Ad hoc and Sensor Networks
Chapter 10: Topology Control
Goals of this Chapter

• Networks can be too dense – too many nodes in close (radio) vicinity
• This chapter looks at methods to deal with such networks by
  – Reducing/controlling transmission power
  – Deciding which links to use
  – Turning some nodes off
• Focus is on basic ideas, some algorithms
  – Complexity results are only very superficially covered
Overview

• Motivation, basics
• Power control
• Backbone construction (dominating sets)
• Clustering
• Combining hierarchical topologies and power control
• Adaptive node activity
Motivation: Dense Networks

- A typical characteristic of wireless sensor networks
  - deploying many nodes in a small area
    - ensure sufficient coverage of an area, or
    - protect against node failures
Motivation: Dense Networks

• In a very dense networks, too many nodes might be in range for an efficient operation
  – Too many collisions
  – Too complex operation for a MAC protocol,
  – Too many paths to chose from for a routing protocol, ...
Motivation: Dense Networks

• Idea: Topology control

• Topology control: Make topology less complex
  – Topology:
    • Which node is able/allowed to communicate with which other nodes
  – Topology control needs to maintain invariants, e.g., connectivity
Options for topology control

Topology control

Control **node** activity – deliberately turn on/off nodes

Control **link** activity – deliberately use/not use certain links
Options for topology control

**Flat network** – all nodes have essentially same role

**Hierarchical network** – assign different roles to nodes; exploit that to control node/link activity

- Power control
- Backbones (dominating sets)
- Clustering
Flat networks

• Main option: Control transmission power
  – Do not always use maximum power
  – Selectively for some links
  – Topology looks “thinner”
  – Less interference, ...

• Alternative: Selectively discard some links
  – Usually done by introducing hierarchies
Hierarchical networks – backbone

• Construct a **backbone** network
  – Some nodes “control” their neighbors
    – they form a (minimal) **dominating set**
  – Each node should have a controlling neighbor
  – Controlling nodes have to be connected (backbone)
  – Only links within backbone and from backbone to controlled neighbors are used
Hierarchical networks – backbone

- Formally: Given graph $G = (V, E)$, construct $D \subseteq V$ such that
  \[ \forall v \in V : v \in D \lor \exists d \in D : (v, d) \in E \]
Hierarchical network – clustering

• Construct clusters
  – Partition nodes into groups (“clusters”)
  – Each node in exactly one group
    • Except for nodes “bridging” between two or more groups
  – Groups can have clusterheads
  – Typically: all nodes in a cluster are direct neighbors of their clusterhead
  – Clusterheads are also a dominating set, but should be separated from each other – they form an independent set
Hierarchical network – clustering

- Formally: Given graph $G = (V, E)$, construct $C \subseteq V$ such that

$$\forall v \in V - C : \exists c \in C : (v, c) \in E$$

$$\forall c_1, c_2 \in C : (c_1, c_2) \notin E$$
Aspects of topology-control algorithms

• **Connectivity**
  - If two nodes connected in $G$, they have to be connected in $T$ resulting from topology control

• **Stretch factor** – should be small
  - *Hop stretch factor*: how much longer are paths in $T$ than in $G$?
  - *Energy stretch factor*: how much more energy does the most energy-efficient path need?
Aspects of topology-control algorithms

• **Throughput**
  – removing nodes/links can reduce throughput, by how much?
  – The reduced network topology should be able to sustain a comparable amount of traffic as the original network

• **Robustness to mobility**
  – require a small amount of such adaptations
  – avoid having the effects of a reorganization

• **Algorithm overhead**
Example: Price for maintaining connectivity

- Maintaining connectivity can be very “costly” for a power control approach
Overview

• Motivation, basics
• **Power control**
• Backbone construction (dominating sets)
• Clustering
• Combining hierarchical topologies and power control
• Adaptive node activity
Power control

• **flat topology**: *all nodes are operational and have the same tasks*

• This problem is closely linked to controlling the transmission power of nodes
Power control – magic numbers?

• Question: What is a good power level for a node to ensure “nice” properties of the resulting graph?

• Idea: Controlling transmission power corresponds to controlling the number of neighbors for a given node

• Is there a “magic number” that is good irrespective of the actual graph/network under consideration?

• Historically, k=6 or k=8 had been suggested as such “magic numbers”
  – However, they do not guarantee connectivity of the graph!!
Controlling transmission range (1/2)

• Assume all nodes have
  – identical transmission range \( r = r(|V|) \),
  – network covers area \( A \),
  – \( V \) nodes,
  – uniformly distribution.

• Fact: Probability of connectivity goes to zero if:

\[
r(|V|) \leq \sqrt{\frac{(1-\epsilon)A \log |V|}{\pi |V|}}, \text{ for any } \epsilon > 0
\]
Controlling transmission range (2/2)

• Fact: Probability of connectivity goes to 1 for

\[ r(|V|) \geq \sqrt{\frac{A(\log |V| + \gamma|V|)}{\pi|V|}} \]

if and only if \( \gamma_{|V|} \to \infty \) with \(|V|\)

\[ P(G \text{ is } k\text{-connected}) \approx \left(1 - \sum_{l=0}^{k-1} \frac{(\rho \pi r^2)^l}{l!} e^{-\rho \pi r^2} \right) \]

• Fact (uniform node distribution, density \( \rho \)):
Controlling number of neighbors (1/2)

• Knowledge about range also tells about number of neighbors
  – Assuming node distribution (and density) is known, e.g., uniform

• Alternative: directly analyze number of neighbors
  – Assumption:
    • Nodes randomly, uniformly placed
    • Only symmetric links are considered
    • Only transmission range is controlled, identical for all nodes,
Controlling number of neighbors (2/2)

• Result:
  For connected network, required number of neighbors per node is $\Theta(\log |V|)$
  – It is not a constant, but depends on the number of nodes!
  – For a larger network, nodes need to have more neighbors & larger transmission range! – Rather inconvenient
  – Constants can be bounded
Some example constructions for power control

• Basic idea: for most of the following methods:
  – Take a graph $G=(V, E)$, produce a graph $T=(V, E')$ that maintains connectivity with fewer edges
  – Assume, e.g., knowledge about node positions
  – Construction should be local (for distributed implementation)
Example 1: Relative Neighborhood Graph (RNG)

- Edge between nodes $u$ and $v$ if and only if there is no other node $w$ that is closer to either $u$ or $v$
- Formally: $\forall u, v \in V : (u, v) \in E' \iff \nexists w \in V : \max\{d(u, w), d(v, w)\} < d(u, v)$

This region has to be empty for the two nodes to be connected.
Example 1: Relative Neighborhood Graph (RNG)

- RNG
  - Maintains connectivity of the original graph
  - Easy to compute locally
  - Remove the longest edge from any triangle
  - But: Worst-case spanning ratio is $\Omega (|V|)$
  - Average degree is 2.6
  - Energy stretch is polynomial
Example 2: Gabriel graph

- Gabriel graph (GG) similar to RNG
- Difference:
  Smallest circle with nodes u and v on its circumference must only contain node u and v for u and v to be connected

This region has to be empty for the two nodes to be connected
Example 2: Gabriel graph

• Formally:

\[ \forall u, v \in V : (u, v) \in E' \iff \nexists w \in V : d^2(u, w) + d^2(v, w) < d^2(u, v) \]

• Properties:
  – Maintains connectivity,
  – Worst-case spanning ratio \( \Omega(|V|^{1/2}) \),
  – Energy stretch \( O(1) \) (depending on consumption model!),
  – Worst-case degree \( \Omega(|V|) \)
Example 3: Delaunay triangulation

- Connect any two nodes for which the Voronoi regions touch.

  \[ \text{Delaunay triangulation} \]

- \((u, v) \in E\) if and only if there is a circle that does contain no other nodes except \(u\) and \(v\).
Example 3: Delaunay triangulation

Problem:
- Might produce very long links; not well suited for power control
- Global information

Solution:
- Restricted Delaunay graph
- Distributed construction

• Properties:
  - Maintains connectivity,
  - Worst-case spanning ratio 2.5
Example: Spanning tree–based construction

• Based on local minimum spanning trees

• The idea:
  – each node will collect information about its neighboring nodes
  – construct a minimum spanning tree for these nodes, with energy costs used as link weights
  – Add links with lowest cost

• Properties:
  – Maintains connectivity,
  – Worst-case degree 6
  – It is possible to restrict to bidirectional links, and power control can be easily added
  – Moreover, the average node degree is small
Example: Relay regions and enclosures

• A crucial part of constructing a topology is deciding which neighbors to use

• relay region:
  – Given a node $i$ and another node $r$, for which points in the plane would $i$ use $r$ as a relay node in order to reduce the total power
Example: Relay regions and enclosures

• For each node $u$ inside this intersection of the complement of all relay regions, $u$ should communicate with $i$ directly.
Example: Relay regions and enclosures

• For each node $u$ outside this intersection of the complement of all relay regions, there is at least one other node that can provide a less power costly route than direct communication
Example: Relay regions and enclosures

• Ensure that a sufficient number of edges are preserved in the graph,
• $x$ and $z$ are maintained as neighbors
Centralized power control algorithm

• Goal:
  – Find topology control algorithm minimizing the maximum power used by any node
  – Ensuring simple or bi-connectivity

• Assumptions:
  – Locations of all nodes are known
  – path loss between all node pairs are known;
  – each node uses an individually set power level to communicate with all its neighbors
Centralized power control algorithm

• Idea: Use a centralized, greedy algorithm
  – Initially, all nodes have transmission power 0
  – Connect those two components with the shortest distance (cheapest) between them (raise transmission power accordingly)

• Second phase:
  – Remove links (=reduce transmission power) not needed for connectivity
Centralized power control algorithm

1) Connect A-C and B-D

2) Connect A-B

3) Connect C-D

4) Connect C-E and D-F

5) Remove edge A-B
Further reading on flat topology control

• Distributed power control

• Asymmetric maximum power
  – nodes having different maximum transmission ranges, resulting in the formation of asymmetric links.
  – They describe a distributed topology-control algorithm that minimizes maximum power and maintains the reachability of every node
Further reading on flat topology control

• Power control and mobility
  – power control interacts with ad hoc routing protocols
  – there is no single optimum density but that density should increase with movement

• Power control and IEEE 802.11
  – A node to choose a separate power level per neighbor, put explicit RSSI information into the RTS/CTS exchange packets
Overview

• Motivation, basics
• Power control
• *Backbone construction (dominating sets)*
• Clustering
• Combining hierarchical topologies and power control
• Adaptive node activity
Hierarchical networks – backbones

• Idea: Select some nodes from the network/graph to form a **backbone**
  – A connected, minimal, dominating set (MDS or MCDS)
  – Dominating nodes control their neighbors
  – Protocols like routing are confronted with a simple topology – from a simple node, route to the backbone, routing in backbone is simple (few nodes)

• Problem: MDS is an NP-hard problem
  – Hard to approximate, and even approximations need quite a few messages
Backbone by growing a tree

initialize all nodes’ color to white
pick an arbitrary node and color it grey

while (there are white nodes) {
    pick a grey node v that has white neighbors
    color the grey node v black
    foreach white neighbor u of v {
        color u grey
        add (v,u) to tree T
    }
}
Backbone by growing a tree – Example

1:

2:

3:

4:
Problem: Which gray node to pick?

• When blindly picking any gray node to turn black, resulting tree can be very bad

Solution:
Look ahead! One step suffices
Performance of tree growing with look ahead

• Dominating set obtained by growing a tree with the look ahead heuristic
  – at most a factor $2(1 + \ln(\Delta + 1))$ larger than MDS
    • $\Delta$ is maximum degree of the graph
  – It is automatically connected
  – Can be implemented in a distributed fashion as well
Connecting separate components

• In the previous approach, the set of nodes is always connected

• An alternative idea is to first construct a not necessarily connected dominating set and then in a second phase, explicitly connect the nodes in this set
  – Pick that node that turns most white nodes gray
  – Nodes which is chosen may not be gray
    • Chosen nodes are not connected
Connecting separate components

• Ensuring connectivity
  – Building a Steiner tree
    • Find a minimum spanning tree that contains all nodes of a predefined set of nodes, adding other nodes as required.
  – Some more gray nodes have to be turned black
Connecting separate components

- Observation:
  - at most two gray nodes can separate two “adjacent” black components

- Solution: turn one or two gray nodes in between black.

- At most $\ln \Delta + 3$ larger than the optimal ones
Some distributed approximations

1. Distribute-growing a tree
   - All gray nodes explore their two-hop neighborhood, determining the biggest yield that each node could achieve

   - Performance
     - $2\ln(\Delta+1)$ larger than the optimal ones
     - $O(|C|*(\Delta+|C|))$ time
     - $O(n|C|)$ messages
Some distributed approximations

1. Connecting a dominating set
   – How to adapt the centralized algorithm, determining a small dominating set and connecting it in a separate step
   – Process:
     • Every node broadcast its degree to all of its neighbors
     • Every node marks the neighboring node with the highest degree as its dominating node
       – Result in a dominating set, but not necessarily connected.
     • Connecting the set: a steiner tree
Start big, make lean

3. Marking nodes with unconnected neighbors
   – Idea:
     start with some, possibly large, connected dominating set, reduce it by removing unnecessary nodes
Start big, make lean

- Initial construction for dominating set
  - All nodes are initially white
  - Mark any node black that has two neighbors that are not neighbors of each other (they might need to be dominated)
  - Black nodes form a connected dominating set (proof by contradiction); shortest path between ANY two nodes only contains black nodes
  - Properties:
    - The set of marked nodes is a dominating set
    - Marked nodes are connected
    - Shortest path does not include any nonmarked nodes
    - The dominating set is not minimal. It can be trivial
- Needed: Pruning heuristics
Pruning heuristics

• Heuristic 1: Unmarked node $w$ if
  – Node $w$ and its neighborhood are included in the neighborhood of some node marked node $v$ (then $v$ will do the domination for $w$ as well)
  – Node $v$ has a smaller unique identifier than $u$ (to break ties)
Pruning heuristics

• Heuristic 2: Unmark node \( v \) if
  – Node \( v \)'s neighborhood is included in the neighborhood of two marked neighbors \( u \) and \( w \)
  – Node \( v \) has the smallest identifier of the tree nodes
  – Nice and easy, but only linear approximation factor
One more distributed backbone heuristic:

4. Span: Construct backbone, but take into account need to carry traffic – preserve capacity
   - Idea:
     - If the stretch factor (induced by the backbone) becomes too large, more nodes are needed in the backbone
     - Example: B has two neighbors A and C that can’t communicate via at most two backbone nodes, then B is a backbone node.
Further reading

• Weakly connected dominating
  – Finding a connected dominating set and are only looking for *weakly* connected dominating sets instead
  – Weakly connected: \( S \cup N(S) \) is connected, where \( S \) is a subset of \( V \)
  – Weakly connected dominating sets can be smaller than CDSs but retain most of their attractive properties
  – BUT, it is still NP-complete to find a minimal weakly connected set
Further reading

• Nontrivial approximation in constant time
  – It’s a nontrivial approximation ratio in a constant number of rounds
  – The approximation is based on a linear programming relaxation

• Generalized pruning heuristics
  – Remove any “gateway” node that is already covered by $k$ other gateways
  – This rule formulation generalizes the two separate heuristics proposed in reference
Overview

• Motivation, basics
• Power control
• Backbone construction (dominating sets)
• *Clustering*
• Combining hierarchical topologies and power control
• Adaptive node activity
Clustering

• Partition nodes into groups of nodes – clusters
• Many options for details
  – Are there clusterheads?
    One controller/representative node per cluster
  – May clusterheads be neighbors?
    If no: clusterheads form an independent set $C$
    Typically: clusterheads form a maximum independent set
  – May clusters overlap? Do they have nodes in common?
Clustering

• Further options
  – How do clusters communicate?
    • Some nodes need to act as gateways between clusters
    • If clusters may not overlap, two nodes need to jointly act as a distributed gateway
Clustering

• Further options
  – How many gateways exist between clusters?
  – What is the maximal diameter of a cluster? If more than two, then clusterheads are not necessarily a maximum independent set
  – Is there a hierarchy of clusters?
Maximum independent set

• Computing a maximum independent set is NP-complete
  – Can be approximate within \((\Delta + 3)/5\) for small \(\Delta\), within \(O(\Delta \log \log \Delta / \log \Delta)\) for large values; \(\Delta\) bounded degree
Maximum independent set

- A maximum independent set is also a dominating set
- Maximum independent set is not necessarily intuitively desired solution
  - Example: Radial graph, with only \((v_0, v_i) \in E\), for \(i = 1, \ldots, n\)
A basic construction idea for independent sets

- Make each node a clusterhead that locally has the largest attribute value.
- Once a node is dominated by a clusterhead, it abstains from local competition, giving other nodes a chance.
Determining gateways to connect clusters

• Suppose: Clusterheads have been found
• How to connect the clusters, how to select gateways?
• It suffices for each clusterhead to connect to all other clusterheads that are at most three hops
  – Resulting backbone is connected
• Formally: Steiner tree problem
Rotating clusterheads

• Serving as a clusterhead can put additional burdens on a node
  – For MAC coordination, routing, ...

• Let this duty rotate among various members
  – Periodically reelect – useful when energy reserves are used as discriminating attribute (round-robin fashion)

• LEACH – determine an optimal percentage P of nodes to become clusterheads in a network
  – Use 1/P rounds to form a period
Some more algorithm examples (1)

• Weighted Clustering
  – A cluster should not exceed a maximum size $\delta$
  – Battery power
  – Mobility (slow nodes are preferred)
  – Closeness of neighbors
Some more algorithm examples (2)

- **Emergent** algorithm for cluster establishment
  - In this algorithm, every node can be in three states:
    - *unclustered* (unaware of any cluster),
    - *clusterhead*,
    - *Follower*
  - Process:
    - unclustered node become clusterhead and recruit their neighbors as followers
    - Clusterheads can abdicate if there is a follower node that would make a better clusterhead
Multi-hop clusters

• Clusters with diameters larger than two can be useful, e.g., when used for routing protocol support
• Formally: Extend “domination” definition to also dominate nodes that are at most $d$ hops away
• Different tilt: Fixing the size (not the diameter) of clusters
  – Idea: Use growth budgets – amount of nodes that can still be adopted into a cluster, pass this number along with broadcast adoption messages, reduce budget as new nodes are found
Passive clustering

• Constructing a clustering structure brings overheads
  – Not clear whether they can be amortized via improved efficiency

• Question: Eat cake and have it?
  – Have a clustering structure without any overhead?
  – Maybe not the best structure, and maybe not immediately, but benefits at zero cost are no bad deal...
Passive clustering

• Passive clustering
  – Whenever a broadcast message travels the network, use it to construct clusters on the fly
  – Node to start a broadcast: Initial node
  – Nodes to forward this first packet: Clusterhead
  – Nodes forwarding packets from clusterheads: ordinary/gateway nodes
  – And so on... Clusters will emerge at low overhead
Overview

• Motivation, basics
• Power control
• Backbone construction (dominating sets)
• Clustering
• *Combining hierarchical topologies and power control*
• Adaptive node activity
Combining hierarchical topologies and power control

1. Pilot-based power control
   - The main advantage is that the power control logic can be “centralized” in the clusterheads, simplifying the problem of a fully distributed power control
   - Clusterheads use power control on both pilot signals and on normal data packets
   - The pilot signal power control is used to control the cluster membership as nodes only join a cluster based on these pilots
   - The data packet power control is used to ensure adequately low errors for faraway nodes and efficient transmission for nearby nodes
Combining hierarchical topologies and power control(2)

- **Ad hoc Network Design Algorithm (ANDA)**
  - Allowing the clusterheads to control the size of their cluster by power control
  - and concrete rules are derived to maximize the network lifetime

- **The assumptions for this approach are**
  - the positions of ordinary nodes and of (preselected) clusterheads are known,
  - the traffic load is evenly distributed over ordinary nodes,
  - the lifetime of a clusterhead is proportional to its initial energy supply and inversely proportional to $cr^\alpha + dn$
    - $r$ is the coverage radius of a clusterhead,
    - $n$ is the number of cluster members,
    - $\alpha$ is the path-loss coefficient,
    - $c, d$ are constants.
Overview

- Motivation, basics
- Power control
- Backbone construction (dominating sets)
- Clustering
- Combining hierarchical topologies and power control
- Adaptive node activity
Adaptive node activity

• Remaining option: Turn some nodes off deliberately
• Only possible if other nodes remain on that can take over their duties
• Example duty: Packet forwarding
  − Approach: Geographic Adaptive Fidelity (GAF)

• Observation: Any two nodes within a square of length \( r < R/5^{1/2} \) can replace each other with respect to forwarding
  − \( R \) radio range
• Keep only one such node active, let the other sleep
Conclusion

• Various approaches exist to trim the topology of a network to a desired shape

• Most of them bear some non-negligible overhead
  – At least: Some distributed coordination among neighbors, or they require additional information
  – Constructed structures can turn out to be somewhat brittle – overhead might be wasted or even counter-productive

• Benefits have to be carefully weighted against risks for the particular scenario at hand