Power/Topology Control
Topics to be covered

- Power/Topology Control problem.
- Importance of topology control
- Some topologies from computational geometry
- Minimum-energy Path-preserving Graphs
- Algorithm
- Performance of the algorithm
- Discussion/question/comment
Wired network topologies

- Fully Connected Network Topology
- Mesh Network Topology
- Common Bus Topology
- Star Network Topology
- Ring Network Topology
- Ring
Wireless Network topology
Wireless Network topology
Power/Topology Control
Power/Topology Control
Power/Topology Control

Changes Topology
Maximum Powered Vs. Optimum Powered

Topology control problem: Choosing optimal power level by each node in a distributed fashion
Need for Topology control

- Drop long-range neighbors:
  - Conserve Energy
  - Reduce Interference
  - Sparse Graph, Low Degree
  - Planarity
- But still stay connected
Motivation:

Affects battery power

Affects many layers performance!
Motivation:

Determines intermediate nodes to use while routing (Network layer)
Motivation:

Number of contending nodes (MAC layer)
Motivation:

Traffic carrying capacity (Overall net performance)
In summary:

Affects many layers performance

- Determines intermediate nodes to use while routing (Network layer)
- Number of contending nodes (MAC layer)
- Level of Congestion (Transport Layer)
- Traffic carrying Capacity (Overall Net performance)
- Reduced interference (Physical layer)
Some Interesting Topologies from computational geometry:

- Minimum Spanning Tree MST(V)
  - A subset of $E$ of $G$ of minimum weight which forms a tree on $V$. 
MST construction: revisited

Kruskal’s

- Create a forest $F$ (a set of trees), where each vertex in the graph is a separate tree
- Create a set $S$ containing all the edges in the graph
- while $S$ is nonempty and $F$ is not yet spanning
  - remove an edge with minimum weight from $S$
  - if the removed edge connects two different trees
    then add it to the forest $F$, combining two trees into a single tree
MST construction: revisited

<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>
MST construction: create a forest

<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>
MST construction: create a forest

<table>
<thead>
<tr>
<th>Edge</th>
<th>ae</th>
<th>cd</th>
<th>ab</th>
<th>be</th>
<th>bc</th>
<th>ec</th>
<th>ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Graph:

- Nodes: a, e, b, c, d
- Edges: a<->e, b<->c, b<->d, c<->d
- Weights: 1, 3, 4, 5, 6, 7
MST construction: create a forest

<table>
<thead>
<tr>
<th>Edge</th>
<th>ae</th>
<th>cd</th>
<th>ab</th>
<th>be</th>
<th>bc</th>
<th>ec</th>
<th>ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Diagram:
- Edge ae is chosen first.
- The graph is divided into three disjoint sets:
  - Set containing a and e.
  - Set containing b and c.
  - Set containing d.
- Edges 3, 4, 6, and 7 are added to connect the sets.
MST construction: create a forest

<table>
<thead>
<tr>
<th>Edge</th>
<th>ae</th>
<th>cd</th>
<th>ab</th>
<th>be</th>
<th>bc</th>
<th>ec</th>
<th>ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Graph:
- Edge 'ae' with weight 1
- Edge 'cd' with weight 2
MST construction: create a forest
MST construction: create a forest

<table>
<thead>
<tr>
<th>Edge</th>
<th>ae</th>
<th>cd</th>
<th>ab</th>
<th>be</th>
<th>bc</th>
<th>ec</th>
<th>ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Diagram:
- a connected to e with weight 1
- b connected to e with weight 3
- e connected to c with weight 4
- c connected to d with weight 2
- a connected to b with weight 5
- be is the edge added first.
- The minimum spanning tree includes edges ae, cd, ab, be, bc, ec, ed.
MST construction: create a forest

SORT THE EDGES

<table>
<thead>
<tr>
<th>Edge</th>
<th>ae</th>
<th>cd</th>
<th>ab</th>
<th>be</th>
<th>bc</th>
<th>ec</th>
<th>ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Diagram:

- Nodes: a, e, b, c, d
- Edges: a-e (weight 1), e-c (weight 2), b-e (weight 5), b-c (weight 6), b-d (weight 7)
MST construction: revisited
(Example taken from Wiki)

<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>

Diagram:
- Edge weights: ab=3, ae=1, bc=5, be=4, cd=2, ed=7, ec=6.
- Graph with vertices a, b, c, d, e and edges with weights as indicated.
Relative Neighborhood Graph RNG(V)

- An edge $e = (u, v)$ is in the RNG(V) iff there is no node $w$ in the “lune” of $(u, v)$,
- i.e., no node with $|u, w| < |u, v|$ and $|v, w| < |u, v|$.
RNG

<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>

Diagram:
- Edge weights: ab=3, ae=1, bc=5, be=4, cd=2, ed=7, ec=6
- Nodes: a, b, c, d, e
- Edges and weights: a→e (1), b→c (5), b→d (3), c→d (2), c→e (4), d→e (7)
Gabriel Graph

Let $\text{disk}(u,v)$ be a disk with diameter $|u,v|$ that is determined by the two points $u$, $v$. 
The Gabriel Graph $\text{GG}(V)$ is defined as an undirected graph (with $E$ being a set of undirected edges). There is an edge between two nodes $u, v$ iff the disk$(u, v)$ including boundary contains no other points.
Mathematically, an edge $e = (u, v)$ is in the $GG(V)$ iff there is no node $w$ with

$$|u, w|^2 + |v, w|^2 \leq |u, v|^2$$
<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>

Gabriel Graph

Diagram of a Gabriel Graph with vertices a, b, c, d, e and edges and weights as specified in the table.
GG

<table>
<thead>
<tr>
<th>Edge</th>
<th>ab</th>
<th>ae</th>
<th>bc</th>
<th>be</th>
<th>cd</th>
<th>ed</th>
<th>ec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>

Graph with weighted edges:

- Edge ab with weight 3
- Edge ae with weight 1
- Edge bc with weight 5
- Edge be with weight 4
- Edge cd with weight 2
- Edge ed with weight 7
- Edge ec with weight 6
RNG and Gabriel Graph

RNG \subseteq GG
Properties:

• Theorem 1:

\[ MST \subseteq RNG \subseteq GG \]

• Corollary:
  Since the MST is connected all the graphs in Theorem 1 are connected.
However not all topologies from computational geometry are possible to be constructed in MANET!

<table>
<thead>
<tr>
<th>Topology</th>
<th>Possible</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST</td>
<td>NO</td>
</tr>
<tr>
<td>RNG</td>
<td>YES</td>
</tr>
<tr>
<td>GG</td>
<td>YES</td>
</tr>
</tbody>
</table>
Now we will see a minimum-energy path-preserving topology control algorithm
What is Energy?

Energy = Power × Time

One can not save much in time

To save Energy save Power

TX power and RX power
Power consumed over an edge and over a path

How to determine the power requirement to transmit a packet from A to B?

Ans: DEPENDS ON THE RADIO PROPAGATION MODEL
Propagation mechanisms

A: free space
B: reflection
C: diffraction
D: scattering

reflection: object is large compared to wavelength
scattering: object is small or its surface irregular
A propagation model determines what will happen to the transmitted signal while in transit to receiver.
Free Space (LOS) Model

- Path loss for unobstructed LOS path
- Power falls off:
  - Proportional to $d^2$
Two Ray Propagation model

- Path loss for one LOS path and 1 ground (or reflected) bounce
- Ground bounce approximately cancels LOS path above critical distance
- Power falls off
  - Proportional to $d^2$ (small $d$)
  - Proportional to $d^4$ ($d>d_c$)
Power consumed over an edge and over a path

- Commonly used path loss model,
  - Transmit power falls off as $1/d^n$, $n \geq 2$

$$P_{\text{transmit}} = td^n$$

- Cost of a $m$ length path $p$,

$$td_1^n + td_2^n + \ldots + td_m^n$$
Relay is useful

\[ P_{\text{transmit}} = td^n \]

\[ td_1^n + td_2^n \leq t(d_1 + d_2)^n \]

- Problem: Infinite number of relays!

\[ P_{\text{total}} = td^n + c \]

- Cost of a \( m \) length path \( p \),

\[ (td_1^n + c) + (td_2^n + c) + \ldots + (td_m^n + c) \]
A graph is said to have minimum energy property if all minimum energy paths are preserved.
Minimum Graph \((G_{\text{min}})\) having minimum energy property

- How to construct \(G_{\text{min}} = (V,E_{\text{min}})\)?
- \(E_{uv}\) will survive in \(G_{\text{min}}\) if \(\text{cost}(E_{uv}) < \text{cost}(r)\) for all path \(r\) in \(G_{\text{max}}\) of length > 1
Example: Naïve Construction of $G_{\text{min}}$

In the graph cost over edge is the required transmission power cost.
Example: Construction of $G_{\min}$

$4+1=5 > 2$
Example: Construction of $G_{\text{min}}$

$4+2+2=8 > 2$
Example: Construction of $G_{\text{min}}$

4 + 2 + 1 + 7 + 2 = 16 > 2
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$

\[ 2 + 1 = 3 < 4 \]
Example: Construction of $G_{\min}$
Example: Construction of $G_{\min}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$

$2+2+1 = 5 < 7$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Example: Construction of $G_{\text{min}}$
Theorem on $G_{\text{min}}$

Suppose $G$ is a maximum-powered graph. The subgraph $G_{\text{min}}$ of $G$ is the smallest graph having minimum energy property.

The proof has two parts:

- $G_{\text{min}}$ has the minimum energy property
- $G_{\text{min}}$ is the smallest subgraph having such property.
A new look at $G_{\text{min}}$

- $G_i = (V, E_i)$ where $e \in E_i$ if, $\text{cost}(e) < \text{cost of any path of exact } i \text{ length in } G_{\text{max}}$

$$G_{\text{min}} = G_2 \cap G_3 \cap \ldots \cap G_{n-1}$$
Example: New Construction of $G_{\text{min}}$
Example: Construction of $G_2$
Example: Construction of $G_3$
Example: Construction of $G_4$
Example: Construction of $G_{\text{min}}$
Only $G_2$

- $G_i = (V, E_i)$ where $e \in E_i$ if, 
  cost($e$) < cost of a $i$-length path in $G_{\max}$

$$G_{\min} = G_2 \cap G_3 \cap \ldots \cap G_{n-1}$$

- $G_2, G_3, \ldots$ all contains $G_{\min}$ that’s why all are minimum-energy path-preserving
- Only $G_2$ can be constructed with one hop neighbor information.
Now we will see a distributed algorithm for constructing $G_2$
To relay or not to relay

- **Case 1**—Use relay if: \[ P_{uv} + P_{vw} + r \leq P_{uw} \]

- **Case 2**—Don’t use otherwise: \[ P_{uv} + P_{vw} + r > P_{uw} \]
Algorithm to construct $G_2$ (a node $s$ is running the algorithm)

$A_s =$ Set of neighbors using Maximum power

$N_s =$ Set of neighbors in $G_2$

$\xi_s =$ set of non-neighbors in $G_2$

- How a node $s$ will create $N_s$?
- Initialize, $A_s = N_s = \xi_s =$ empty set.
- Broadcast Neighbor Discovery message (NDM) with max power and collect responses
- Each node $v$ receiving NDM will send back response including its position information. $<$v, x, y$>$
Phase 1: Algorithm to construct $G_2$

- For the response from a node $v$ do the following:
  - For each $w$ in $A_s$ do
    - if $\text{cost}(s,w) + \text{cost}(w,v) + r < \text{cost}(s,v)$ then
      $$\xi_s = \xi_s \cup \{v\}$$
    - else if $\text{cost}(s,v) + \text{cost}(v,w) + r < \text{cost}(s,w)$ then
      $$\xi_s = \xi_s \cup \{w\}$$
  - $A_s = A_s \cup \{v\}$
  - $N_s = A_s - \xi_s$
  - Set the transmission range to only reach nodes in $N_s$
Algorithm to construct $G_2$ (When cost is replaced by tx power)

- For the response from a node $v$ do the following:

  - For each $w$ in $A_s$ do

    - if $P_{sv} + P_{vw} + r \leq P_{sw}$ then
      
      $\xi_s = \xi_s \cup \{w\}$

    - else if $P_{sw} + P_{wv} + r \leq P_{sv}$ then
      
      $\xi_s = \xi_s \cup \{v\}$

  - $A_s = A_s \cup \{v\}$

  - $N_s = A_s - \xi_s$
Example:

\[ A_s = \phi \quad N_s = \phi \]
Example

\[ A_s = \phi \quad N_s = \phi \quad \xi_s = \phi \]
Example

\[
\begin{align*}
A_s &= \{a\} \\
\xi_s &= \emptyset \\
N_s &= \{a\}
\end{align*}
\]
Example

\[ A_s = \{a\} \quad \xi_s = \emptyset \quad N_s = \{a\} \]
Example

\[
\begin{array}{|c|c|c|}
\hline
\text{Reply from a} & A_s = \{a\} & \xi_s = \emptyset & N_s = \{a\} \\
\text{Reply from c} & A_s = \{a,c\} & \xi_s = \emptyset & N_s = \{a,c\} \\
\hline
\end{array}
\]
Example

<table>
<thead>
<tr>
<th>Reply from</th>
<th>$A_s$</th>
<th>$\xi_s$</th>
<th>$N_s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>${a}$</td>
<td>$\emptyset$</td>
<td>${a}$</td>
</tr>
<tr>
<td>c</td>
<td>${a,c}$</td>
<td>$\emptyset$</td>
<td>${a,c}$</td>
</tr>
<tr>
<td>d</td>
<td>${a,c,d}$</td>
<td>${d}$</td>
<td>${a,c}$</td>
</tr>
</tbody>
</table>

Diagram:

```
 a
    \
   /   \    \
 b  c  d
```
# Example

<table>
<thead>
<tr>
<th>Reply from a</th>
<th>$A_s = {a}$</th>
<th>$\xi_s = \emptyset$</th>
<th>$N_s = {a}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reply from c</td>
<td>$A_s = {a,c}$</td>
<td>$\xi_s = \emptyset$</td>
<td>$N_s = {a,c}$</td>
</tr>
<tr>
<td>Reply from d</td>
<td>$A_s = {a,c,d}$</td>
<td>$\xi_s = {d}$</td>
<td>$N_s = {a,c}$</td>
</tr>
<tr>
<td>Reply from b</td>
<td>$A_s = {a,c,d,b}$</td>
<td>$\xi_s = {a,d}$</td>
<td>$N_s = {b,c}$</td>
</tr>
</tbody>
</table>
Now we will see an energy efficient construction of $G_2$.
Basic idea: discover neighbors with incrementally
Basic idea: discover neighbors with incrementaly
Basic idea: discover neighbors with incrementally

Diagram:

- a
- b
- NDM
- s
- c
- d
Basic idea: discover neighbors with incrementaly
Relay Region

\[ P_{uv} + P_{vz} + r < P_{uz} \]
Enclosure
Enclosure
RNSA

1. Transmission power = $P_0$
2. Loop
   3. Broadcast NDM and collect responses
   4. Update neighbor set
   5. If enclosure is found or reached at Max power exit
   6. Increase transmission power by $P_{inc}$
7. endloop
Comparison on number of edges

(a) Original graph  
(b) $G_2$ generated by RNSA
RNSA

1. Transmission power = $P_0$

2. Loop
   3. Broadcast NDM and collect responses
   4. Update neighbor set
   5. If enclosure is found or reached at Max power exit
   6. Increase transmission power by $P_{inc}$

7. endloop
Problem of RNSA

Enclosure by neighbor

Enclosure by maximum transmission range
Problem of RNSA

Deployment area 670m X 670m, TX range 250m

Figure 4.10: Percentage of nodes with enclosures by neighbors