding routes between nodes is much easier and generates less overhead than in the original graph

# Power spanners

- Let *G* be the communication graph obtained when all the nodes transmit at maximum power $r_{max}$, and assume *G* is connected. Label every edge (*u*,*v*) in *G* with the minimum power needed to send a message between *u* and *v*. Given any path *P* in *G*, the power cost of *P* is the sum of all the weights along the path. The *minimum-power path* between *u* and *v* in *G* is the path of minimum power cost among all the paths that connect *u* and *v*

- Let *G'* an arbitrary subgraph of *G*. The *power stretch factor* of *G'* with respect to *G* is the maximum over all possible node pairs of the ratio between the minimum-power path in *G'* and in *G*

- In words, the power stretch factor is a measure of the increase in the energy cost due to the fact that we communicate using the routing graph *G'* instead of *G*

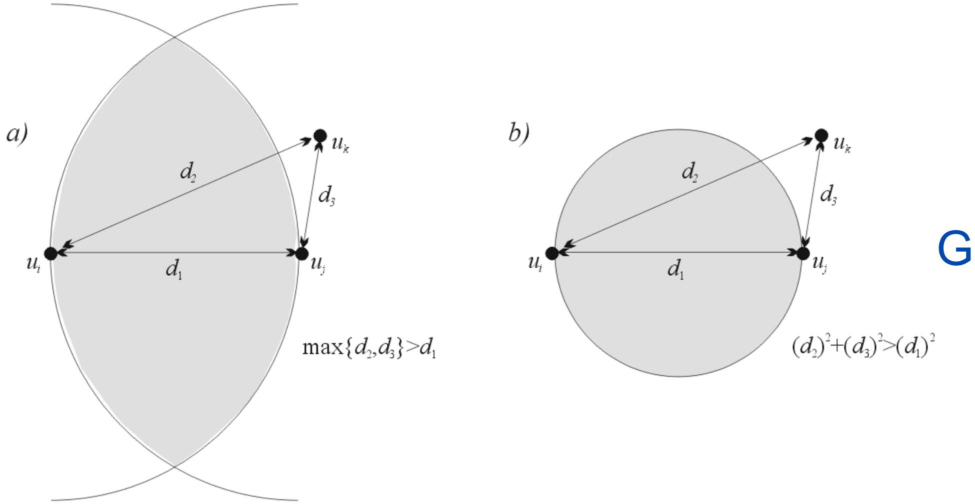
# Power spanners (2)

- Ideal features of a routing graph:
  - Low power stretch factor (i.e., *G'* should be a *power spanner* of *G*)
  - Linear number of edges (i.e., *G'* should be sparse)
  - Bounded node degree
  - Easily computable in a distributed and localized fashion

# RNG, GG, and other routing graphs

- The routing graphs introduced in the literature are variations of graphs known in the computational geometry community (*distance spanners*)
- Example of power spanners: the Relative Neighborhood Graph (RNG) and the Gabriel Graph (GG)

RNG




GG

# RNG, GG, and other routing graphs (2)

- Other routing graphs considered in the literature are the Restricted Delaunay Graph and the Yao Graph
- The table below summarizes the power stretch factor and maximum node degree of these routing graphs, assuming $\alpha = 2$

| | **Power** | **Degree** |
|---|---|---|
| RNG | $n - 1$ | $n - 1$ |
| GG | 1 | $n - 1$ |
| RDG | $\approx$25.84 | $\Theta(n)$ |
| YG | $\approx$4.05 | $n - 1$ |

**Remark 1:** the Gabriel Graph has optimal power stretch factor

**Remark 2:** all the routing graphs above are sparse (i.e., constant *average* node degree), but have *maximum* node degree linear in $n$

# Energy-efficient broadcast

- Other problem considered in the literature: determination of energy-efficient *broadcast graphs*
- Similarly to the case of unicast, the concept of *broadcast stretch factor* of a subgraph *G'* of *G* can be defined
- Also in this case, the goal is to find sparse broadcast spanners that can be computed in a distributed and localized fashion
- Unfortunately, this task is more difficult than in the case of unicast

# Energy-efficient broadcast (2)

- Finding the energy-optimal broadcast tree rooted at an arbitrary node *u* of *G* is NP-hard
- [Wieselthier et al.00]: the authors introduce three greedy heuristics for the minimum-power broadcast problem, based on the construction of the MST
- It has been proven that the broadcast stretch factor of the MST is *c*, for some $6 \leq c \leq 12$
- Unfortunately, *the MST cannot be computed using only local information*

# Corso Reti Mobili

# Distributed Topology Control

# Distributed Topology Control

- Previous Section: emphasis on finding a subgraph *G'* of the communication graph with "good" properties (for unicast/broadcast communications).
- Implicit in the previous approach: nodes adjust their transmit power on a per-packet basis (e.g., transmitting a message along an energy-efficient path in *G'*)
- Other research focused on trying to adjust the *maximum* nodes' transmitting range, in such a way that the communication graph remains connected.

⇓

*the topology of the communication graph itself is changed*

- Implicit in this approach: nodes set the maximum transmitting range periodically, and use the same (maximum) transmit power to send the messages.
- We call this approach **periodical topology control**

Istituto di Informatica e Telematica

# Distributed TC: desired properties

- Ideally, a TC protocol should:
  - Generate a *connected* communication graph of *low energy cost*
  - Generate a communication graph with small *physical* degree
  - Be fully distributed, asynchronous, and localized (esp. in case of mobility)
  - Rely on "low quality" information
  - Generate a connected topology free of unidirectional links

# TC protocols: information quality

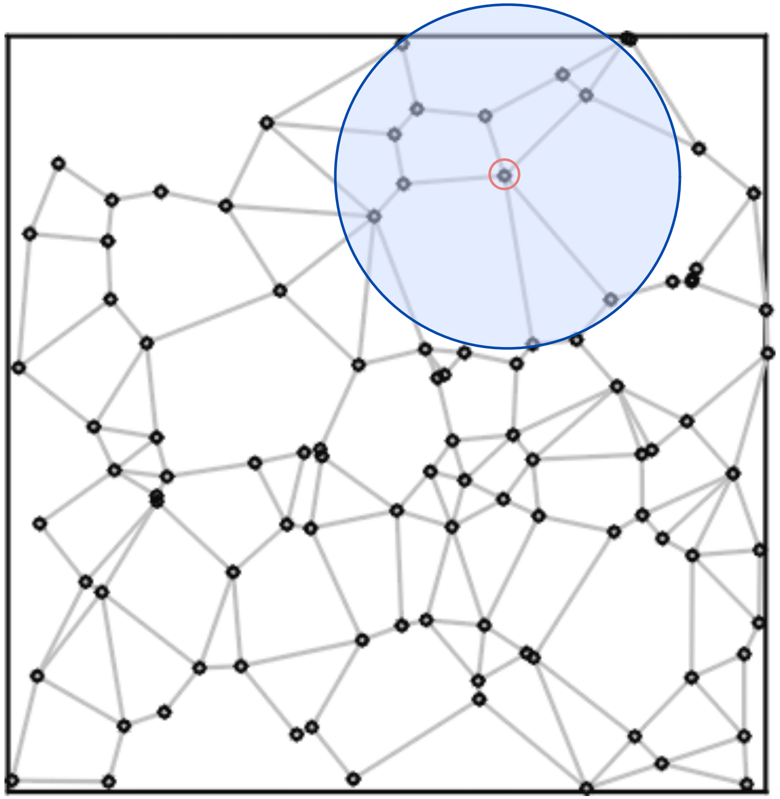
- Direct relationship between *information quality* and *energy consumption*: the more accurate is the information used by the protocol (e.g., location information), the more energy savings can in principle be achieved

- However, information quality (and, thus, the energy savings) must be carefully traded off with the *cost* incurred for making the information available to the nodes. With cost, we mean here either additional HW required on the nodes (e.g., GPS receiver), or message overhead, or both

# Physical vs. logical node degree

- Major advantage of topology control: reduce interferences, thus increasing network capacity
- node degree = "measure" of expected interference (*low is good*)
- So far, emphasis on reducing the *logical* node degree (number of edges in the final communication graph), and not on reducing the *physical* node degree (number of nodes in the transmitting range)
- **It is the physical node degree, not the logical, which determines the expected interference**

# Physical vs. logical node degree (2)



Logical degree = 5

Physical degree = 10

Example of communication graph produced by the CBTC protocol

# Distributed TC protocols

- We classify distributed TC protocols depending on the type of information used by the nodes to compute the topology
  - Location-based (High quality information):

    a node knows its own location, and the location of the neighbors
  - Direction-based (Medium quality information):

    a node knows the relative direction and distance to its neighbors
  - Neighbor-based (Low quality information):

    a node knows the IDs of its neighbors, and can order them according to some measure (e.g., distance, link quality, and so on)

# A location-based TC protocol

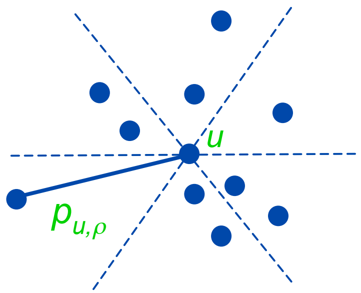
- LMST (Localized MST):
  - The MST topology has several desirable properties:
    - It is the sparsest possible connected topology
    - It approximates within a constant factor the optimal RA and the optimal broadcast tree
  - Drawback of the MST: its computation requires global knowledge, which is highly inefficient in ad hoc networks
  - Goal of LMST: building an approximation of the MST using only local information
  - Protocol (sketch):
    - every node computes a local MST on its visible neighborhood (all the nodes within maximum transmitting range)
    - these local MSTs rooted at each node are composed into a unique topology, which approximates the network-wide MST

# A direction-based TC protocol

- The Cone Based Topology Control (CBTC) protocol is based on the following idea:

  a node $u$ transmits with the minimum power $p_{u,\rho}$ such that there is at least one neighbor in every cone of angle $\rho$ centered at $u$

$\rho = \pi/3$



# Properties of the CBTC protocol

- The CBTC protocol produces a connected communication graph if

  $\rho \leq 2\pi/3$

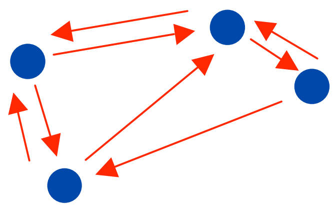
- The obtained communication graph is made symmetric by adding the reverse edge to every unidirectional link

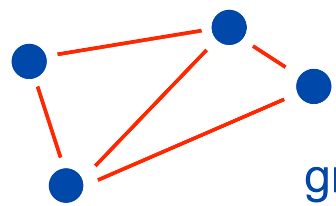
- A set of optimizations are also proposed, that prune energy-inefficient edges while not impairing connectivity and symmetry

# A neighbor-based TC protocol

- The goal of the KNeigh protocol is to connect every node in the network to its *k* closest neighbors, where *k* is a properly chosen constant
- The produced graph is made symmetric by adding reverse edges to all the unidirectional links

*k* = 2



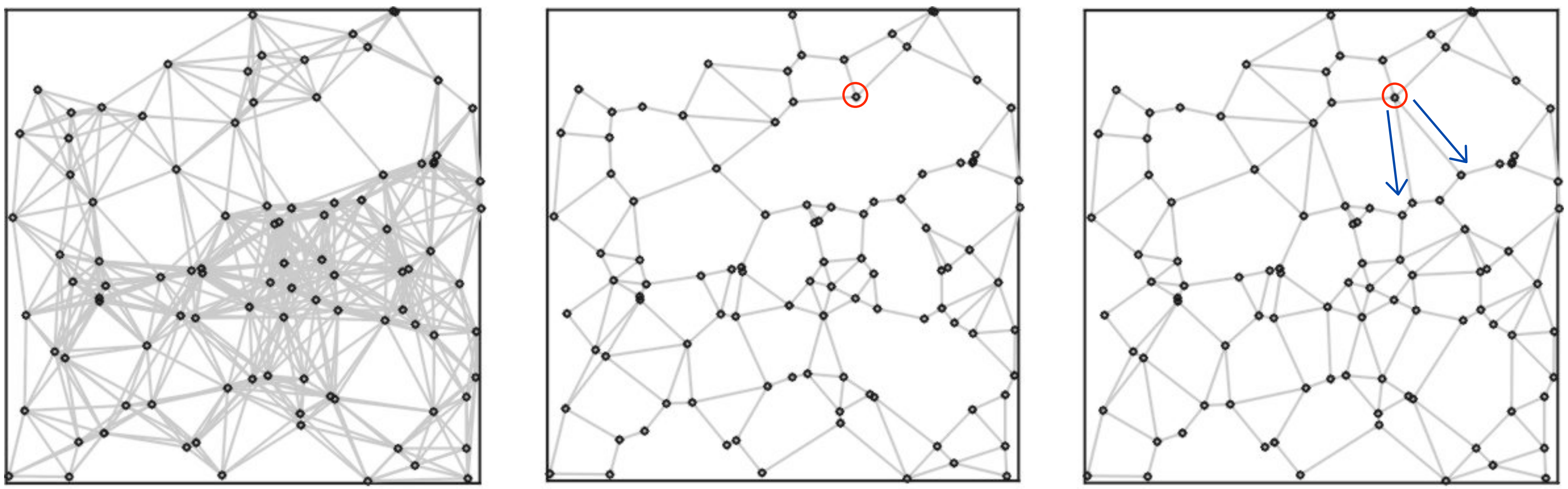


graph produced by KNeigh

# Properties of the KNeigh protocol

- If *n* network nodes are distributed uniformly at random in a square region, then setting $k = \log n$ is a necessary and sufficient condition (asymptotically) for obtaining a connected graph with high probability
- On the average, it is 20% more energy-efficient than CBTC (based on simulations)

# Sample topologies



*Homogeneous* *K-Neigh Ph2* *CBTC Ph2*

Sample topologies generated in case of CTR topology control (left), and after KNeigh (center) and CBTC (right) execution. The number of nodes is $n$ = 100

# Node Mobility

# Mobile networks

- Which is the impact of mobility on TC?
  - *Increased message overhead*: contrary to the stationary case, the protocol must be re-executed periodically in response to node mobility

    ⇩

    the “message efficiency” of the protocol is fundamental: protocols that exchange few messages to maintain the topology are needed

# Distributed TC and mobility

- Overhead depends on the frequency with which the reconfiguration procedure is executed, which in turn depends on:
  - The mobility pattern
  - The properties of the topology generated by the protocol

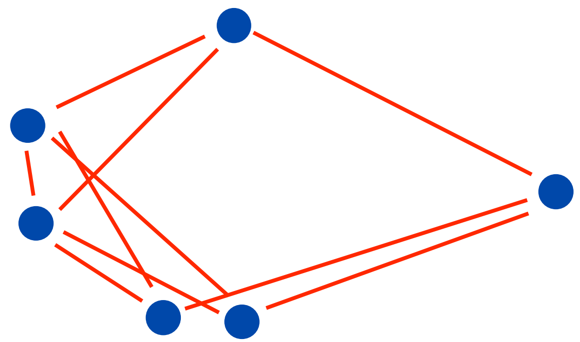
- Example: MST-based vs. *k*-neighbor based TC
  - The message overhead needed to build the MST is much larger than that needed to build the *k*-neighbors graph
  - Given the same mobility pattern, the MST should be reconfigured much more frequently than the *k*-neighbors graph

⇓

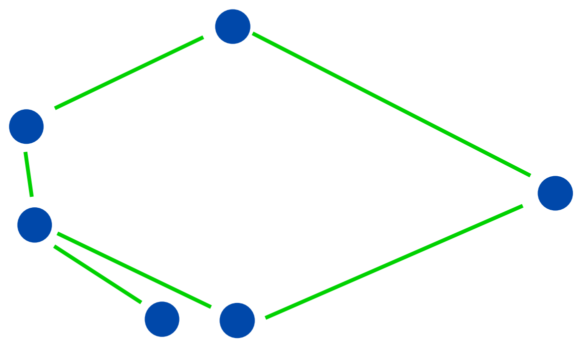
*k-neighbor based TC is more resilient to mobility than MST-based TC*

# MST vs KNeigh

KNeigh



MST



# Mobile TC protocols

- In order to be resilient to mobility, a TC protocol should be based on local information only

- Many protocols presented in the literature enjoy this property, but only some of them have been adapted to explicitly deal with node mobility
  - e.g.,  the authors of CBTC present a reconfiguration protocol that deals with node mobility

# Level-based Topology Control

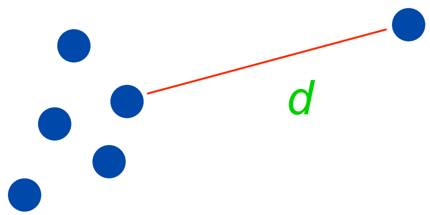
# Towards an implementation of TC

- To end this tutorial, we present two protocols (CLUSTERPOW and KNeighLev) that explicitly take into account a feature of current wireless transceivers: the transmit power can be set only to relatively few (5-6) levels

- For instance:
  - The CISCO Aironet 350 802.11 wireless card has the following transmit power levels: 1mW, 5mW, 20mW, 30mW, 50mW, 100mW
  - The transceiver of the Rockwell's Wins sensor node has the following transmit power levels: 0.12mW, 0.30mW, 0.96mW, 2.51mW, 3.47mW, 13.8mW, 19.1mW, 36.3mW

# The CLUSTERPOW protocol

- The protocol is an extension of the COMPOW protocol
- The goal of the CLUSTERPOW is to overcome a problem of COMPOW: when the node distribution is not “uniform”, the protocol performs very poorly



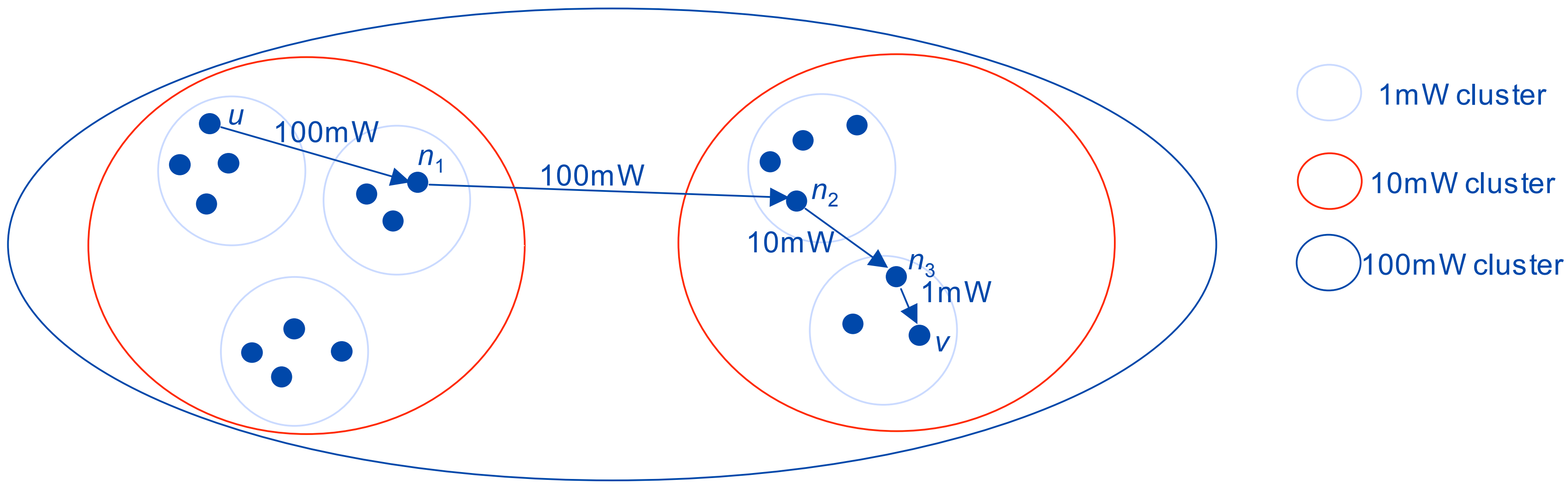

**COMPOW inefficiency:**

all the nodes have the same tx range, which must be at least equal to *d*

# The CLUSTERPOW protocol (2)

- Basic idea of CLUSTERPOW: every node *u* in the network maintains one routing table for each power level
- The routing table for level *i*, $RT_i$, is updated by a routing daemon (one for each level), and contains all the nodes that are reachable by *u* using power at most *i*
- This way, CLUSTERPOW induces a node clustering: for every node *u*, several clusters are defined, with the cluster at level *i* formed by the nodes in $RT_i$
- When *u* needs to send a message to *v*, it sends the message with power level *j*, where *j* is the minimum level such that $v \in RT_j$
- Intermediate nodes relay the message according to the same rule, until *v* is reached

# The CLUSTERPOW protocol (3)



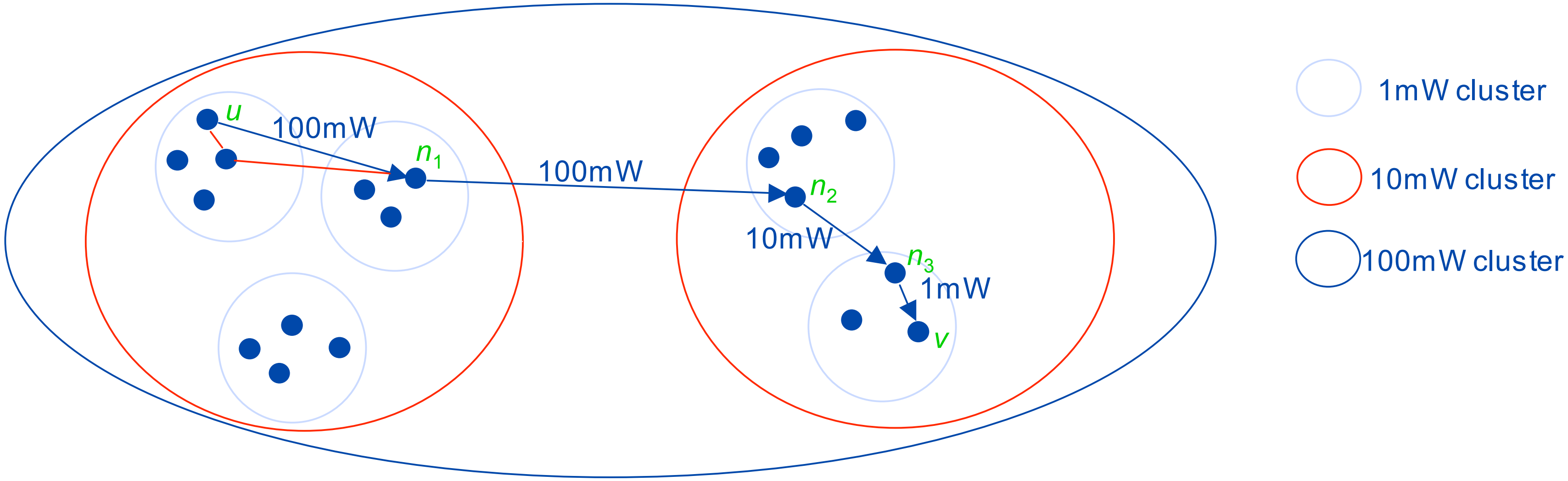
# CLUSTERPOW implementation

- CLUSTERPOW has been implemented in the 2.4.18 Linux kernel, on laptops using CISCO Aironet 350 cards
- Several routing daemons (one for each power level) are started on pre-assigned ports
- From the routing tables at all the power levels, the composition of the kernel routing table is done by the CLUSTERPOW agent running in user space
- The efficacy of CLUSTERPOW has been tested on the field, using 5 laptops
- Source code is available at *http://www.uiuc.edu/~kawadia/txpower.html*

# Technological problems

- The authors of CLUSTERPOW experienced several problems in its implementation
- The firmware of the CISCO cards forces a card reset every time the transmit power is changed. Then:
  - The power change latency is very large (about 100ms)
  - Changing the transmit power consumes a lot of energy
- Furthermore, *frequent power changes are very likely to crash the wireless card*
- As a consequence, any experimentation of CLUSTERPOW with a significant amount of traffic was impossible
- **Is per-packet topology control feasible?** With current technology, **NO**

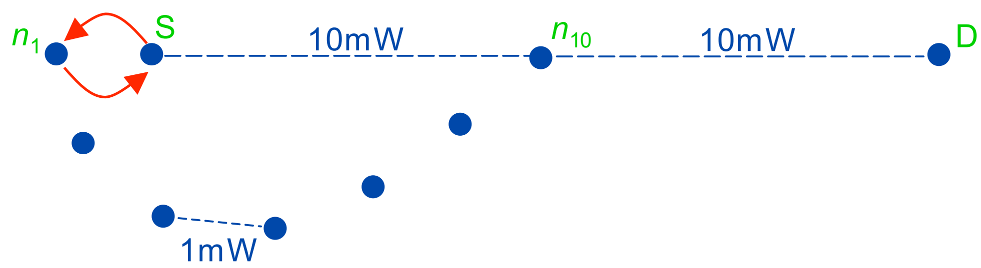
# A CLUSTERPOW inefficiency




**Remark:** the energy-efficiency of CLUSTERPOW can be improved. For instance, node *u* might have reached $n_1$ using two shorter hops, with an overall power consumption of 11mW, instead of 100mW

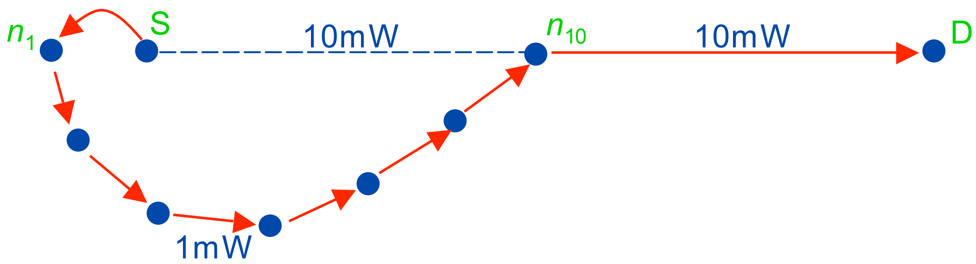
# Infinite loop

- If not implemented carefully, the optimization described in the previous slide can lead to packets getting into infinite loops!



# Tunneled CLUSTERPOW

- To avoid this, the packet is “tunneled” to its next hop using lower power levels, instead of sending the packet directly




- The implementation of T-CLUSTERPOW is very difficult: a dynamic per-packet tunneling mechanism would be needed, which is not available and hardly implementable

- Other problem: when the path between source and destination is long, the packet header becomes very large

# The KNeighLev protocol

- KNeighLev is a level-based implementation of *k*-neighbors topology control
- The basic idea is the following:
  - Every node starts transmitting at minimum power
  - After a certain stabilization period, the node checks its *symmetric* neighbors count (which can be easily derived from the set of detected incoming neighbors and its own power level)
  - If the symmetric neighbors count is below *k*, the node increases its power level, and sends a help message to inform its outgoing neighbors that it needs more symmetric neighbors
  - This process is repeated until the node has at least *k* symmetric neighbors, or the maximum transmit power is reached

# The KNeighLev protocol (2)

- The authors of KNeighLev show through simulation that $k = 4$ guarantees the formation of a communication graph which is connected w.h.p., for values of $n$ in the range 100 – 500
- They also present a set of optimizations, which remove energy-efficient links without impairing connectivity and symmetry
- Through simulation, it is shown that KNeighLev maintains its relative advantage in terms of energy efficiency (around 20%) with respect to the level-based version of CBTC, in which $p_{u,\rho}$ is rounded to the next higher power level

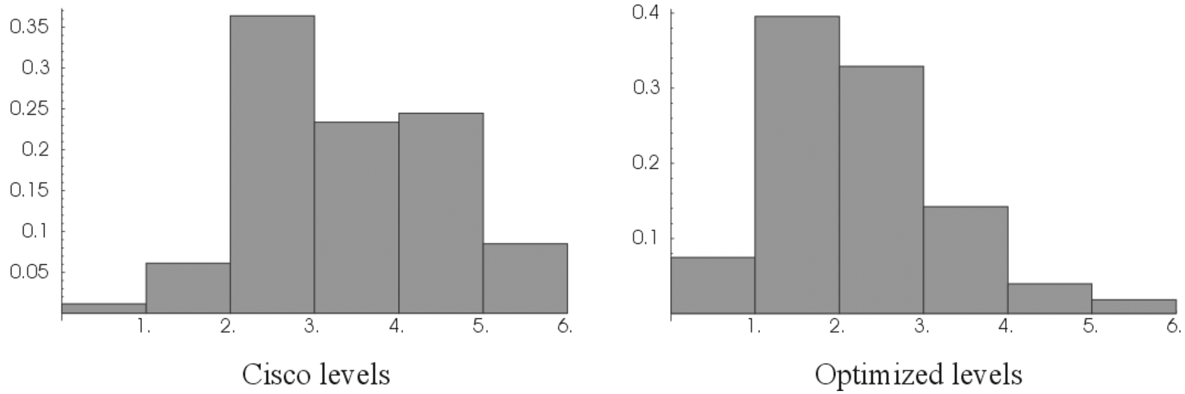
# Optimizing the power levels

- The power levels used in the simulation of KNeighLev are those typical of the CISCO Aironet 350 card
- This choice of the power levels is not necessarily optimal (see table below)

| level | CISCO | Optimized |
|---|---|---|
| 0 | 0.18 | 1 |
| 1 | 0.94 | 4 |
| 2 | 3.69 | 7 |
| 3 | 5.58 | 10 |
| 4 | 9.3 | 13 |
| 5 | 18.5 | 18.5 |

**Table 3.** Expected number of neighbors (under the assumption of uniform node distribution, with $n$=100) at the different transmit power levels, in case of CISCO power levels, and after optimization

# Optimizing the power levels (2)




Empirical distribution of the node power levels using the CISCO and optimized power levels (from [Blough et al.03b])

- Using the optimized power levels, the energy-efficiency of the topology generated by KNeighLev is improved of about 10% (with respect to the case of CISCO power levels)

- *Accurately choosing the power levels is very important, since it can provide further power savings at virtually no cost*

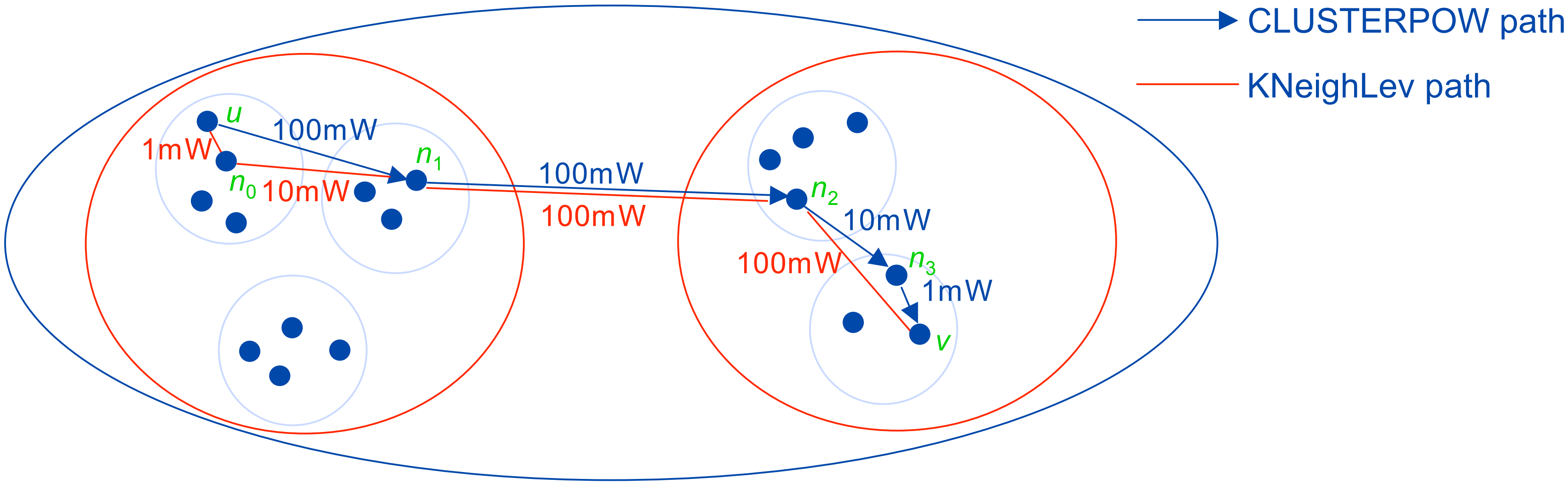
# CLUSTERPOW vs. KNeighLev

- CLUSTERPOW performs per-packet TC (hardly achievable with current technology)

- KNeighLev performs periodical TC: once the transmit power level is set, all the packets are sent using the same power. This approach is more coherent with the current transceiver technology

- What about the energy savings achieved by the two protocols? Let us return to the previous example....

# CLUSTERPOW vs. KNeighLev (2)




- Assuming that the power levels of $u$,$n_0$,$n_1$,and $n_2$ after KNeighLev execution are 1mW, 10mW, 100mW, and 100mw, respectively, we have that the overall power consumption of communicating a packet from $u$ to $v$ is 211mW for both protocols
- However, examples can be easily found in which CLUSTERPOW is more efficient than KNeighLev, or in which the contrary holds
- Intuitively, KNeighLev is more efficient in the uplink (from $u$ to $n_1$), while CLUSTERPOW is more efficient in the downlink (from $n_1$ to $v$)

# Conclusion

- In conclusion: the relative energy-efficiency of CLUSTERPOW and KNeighLev depends on several factors, such as node distribution and data traffic patterns

- The previous example motivates our feeling:

  once the technological problems with per-packet TC will be solved, ***a combination of periodical TC*** (to adjust the maximum transmit power and send broadcast messages) ***and per-packet TC*** (to send point-to-point messages) ***will be the best choice***