Routing in WSNs

Preliminaries

- **Communication**
  - Communication is expensive ➡ Use only if necessary
  - A node communicates only with its neighbors
  - Wireless advantage: all neighbors in broadcasting range listen
- **Unit Distance Graph (UDG)**
  - Communication range is the same for all nodes
  - Connectivity graph of nodes: nodes are connected if their normalized distance is less than 1

Topology Control

- **Establish communication with other nodes**
  - Discovery of nodes in communication range
  - Topology is determined by communication range
  - Adjust communication range (save energy)
- **Critical transmission range (CTR)**
  - Minimum transmission range to connect all nodes
- **Assumption**
  - All nodes have the same transmission range r

Topology Control

- **Locations are known a priori**
  - *Spanning tree* (ST) of a graph $G = (V, E)$
    - Set of $|V|-1$ edges connecting all vertices
  - *Minimum spanning tree* (MST) of G
    - ST where the sum over all costs $c_{ij} = (v_i, v_j)$ is minimal
  - CTR is longest edge in MST
- **Nodes locations are not known accurately**
  - Theory of geometric random graphs if nodes are randomly and uniformly distributed
  - CTR for n nodes: $r = C \times (\log n / n)^{½}$

Criteria for Routing Algorithms

- **Size of the routing table**
  - Preferably small
- **Quality of the route for a given destination**
  - Fastest, most reliable, highest throughput route
  - Most energy-efficient route
- **Update cost**
  - Nodes can die, move, or join

MANETs vs WSNs

- **Commonality**
  - Wireless Sensor Networks (WSNs) are MANETs
- **Differences**
  - Scalability: number of nodes is potentially much larger
  - Fault tolerance: sensor nodes are more prone to failure
  - Energy-awareness: nodes have a very limited amount of energy

Terminology

- **Routing**
  - Transport messages between two nodes
- **Data dissemination**
  - Transport messages from a node to many nodes
- **Broadcasting**
  - Transport messages from a node to all nodes (in range)
- **Data gathering**
  - Transport messages from nodes (within a region) to a sink

Terminology

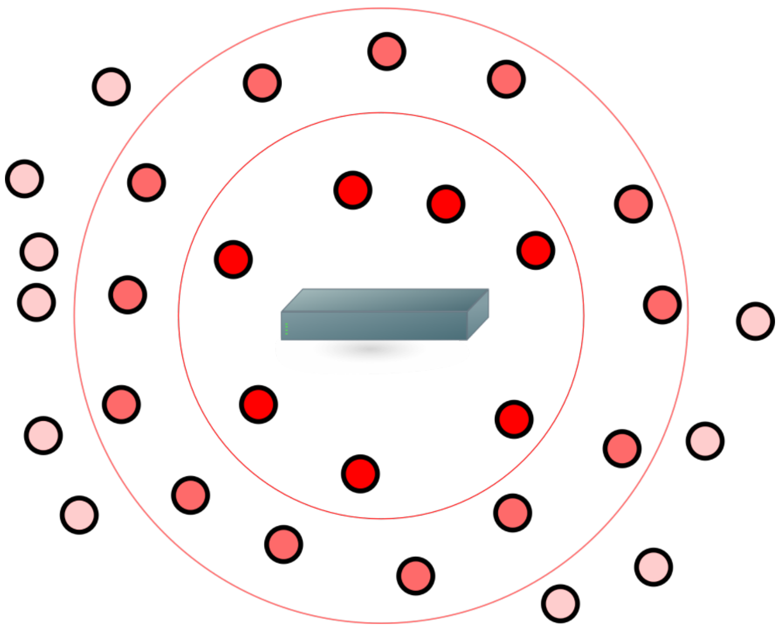
- **Base station**
  - Node providing a gateway or central processing
- **Sink**
  - Node requesting information
- **Source**
  - Node generating information (event)
- **Interest**
  - Message requesting a certain type of information

Sensor Network Architectures

- Layered architecture
- Flat architecture
- Hierarchical or clustered architecture

Layered Architecture

- **Paradigm**
  - A single powerful base station (BS)
  - Layers: nodes with the same hop-count to the BS
- **Application**
  - In-building wireless backbones: BS is an access point to a wired connection

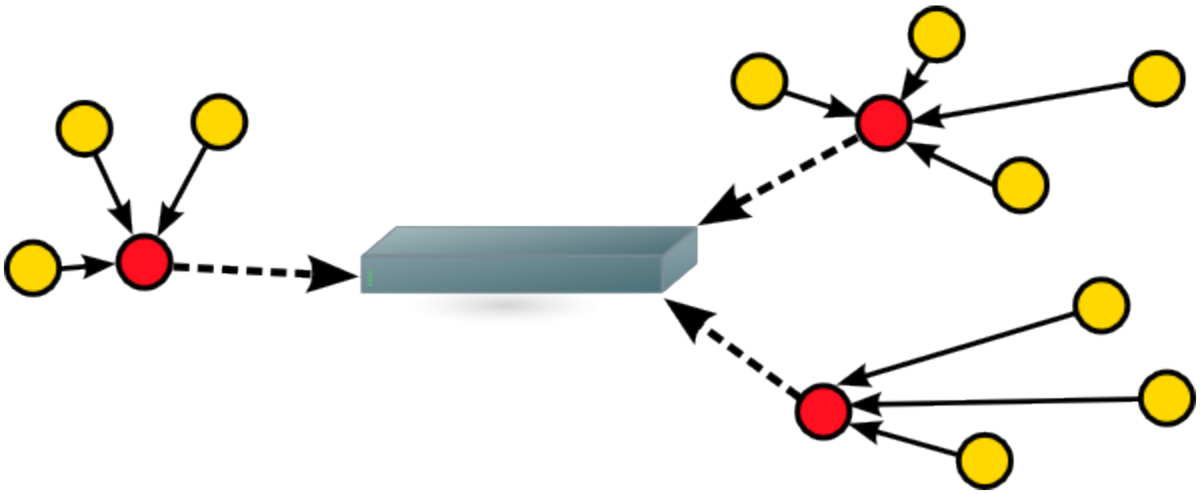


Flat Architecture

- **Paradigm**
  - Each node has the same role
- **Large number of nodes**
  - No unique *global* identifier
  - Data-centric routing
  - Location-based routing

Hierarchical Architecture

- **Paradigm**
  - Nodes are organized into clusters
  - Nodes in a cluster send their messages to cluster heads (CHs)
  - CHs send their messages to a base station (BS)



Topology-based Routing

Review: Topology-based Routing

- **Approach**
  - Use information about *links* in the network
- **Proactive protocols**
  - Compute routes before routing
- **Reactive protocols**
  - Discover routes on-demand
- **Hybrid protocols**
  - Compute routes once, then update

Flooding

- **Technique**
  - Each node that receives a message broadcasts this message if
    - the node is not a goal node
    - the maximum hop count is not reached
- **Reactive protocol**
  - Requires no topology maintenance
  - No (complex) route discovery necessary
  - Often used as backup strategy: limited flooding

Disadvantages of Flooding

- **Implosion**
  - A node often receives the same message from different neighbors
- **Duplication**
  - Nodes send the same message to their neighbors
- **Resource blindness**
  - Not aware of the energy levels of the mobile device

Gossiping

- **Limited broadcast**
  - Nodes do not broadcast received messages to every neighbor but only to a randomly selected neighbor
- **Advantage**
  - No implosion and lower overhead
- **Disadvantages**
  - Long travel time for messages
  - No delivery guarantee

Radius Growth

- **Problem: locality insensitivity**
  - Destination is a few hops away but the entire network is flooded
- **Flood with growing radius**
  - For a message the time-to-live (TTL) is decreased at every node
  - Rounds of different floods with increasing TTLs (1, 2, 3, …)
  - But: how to stop if the destination is found?
- **Slow flooding**
  - A timeout for nodes before a message is forwarded
  - Destination is found: a second fast flooding that stops the previous flood

Source Routing

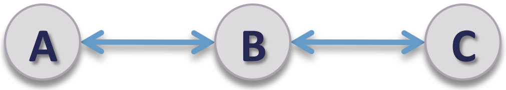
- **Problem**
  - Nodes store routing information for other nodes
- **Idea**
  - Source node stores the whole path to the destination
  - Source node encodes the path with every message
  - Nodes on the path remove their ID from the message before relaying the message to the next node
- **Discussion**
  - Nodes only store the paths they need
  - However, not efficient if mobility/data ratio is high
  - How to deal with asymmetric links?

DSR (Dynamic Source Routing)

- **Route discovery**
  - Packet needs to be sent: a node checks whether a *cached* route is available
  - If not available: RREQ with the address of S and D and a unique identification number
  - Node adds its own address to the route record
  - RREP: message reaches D or a node with a route to D
- **Route maintenance**
  - Acknowledgments
  - Route errors lead to updates

Improving Source Routing

- Caching of routes (DSR)
- Local search
  - Flooding with TTL+1
- Hierarchy of nodes
  - Nodes with the same IP prefix are in the same direction
- Clustering
  - Good if heterogeneous network but level of indirection and overhead
- Implicit acknowledgment
  - Symmetric links: node A automatically hears the communication from B to C



Directed Diffusion (DD) I

- **Motivation**
  - No central authority
  - Sensor networks are resource constrained
  - Nodes are tied to physical locations
  - Nodes may not know the topology
  - Nodes are generally stationary
- **How can we get data from the sensors?**

Directed Diffusion II

- **Data centric**
  - Individual nodes are unimportant
- **Request driven**
  - Sinks place requests as interests
  - Sources satisfying the interest to be found
  - Intermediate nodes route data toward sinks
- **Localized reinforcement and repair**
- **Multi-path delivery**

DD: Interest and Event Naming

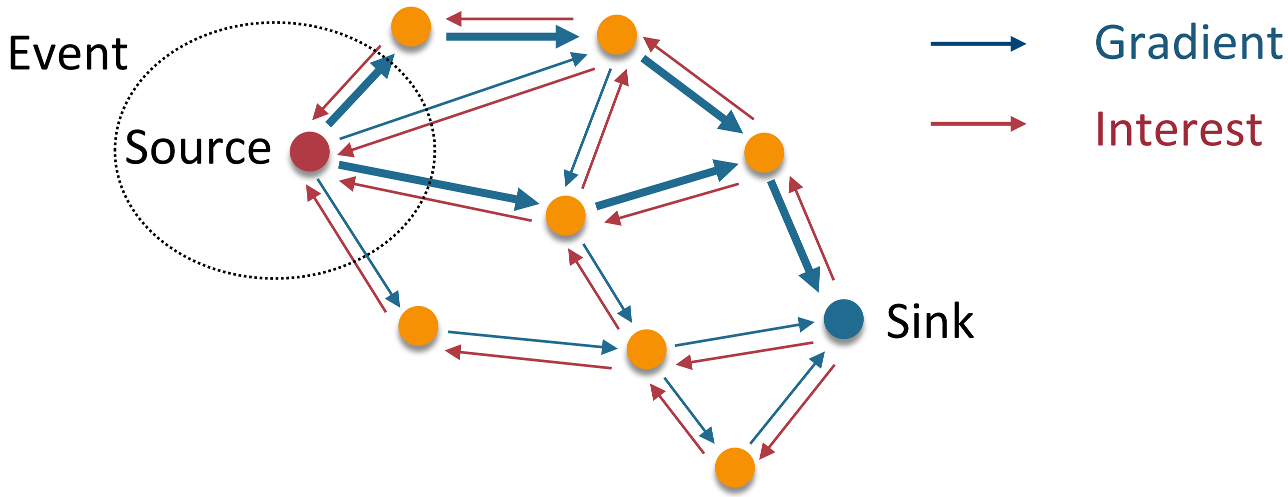
- **Query/interest of sink**
  - `Type = wheeled vehicle  // detect vehicle location`
  - `Interval = 20 ms (event data rate)  // e.g., 1 sec initially`
  - `Duration = 10 sec (time to cache)  // for the next 10 sec`
  - `Rect = [-100, 100, 200, 400]`
- **Reply of sensor node**
  - `Type = wheeled vehicle  // type of vehicle seen`
  - `Instance = truck  // instance of this type`
  - `Location = [125, 220]  // node location`
  - `Intensity = 0.6  // signal amplitude measure`
  - `Confidence = 0.85  // confidence in the match`
  - `Timestamp = 01:20:40  // event generation time`
- **Attribute-Value pairs**
  - No advanced naming scheme

Directed Diffusion IV

- **Sinks**
  - Broadcast interest to neighbors
  - Initially use a low data rate to find sources with minimal energy consumption
- **Interests**
  - Cached by neighbors
- **Gradients**
  - Point back to where interests came from
- **Sources**
  - Receive an interest: route data along gradients

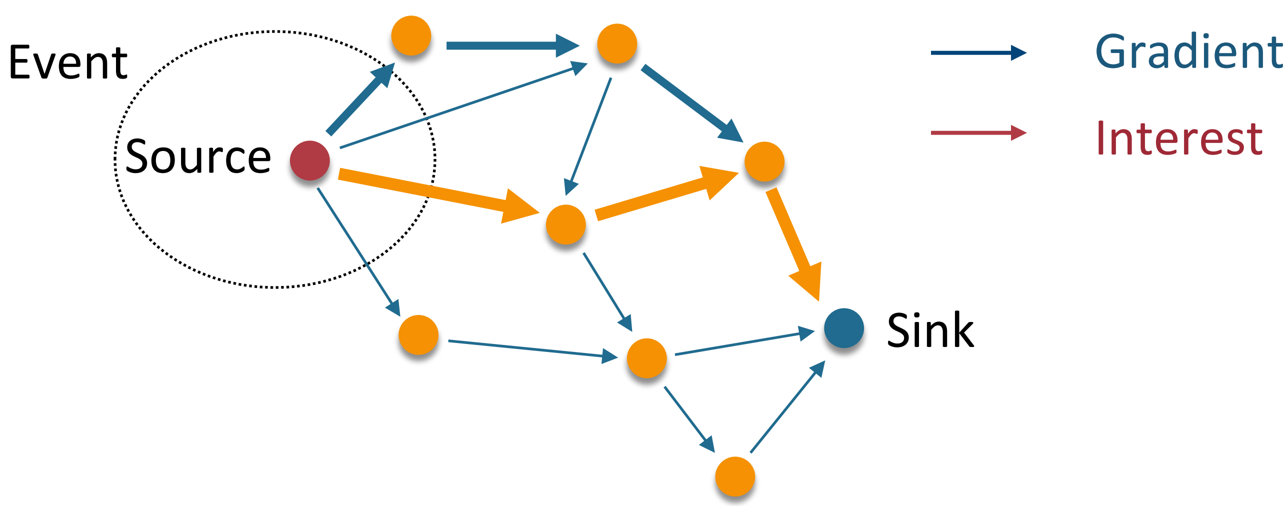
DD: Interest Propagation

- Flood interest
- Constrained or directional flooding based on location is possible
- Directional propagation based on previously cached data



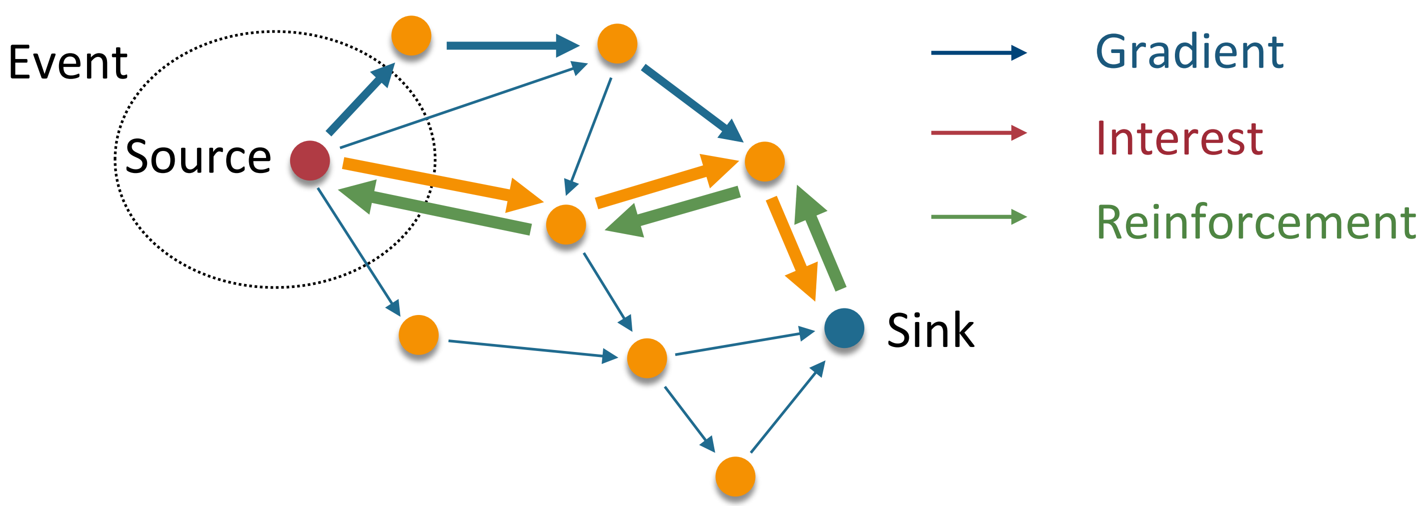
DD: Data Propagation

- **Multipath routing**
  - Consider each gradient's link quality



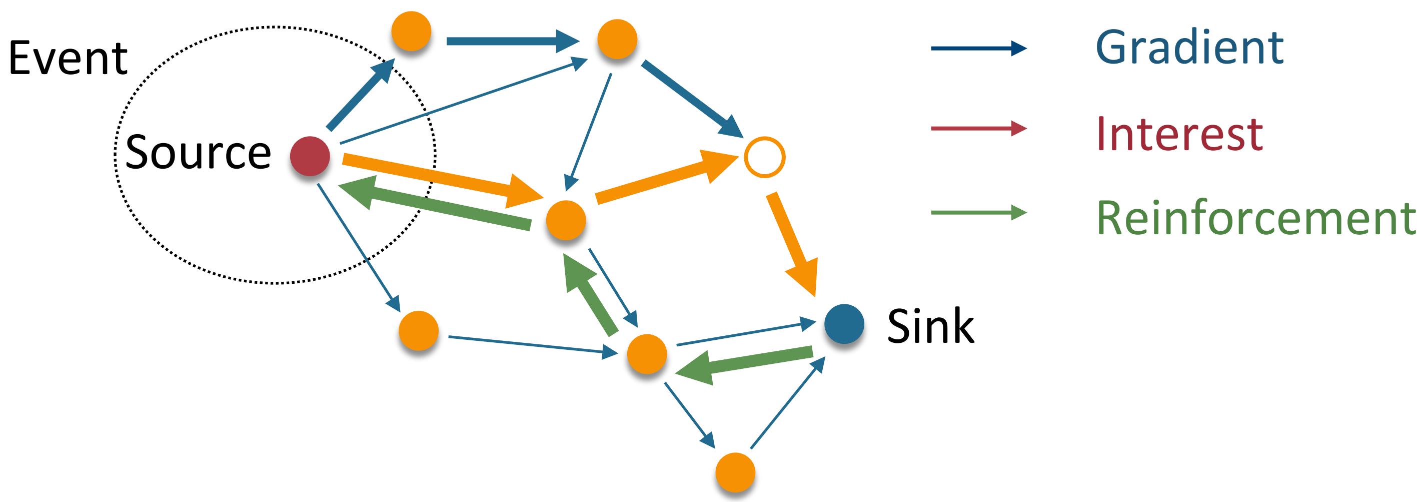
DD: Reinforcement

- Reinforce one of the neighbors after receiving initial data
- Neighbor who consistently performs better than others
- Neighbor from whom most events received



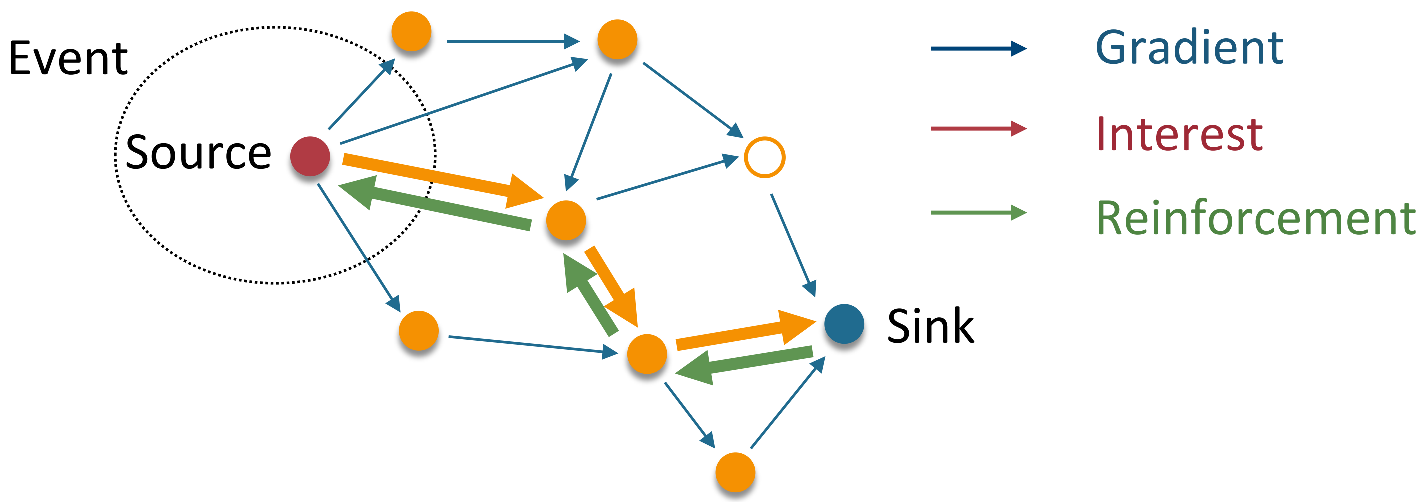
Negative Reinforcement I

- **Explicitly degradation**
  - Resend interest with lower data rate along path
- **Time out**
  - Without periodic reinforcement, a gradient will be torn down



Negative Reinforcement II

- **Explicitly degradation**
  - Resend interest with lower data rate along path
- **Time out**
  - Without periodic reinforcement, a gradient will be torn down



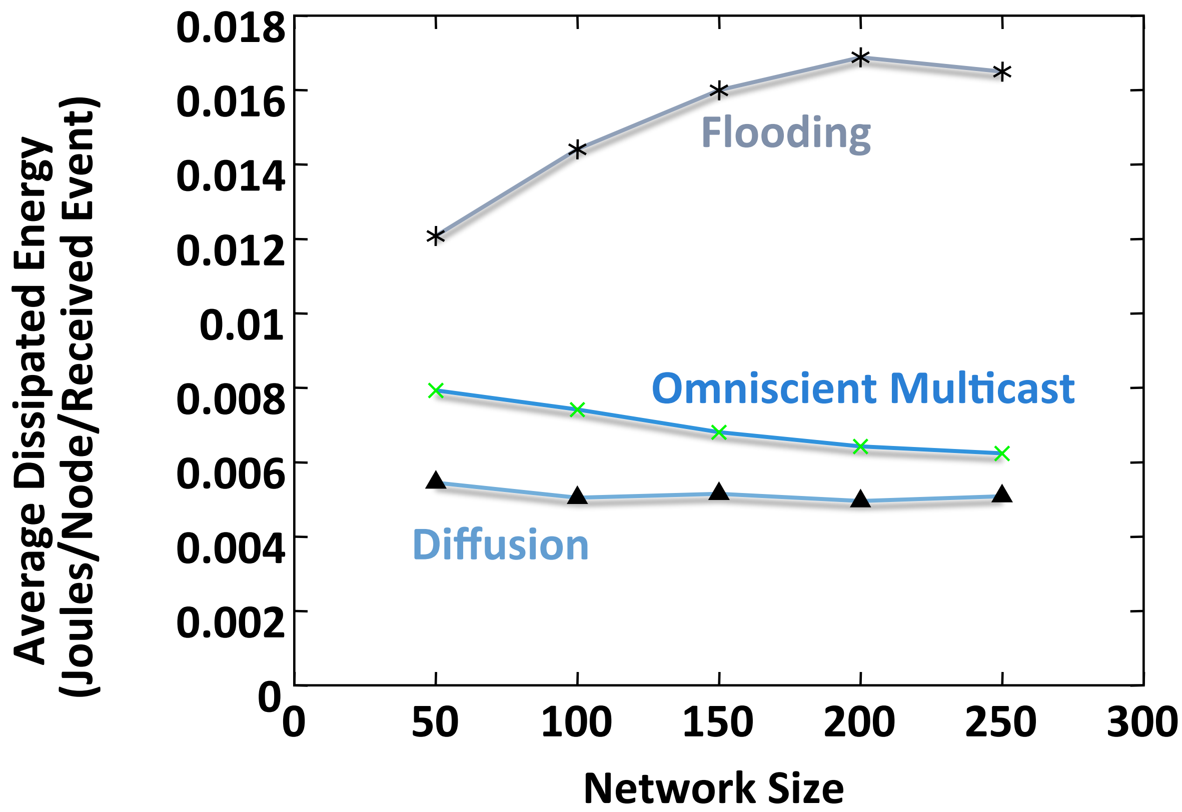
Evaluation

- **Simulation: ns2**
- **Modified 802.11 MAC for energy use calculation**
  - Idle time: 35mW, receive: 395mW, transmit: 660mW
- **Random node placement**
  - 50 – 250 nodes (increment by 50), 50 nodes are deployed in 160m×160m (increase the SN size to keep the density constant)
  - 40m radio range

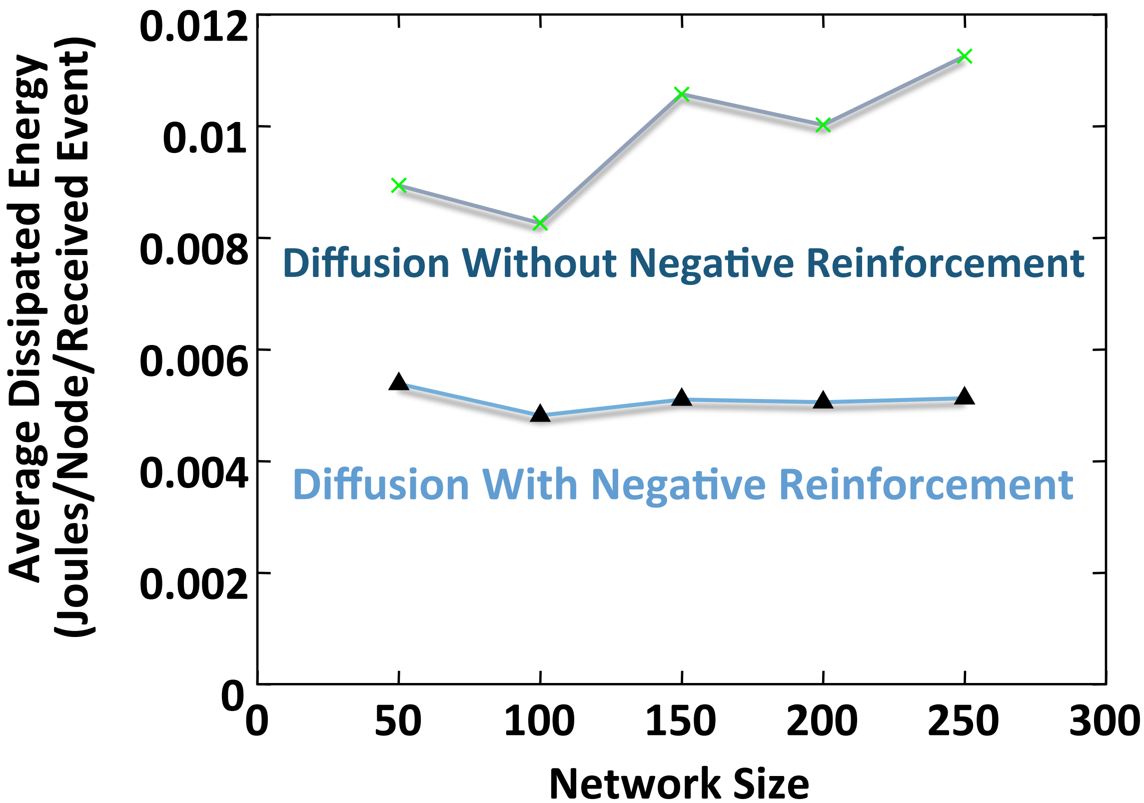
Metrics

- **Baselines**
  - Flooding
  - Omniscient multicast: each source transmits its events along a shortest path tree to all sinks; ignore tree construction cost (centrally computed)
- **Average dissipated energy**
  - Ratio of total energy expended per node to number of distinct events received at sink
  - Measures average work expenditure as function of network size

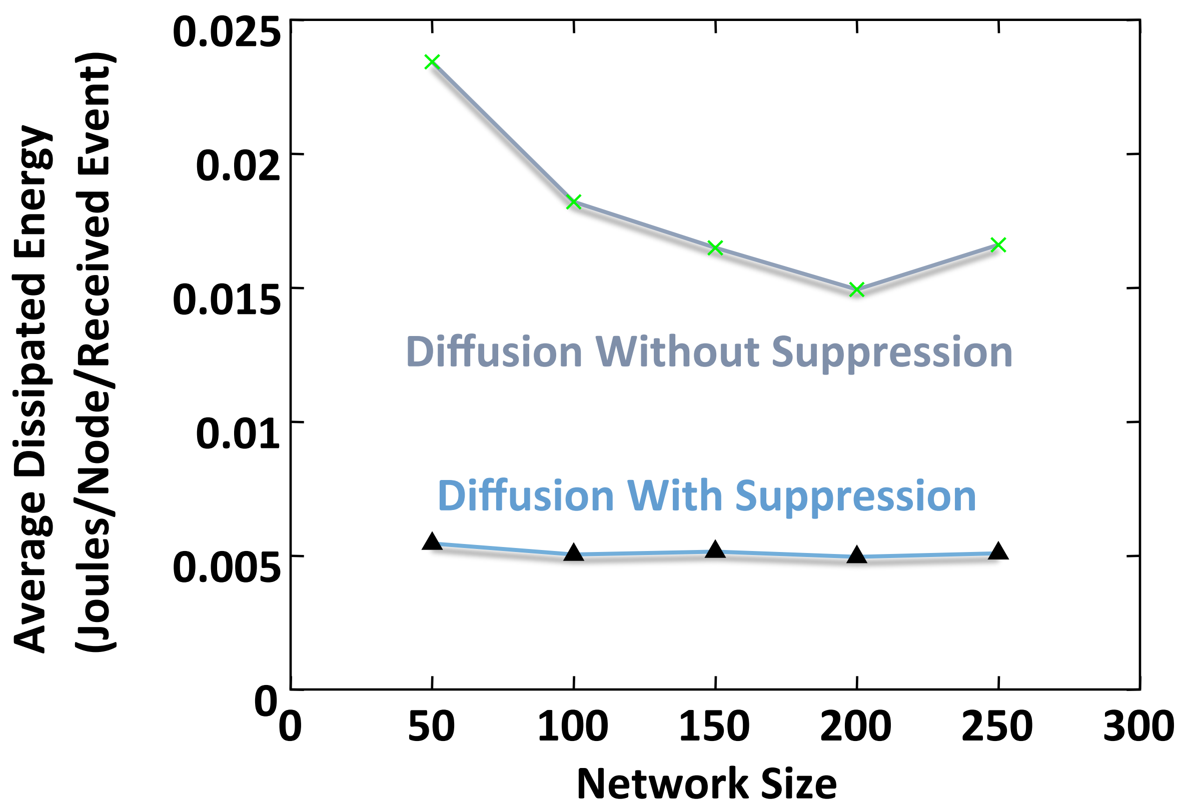
Average Dissipated Energy



Impact of Negative Reinforcement

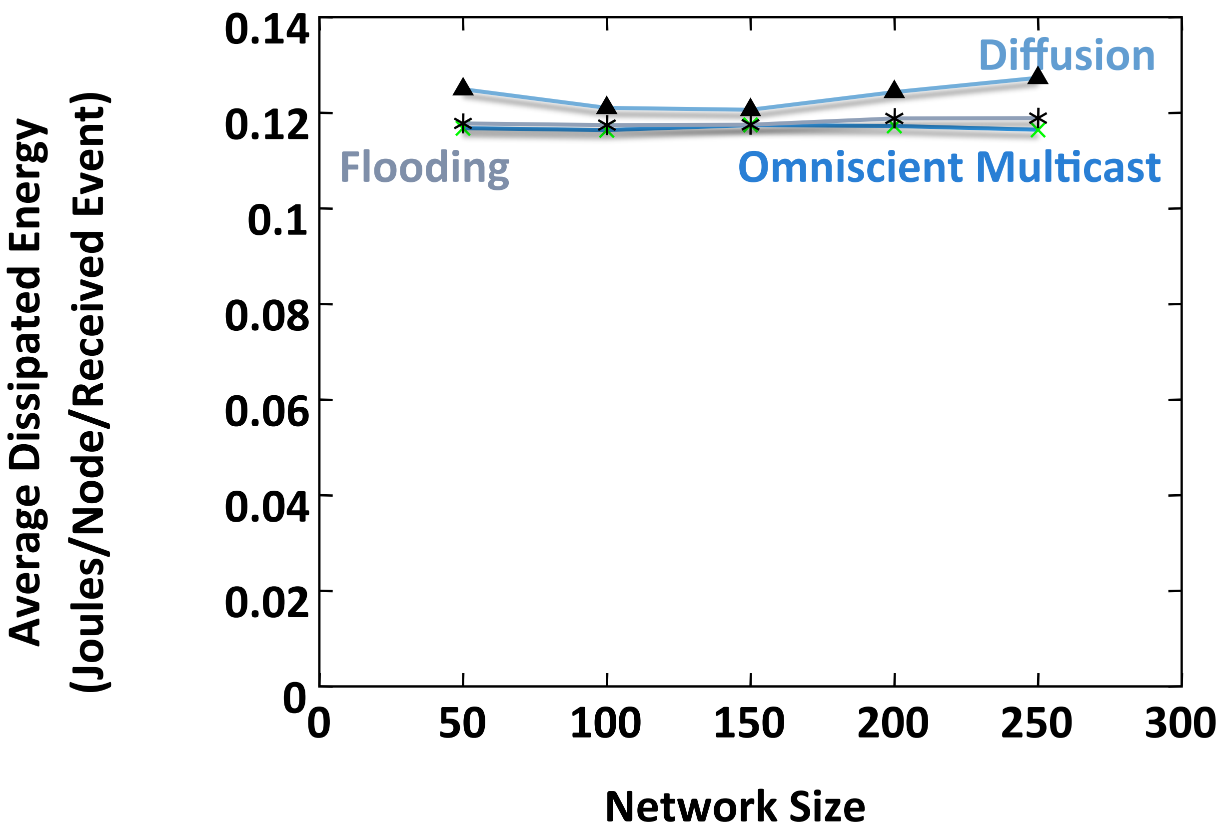


Impact of In-Network Processing



Why no suppression for omniscient multicast?

Average Dissipated Energy



Unmodified 802.11 MAC is dominated by idle energy
1.6W transmission, 1.2W reception, and 1.15W idle

Directed Diffusion: Extension

- **Push diffusion**
  - Sink does not flood interest
  - Source detecting events disseminate exploratory data across the network
  - Sink having corresponding interest reinforces one of the paths

Directed Diffusion: Design Choices

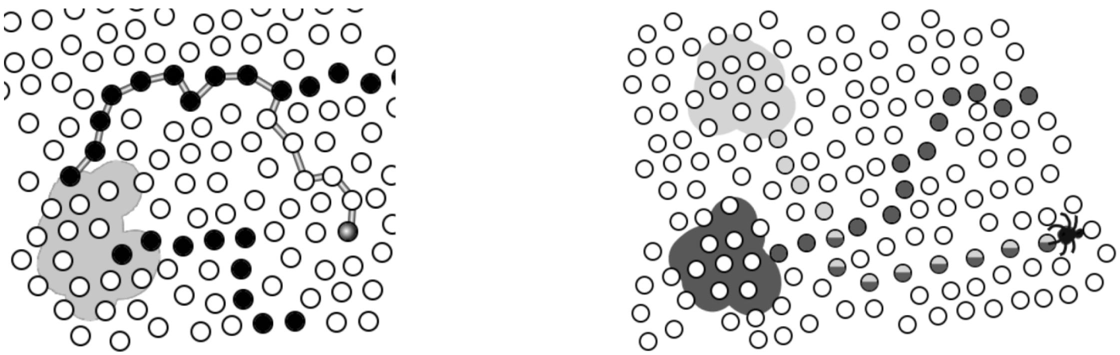
Diffusion Element	Design Choices
Interest Propagation	• Flooding • Constrained or directional flooding based on location • Directional propagation based on previously cached data
Data Propagation	• Reinforcement to single path delivery • Multipath delivery with selective quality along different paths • Multipath delivery with probabilistic forwarding
Data Caching and Aggregation	• For robust data delivery in the face of node failure • For coordinated sensing and data reduction • For directing interests
Reinforcement	• Rules for deciding when to reinforce • Rules for how many neighbors to reinforce • Negative reinforcement mechanisms and rules

Rumor Routing

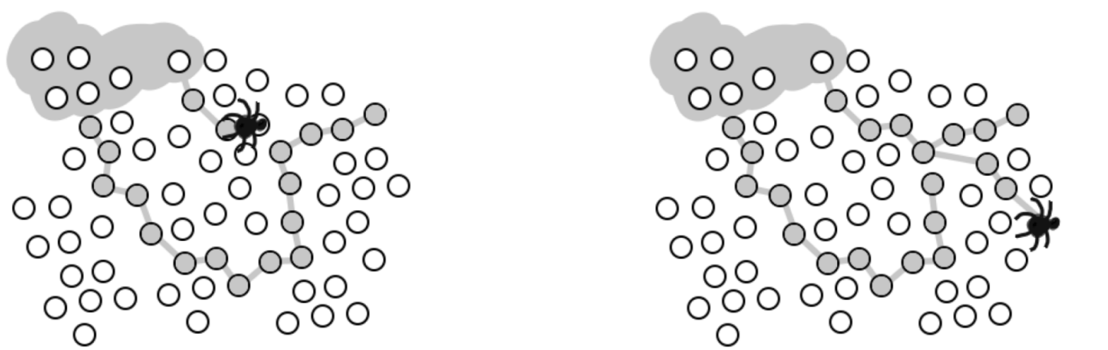
- **Agent-based algorithm**
  - A compromise between query flooding and event flooding
    - Spread information from both: sink and sources
    - Use only linear (straight) paths to preserve energy
  - Long-lived messages, called agents, circulate in the network in order to find shortest paths
  - Agents inform other sinks about events
  - Routes are not optimal
- **Disadvantages**
  - No delivery guarantee
  - Performance depends on topology

Rumor Routing

- Path creation

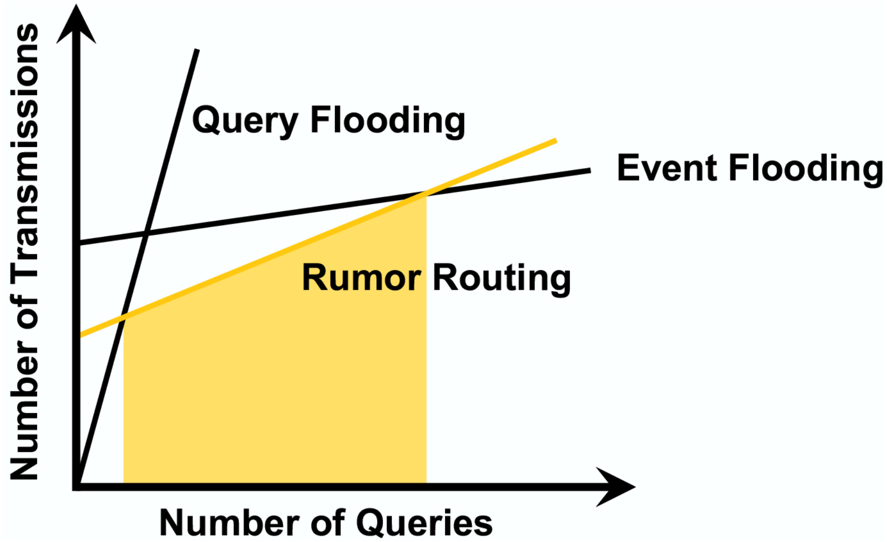


- Path optimization



Rumor Routing

- **Query flooding**
  - A node interested in an event floods the network
  - Transmission for n nodes: $Q \times n$ (Q number of queries)
- **Event flooding**
  - A node sensing an event floods the network
  - Transmission for n nodes: $E \times n$ (E number of events)



LEACH

- **Low-Energy Adaptive Clustering Hierarchy**
- **Goal**
  - Minimize energy dissipation in SNs
  - All nodes consume a similar amount of energy
- **Architecture**
  - Hierarchical protocol
  - Select random nodes as cluster heads
  - Periodic reselection of CHs (cluster heads) after a steady phase

LEACH II

- **Works in Rounds**
  - Short set-up state
  - Long steady state
- **Set-Up Phase:**
  - Advertisement (I am CH)
  - Cluster Set-Up (I am your CH)
  - Schedule Creation (This is your slot)
- **Steady-State Phase:**
  - Data Transmission using TDMA

LEACH III

- **Communication**
  - Every node uses the same channel
  - Different clusters use different CDMA codes
  - Code chosen randomly
  - Only CHs communicate with sink
- **Hierarchical Clustering**
  - Basic LEACH protocol is a 1 hop protocol
  - Extension possible

LEACH IV

- **Set up phase**
  - Each node randomly selects a number between 0 and 1
  - Number is less than a threshold value $T(n)$ ➡ node becomes CH
  - A node advertises its CH role
- **Selection of CHs**
  - $T(n) = \begin{cases} P/(1 - P \times (r \bmod (1/P))) & \text{if } n \in G \\ 0 & \text{otherwise.} \end{cases}$
  - P: desired percentage of CHs
  - G: nodes which were not CHs in the last $1/P$ rounds
  - r: current round

LEACH: Disadvantages

- **“Hot Spot” Problem**
  - Nodes on a path from an event-congested area to the sink may drain
- **Stationary Sink**
  - May be unpractical
- **1 hop neighbors**
  - Basic algorithm assumes any node can communicate with sink but: extensions are possible

TEEN

- **Threshold sensitive Energy Efficient Network protocol**
  - Reactive, event-driven protocol for time-critical applications
  - A node senses the environment continuously, but turns radio on and transmits only if the sensor value changes significantly
  - Save energy if data is not critical
- **CH sends hard and soft thresholds**
  - Hard threshold: A member only sends data to CH only if data values are in the range of interest
  - Soft threshold: A member only sends data if its value changes by at least the soft threshold
- **Hierarchical clustering**

TEEN Discussion

- **Advantages**
  - Good for time-critical applications
  - Less energy consumption compared to proactive approaches
  - Hard and soft threshold can be adapted depending on applications
- **Disadvantages**
  - Inappropriate for periodic monitoring such as habitat monitoring
  - Ambiguity between packet loss and unimportant data (indicating no drastic change)

APTEEN

- **AdaPtive Threshold sensitive Energy Efficient Network protocol**
  - Extends TEEN to support both periodic sensing & reacting to time critical events
  - In contrast to TEEN a node must sample and transmit a data if it has not sent data for a time period equal to CT (count time) specified by CH
- **Compared with LEACH and TEEN**
  - APTEEN consumes less energy than LEACH but more than TEEN
  - Network lifetime: TEEN ≥ APTEEN ≥ LEACH
- **Drawbacks of TEEN and APTEEN**
  - Overhead and complexity of forming clusters in multiple levels and implementing threshold-based functions

SPIN

- **Sensor Protocols for Information via Negotiation**
  - Communicating raw sensor data is expensive but meta-data about sensor data is not
- **Extend lifetime of the system**
  - Nodes need to monitor and adapt to changes in their own energy resource
- **Solves flooding disadvantages!**

SPIN: Metadata

- **Completely describe the data**
  - Must be smaller than the actual data
  - If data is different, their meta-data must differ
  - Metadata is application specific
  - Application has to interpret and synthesize its metadata
- **SPIN messages**
  - ADV: a node A advertises data
  - REQ: an interested node B requests this data
  - DATA: the node A sends the actual data to node B

SPIN-1

- **3-Stage Handshake Protocol**
  - Needs knowledge about single-hop network neighbors
- **Adaptation for lossy networks**
  - Compensate for lost ADVs by re-advertising periodically
  - Compensate for lost REQ/DATA by re-requesting after fixed time
- **Adaptation for mobile networks**
  - Topology changes trigger updates to node neighbor lists
  - A node's neighbor list changes: re-advertise all its data

SPIN-2

- **Energy-conservation**
  - Incorporate low-energy-threshold
  - Works as SPIN-1 when energy level is high
  - Reduce participation of node when approaching low-energy-threshold
    - When node receives data, it only initiates protocol if it can participate in all three stages with all neighbor nodes
    - When node receives advertisement, it does not request the data
  - Caveat: node still exhausts energy below threshold by receiving ADV or REQ messages

Localization

Need for Localization

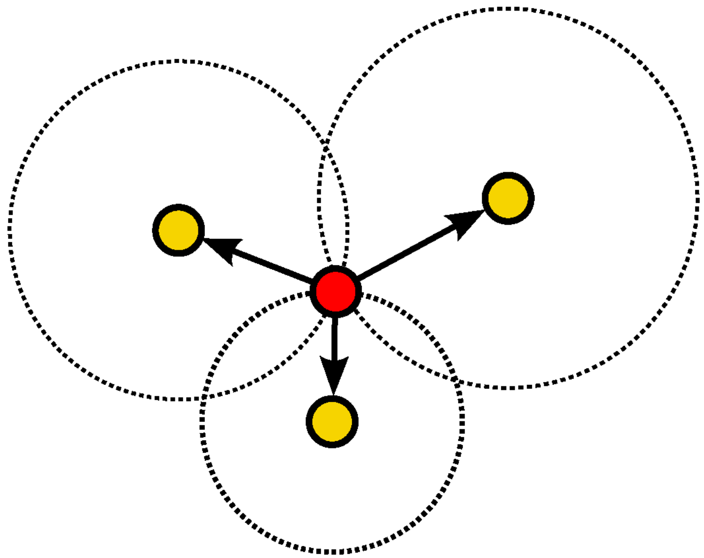
- **Localization**
  - Means for a node to determine its physical position.
- **Why important?**
  - Increase the use of sensor readings!
  - Essential in some communication protocols
- **Why not locate nodes during deployment**
  - Large number of nodes
  - Mobile nodes
  - Air-drop, hostile environment
- **Limits of GPS**
  - Indoor environments, under foliage, next to high-rise buildings
  - Cost in terms of hardware and energy expenditure

Range-based methods

- **Idea**
  - Absolute point-to-point distance estimates
  - Angle estimates
- **Techniques**
  - Received Signal Strength Indicator (RSSI)
  - Time of Arrival (TOA)
  - Time Difference of Arrival (TDOA)
  - Angle of Arrival (AOA)

Localization by Landmarks

- **Landmarks**
  - Landmark: a node that knows its own location
- **Atomic multilateration**
  - Compute a node's location from 3 or more landmarks using distances
  - Least-square technique for *n* landmarks nodes to improve precision



- **How do we compute the distance?**

RSSI

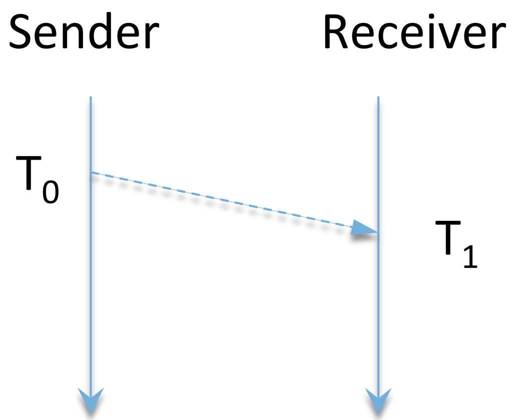
- **Received Signal Strength Inverse (RSSI)**
  - Inverse power of distance $O(1/r^{\alpha})$, but: imprecise
- **Path loss model**

$$P_{RX} = c \times \frac{P_{TX}}{d^{\alpha}}$$

  - Simple, but unreliable due to inaccurate range estimates
  - Fading, interference, position of antenna

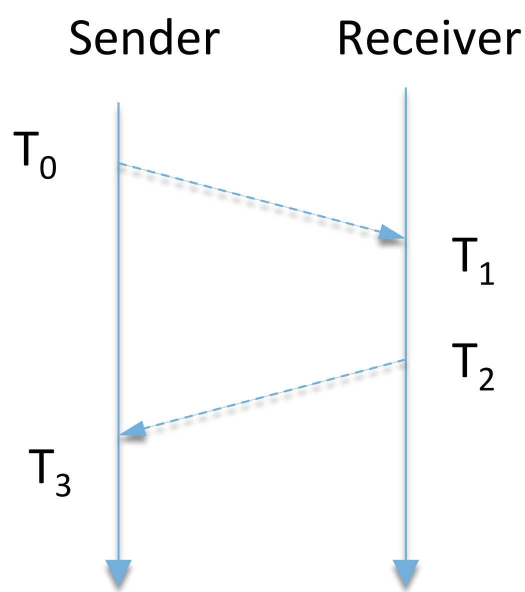
TOA: One-way Delay

- **Time On Arrival (TOA)**
  - Time synchronized sender and receiver
  - Distance = $(T_1 - T_0)$ x Speed
  - Since speed is large, distance cannot be short for radio waves
- **Acoustic TOA**
  - Accuracy is about 10cm
  - Range is tens of meters



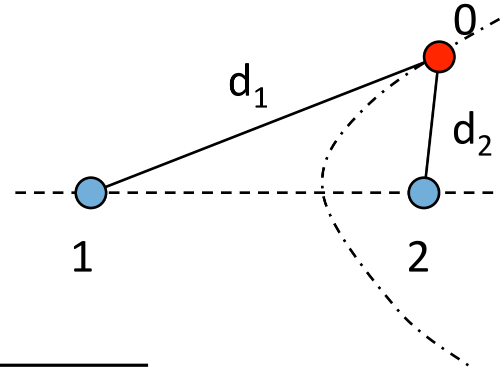
TOA: Round-trip Delay

- **Round trip**
  - No time synchronization required
  - Computing latency affects estimation accuracy
- **TDOA (Time Difference on Arrival)**
  - Use two receivers and measure time difference to estimate the difference in distance



TDOA: Same Frequency

- Coordinated senders
  - Time difference of arrivals translated to distance difference

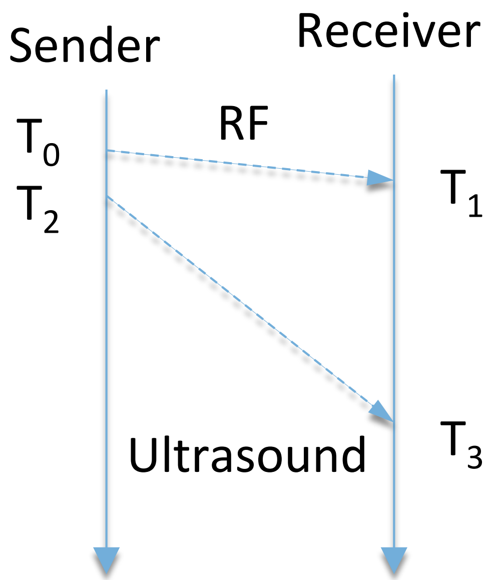


$$\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} - \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2} = \nabla T \times Speed$$

- Problem: calibration

TDOA: Different Frequencies

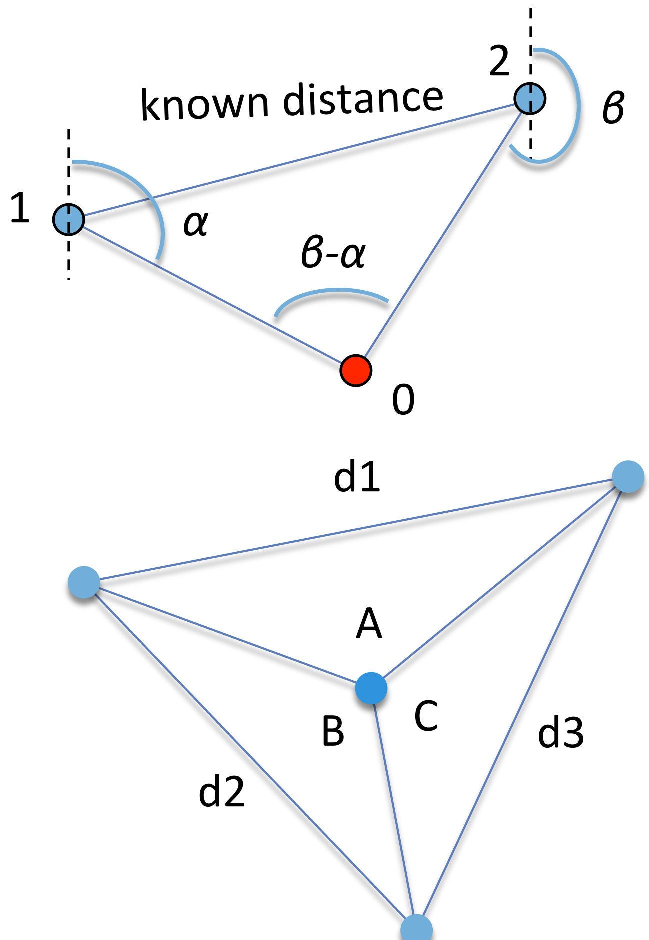
- **Idea**
  - Use two frequencies
  - Speed of wireless signal: s_{RF}
  - Speed of ultrasound: s_{US}
  - $D = ((T_3 - T_1) - (T_2 - T_0)) \times (s_{RF} - s_{US})$



- **Problem**
  - Hardware cost

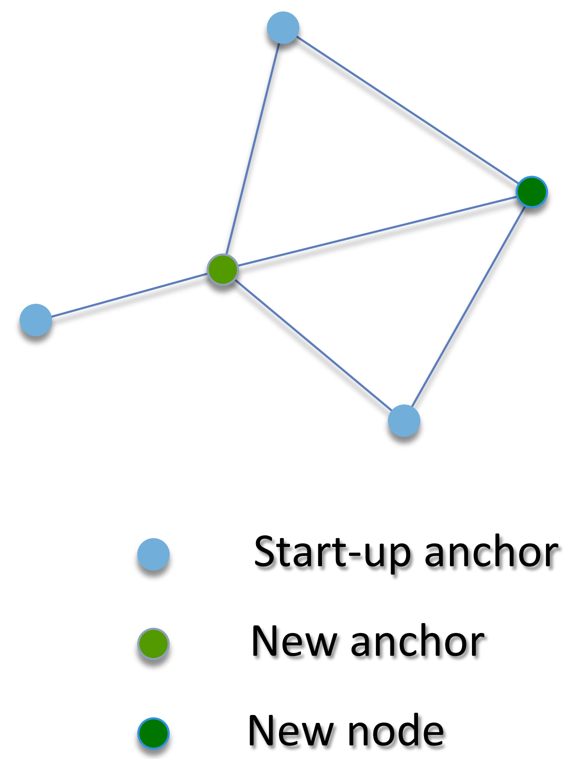
AOA: Angle on Arrival

- **Use an antenna array**
  - Estimate AOA of anchors
- **Problem**
  - Unrealistic for most of WSN applications due to complex hardware and AOA estimation algorithms



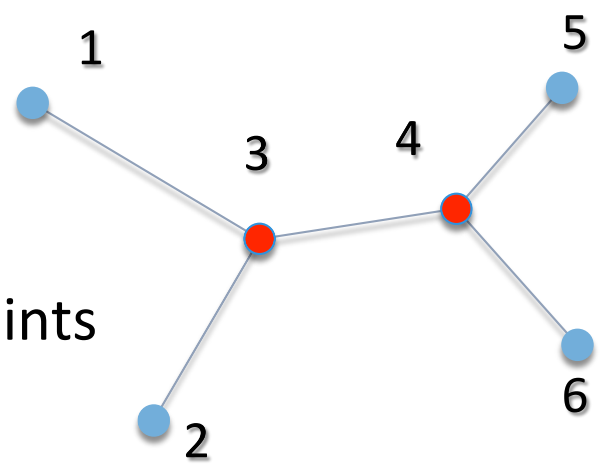
Iterative Multilateration

- **Assumption**
  - Some nodes can hear at least three anchors (to perform triangulation) but not all of them
- **Idea**
  - Nodes recursively compute position estimates and spread position information
- **Problem**
  - Errors accumulate



Collaborative Multilateration

- **Problem**
  - 2 nodes cannot communicate with 3 landmarks but with 2



- **Idea**
  - Collaborate and use all available measurements are used as constraints
  - Solve for the positions of multiple unknowns simultaneously
  - This is a non-linear optimization problem

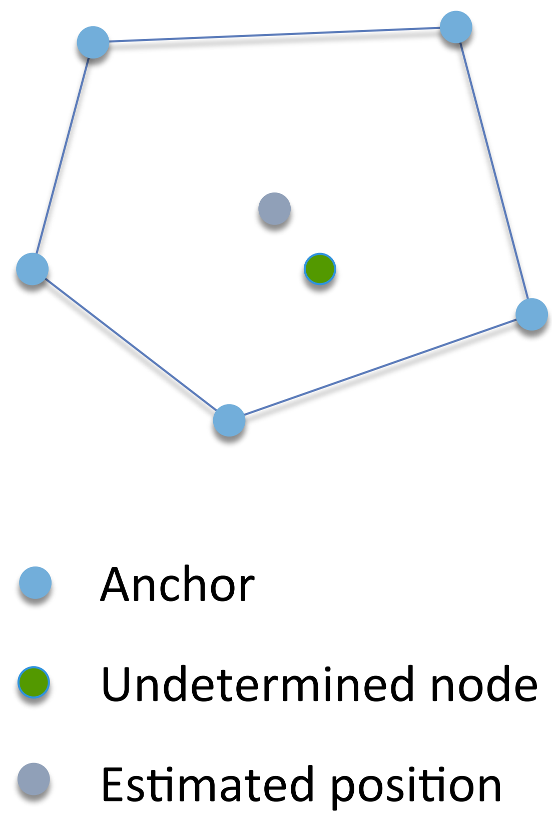
Range-free Methods

- **Assumption**
  - No absolute range estimates are used
- **Advantage**
  - Normally more cost-effective than range-based methods
- **Disadvantage**
  - Normally less accurate than range-based methods

Centroid Approach

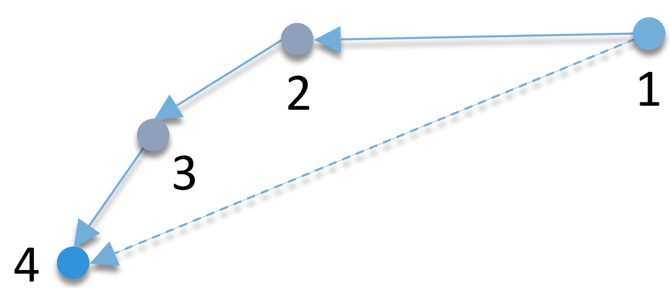
- **Centroid formula**
  - Estimate local location based on anchors' positions

$$(x, y) = (\frac{\sum_i^N x_i}{N}, \frac{\sum_i^N y_i}{N})$$



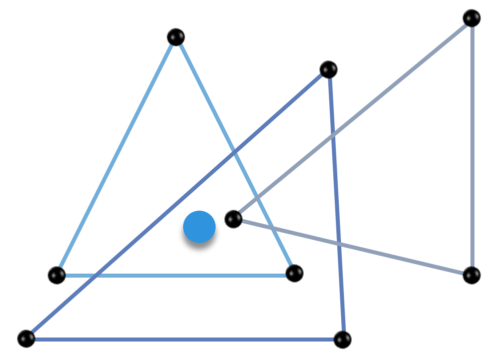
DV-Hop (Distance Vector Hop)

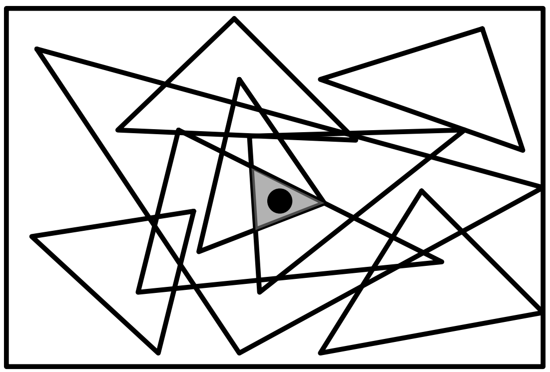
- **Idea**
  - Anchor locations are flooded through the network
  - Use shortest hop distance between nodes
  - The hop distance approximate the Euclidean distance
  - Distance = Hops * Avg_hop_len
  - Average hop length can be obtained online or offline
  - Apply trilateration after estimating more than 3 distances



APIT (Approximate Point in Triangle)

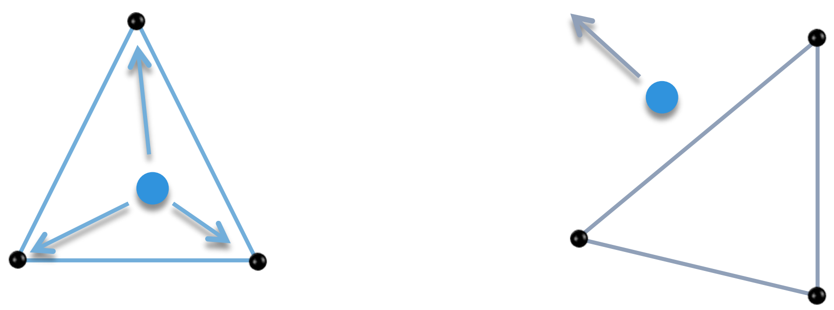
- **Idea**
  - APIT uses an area-based approach
  - Anchors define triangular regions
  - Test whether a node is inside or outside a triangle
- **Location of a node**
  - Intersection area of all the triangles which contain the node





APIT III

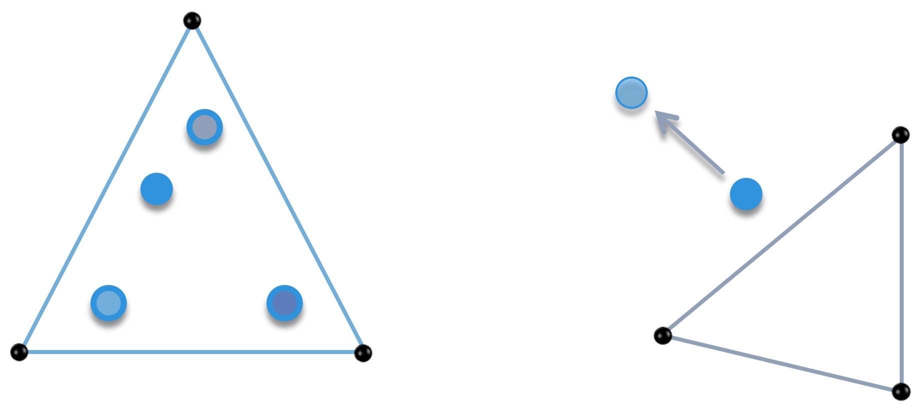
- **PIT theory**
  - *If there is a direction in which M is moves away from points A, B, and C simultaneously, then M is outside of ΔABC; otherwise, M is inside ΔABC.*



- **In practice**
  - Nodes cannot move!
  - How do we determine a direction?
  - Exhaustive test on all directions is not possible

APIT IV

- **Use signal strength**
  - If no neighbour of *M* is further from/closer to all three anchors *A*, *B* and *C* simultaneously, *M* assumes to be inside the triangle
  - Otherwise M assumes to be outside this triangle



APIT V

- **Neighboring nodes**
  - Each node maintains a table of anchor ID, location & signal strength

	(X,Y)		SS
A	20	20	1mv
B	45	31	2mv
C	23	56	3mv

Node M

	(X,Y)		SS
A	20	20	2mv
B	45	31	3mv
C	23	56	1mv

Node 1

  - Nodes exchange anchor tables with the neighbors

	(X,Y)		MySS	SS1	.....	SSn
A	20	20	1mv	2mv		6mv
B	45	31	2mv	3mv		7mv
C	23	56	3mv	1mv		7mv

Node M

APIT VI

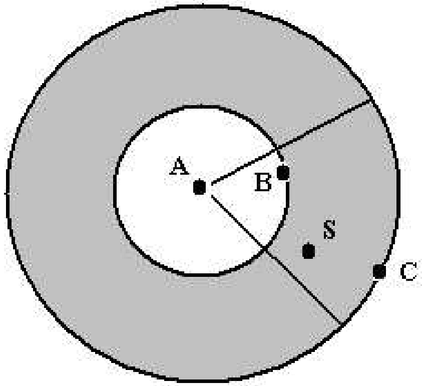
- **Main algorithm**
  - Receive beacons (X_i, Y_i) from N anchors
  - N anchors form $\binom{N}{2}$ triangles **T**
  - For each triangle $T_i \in \mathbf{T}$:
    - If Point-In-Triangle-Test(T_i) == True:
      - InsideSet = InsideSet $\cup \{T_i\}$
  - Position = Center of Gravity $(\{\cap T_i \mid T_i \in \text{InsideSet}\})$;

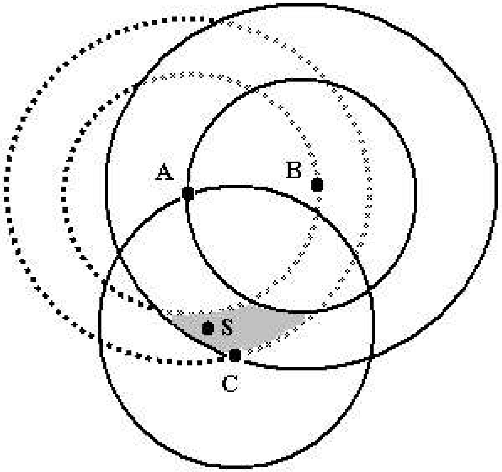
Centroid vs DV-Hop vs – APIT

	Centroid	DV-Hop	APIT
Accuracy	Fair	Good	Good
Node Density	> 0	> 8	> 6
Anchor	> 10	> 8	> 10
Anchor to Node Ratio	> 0	> 0	> 3
Degree of Irregularity	Good	Good	Good
GPS Error of Anchors	Good	Good	Good
Overhead	Smallest	Largest	Small

ROCRSSI

- **Ring Overlapping based on Comparison of RSSI**
  - Anchor A is sender
    - If B’s RSSI < S’s RSSI < C’s RSSI
    - Then S is in ring
  - ROCRSSI only compares the relative strength of RSS
  - Compute ring for each anchor S can hear
  - Center of gravity of intersection of rings is S’s position





Location Verification

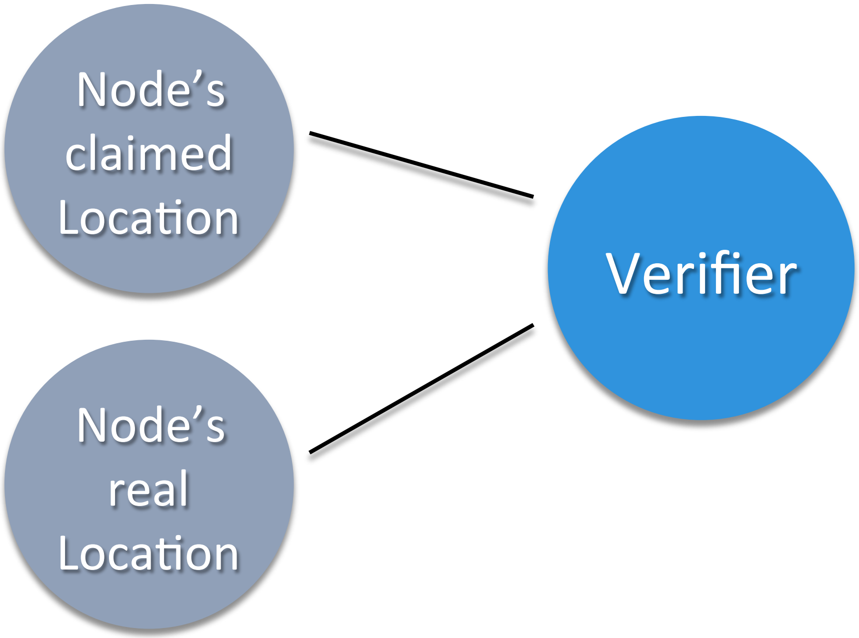
- **How to deal with malicious nodes that lie about their location?**
- **Sample attack scenario**
  - Pretend to be close to the sink
  - Attract many packets
  - Drop some or all of them
  - DoS attacks for geographic routing protocols (see later in the lecture)

SerLoc

- **Approach**
  - Node i claims its location is (x, y)
  - Node i needs to send (x, y) a location verification request message to a nearby verifier
  - A verifier can be a normal sensor node
  - The verifier sends a random nonce to node i and starts its clock
  - Node i has to immediately return the challenge through both radio and ultrasonic channels
- **Verifier**
  - Measures the time for node i returning the challenge
  - Take the difference between the radio and ultrasonic signal propagation
  - Verifies the claimed location based on the measurement

Weaknesses of SerLoc

- Requires extra hardware, i.e., ultrasonic channel
- Valid nodes may respond late due to backlog
- Not location verification but range verification!



Research Issues

- **Most localization work is mathematical and evaluated via (high level) simulations**
  - More realistic work is needed
- **Indoor localization is harder**
  - Look at CodeBlue project at Harvard
- **Location verification**
  - Cannot trust sensors
- **Secure localization**
  - Cannot trust anchors

Geographic Routing

Location Awareness

- **Assumptions**
  - Each node knows its location
  - Each node knows the location of its (1-hop) neighbors
  - The location of a destination node is known
  - Each node can store a constant amount of routing information
- **Advantages**
  - No route discovery necessary
  - No maintenance of routes necessary
  - Facilitates *geocasting*, i.e., delivery of packets to all nodes in a region

Location Services

- Idea: map an address to a node location

- **Viable solutions?**
  - One central location server
  - Every node is a location server
- **Distributed location service**
  - Robust to single node failures
  - Spread load uniformly among nodes
  - Locality-preserving

Grid Location Service (GLS)

- **Idea**
  - Nodes act as a location server in their neighborhood
  - Nodes are hierarchically organized based on a quad-tree
- **Assumptions**
  - Each node has a unique ID, e.g. its MAC address
  - All IDs are distinct and ordered
- **Mapping**
  - GLS provides a mapping from node IDs to node locations

GLS: Setting up a Hierarchy I

- **Level 1 (leaf) tiles**
  - Size of leaf tile: all nodes are in communication range
  - A node *N* can act as a location server for itself and all other nodes in that leaf tile
  - *N* selects three nodes in the three sibling leaf tiles that act as a location server for *N*
- **Level 2 tiles**
  - Four sibling leaf tiles form a tile *T* of level 2
  - *T* is the unique level 2 tile containing *N*
  - *N* selects three nodes in the three sibling level 2 tiles

GLS: Setting up a Hierarchy II

- **Recursion: level *k* tiles**
  - Four sibling level k-1 tiles form a tile of level k
  - Each level k-1 tile is only part of a single level k tile
  - A node is located in exactly one tile of each level
- **Properties**
  - Depth of the tree for n nodes: $O(\log n)$
  - Number of location servers for node N: $O(\log n)$

GLS: Setting up a Hierarchy III

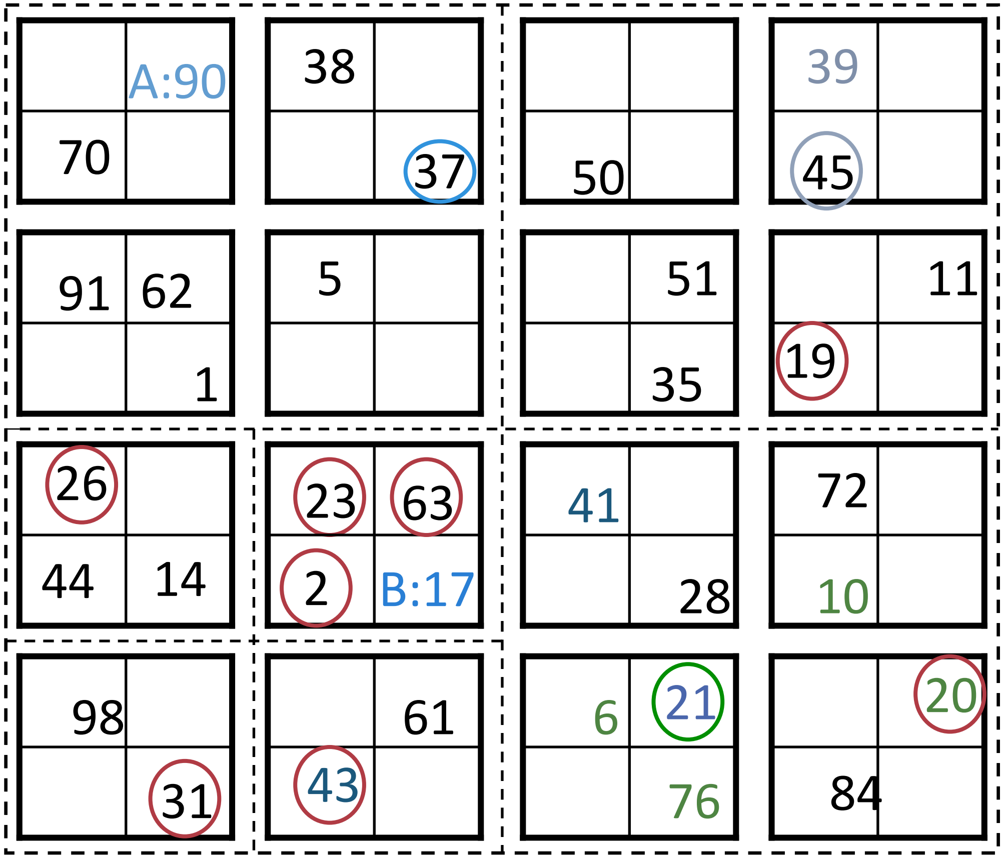
- **Determining a location server**
  - Avoid that all nodes choose the same node
  - The location server for *N*: node with the least ID greater than *N*'s ID (also called the *closest* node)
  - Wrap around if none available: node with the smallest ID
- **How to search for a location server?**
  - Node *N* uses geographic forwarding and sends a packet including its position to the selected tile on level *k*
  - First node in that tile searches for the closest location server (node) for *N*

Communicating with GLS

- **Communication between nodes A and B**
  - *A* sends a request to the node *N* that has the least ID ≥ than *B*'s ID and is known to *A*
  - If *B*'s location is known to *N*: geographic forwarding of *A*'s packet to *B*, *B* can reply as *A*'s location is part of message
  - If not: *N* repeats the algorithm and sends the sends a request to the node *N*' that has the least ID ≥ than *B*'s ID and is known to *A*
  - Each iteration moves one level up in the hierarchy
  - Recursion must stop because *B*'s location is known at the highest level

Understanding GLS I

- Each step brings the query to the closest node in a larger square
- Node 21 is a location server for node 6, 10, 20, 76
- Node 45 is a location server for node 39, 41, 43



Understanding GLS II

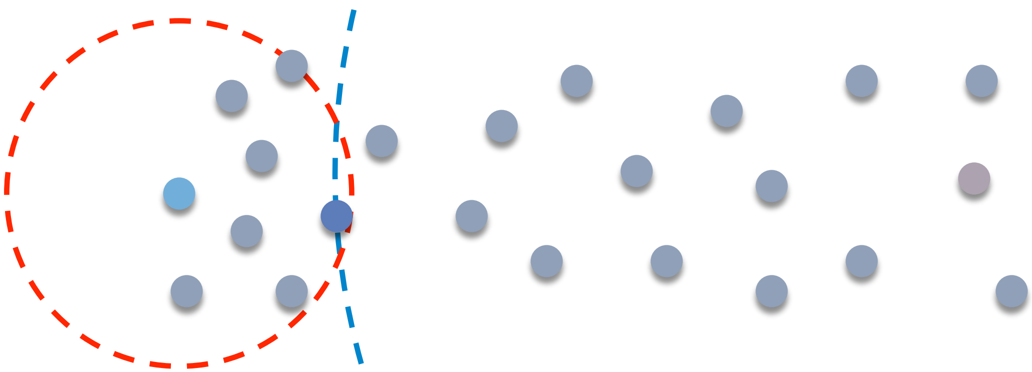
	70,72,76,81 82,84,87 A: 90	1,5,6,10,12 14,37,62,70 90,91 38				19,35,37,45 50,51,82 39	
1,5,16,37,62 63,90,91 70			16,17,19,21 23,26,28,31 32,35 **37**	19,35,39,45 51,82 50		39,41,43 45	
1,62,70,90 91	1,5,16,37,39 41,43,45,50 51,55,61,91 62	1,2,16,37,62 70,90,91 5			35,39,45,50 51		19,35,39,45 50,51,55,61 62,63,70,72 76,81 11
	62,91,98 1				19,20,21,23 26,28,31,32 51,82 35	1,2,5,6,10,12 14,16,17,82 84,87,90,91 98 **19**	
14,17,19,20 21,23,26,87 **26**		2,17,23,63 **23**	2,17,23,26 31,32,43,55 61,62 **63**	28,31,32,35 37,39 41		10,20,21,28 41,43,45,50 51,55,61,62 63,70 72	
14,23,31,32 43,55,61,63 81,82,84 87	2,12,26,87 98 14	1,17,23,63,81 87,98 **2**	2,12,14,16 23,63 B: 17		6,10,20,21 23,26,41,72 76,84 28	6,72,76,84 10	
31,81,98 32	31,32,81,87 90,91 98	12,43,45,50 51,61 55	12,43,55 61	1,2,5,21,76 84,87,90,91 98 6	6,10,20,76 21		6,10,12,14 16,17,19,84 **20**
31,32,43,55 61,63,70,72 76,98 81	2,12,14,17 23,26,28,32 81,98 **31**	12,14,17,23 26,31,32,35 37,39,41,55 61 **43**	2,5,6,10,43 55,61,63,81 87,98 12		6,21,28,41 72 A: 76	20,21,28,41 72,76,81,82 84	

Discussion of GLS

- **Advantages**
  - Each node maintains a small amount of state
  - Querying even works well if location servers fail
- **Cons**
  - Prone to performance degradation due to node failures and high degrees of mobility
  - Fixed size squares: nodes in high density areas have to maintain more state information (more power required)
  - The nodes have to know the quadtree structure in advance

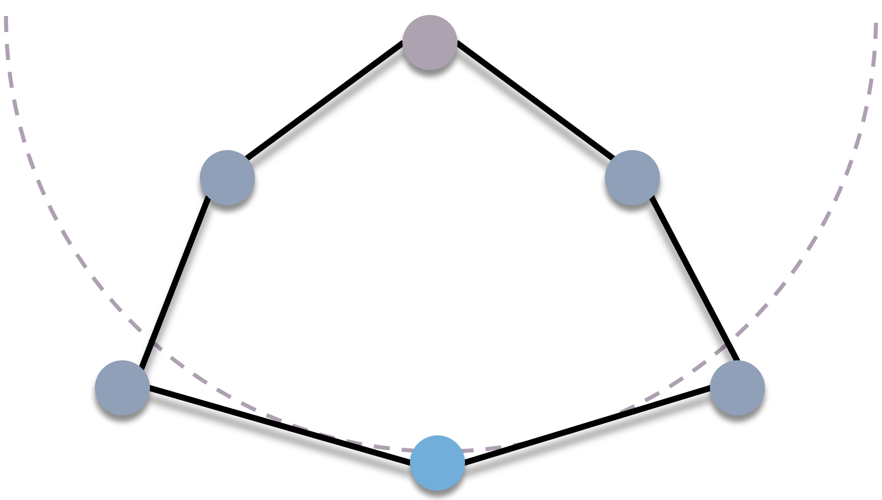
Greedy Forwarding I

- **Local strategy**
  - A node forwards a message to a neighbor node that is “closer” to the destination than itself
  - Repeat until the destination is reached
- **Loop-free**
  - If nodes have consistent location information



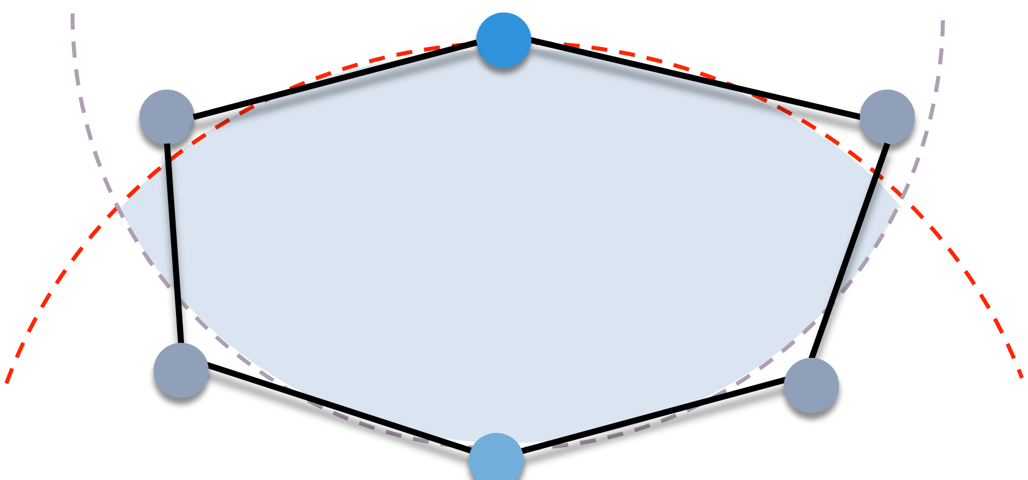
Greedy Forwarding II

- **Similar approaches**
  - MFR (most forward within radius): closest projection
  - DIR/GEDIR: direction
- **Greedy forwarding can fail!**
  - Require recovery strategy



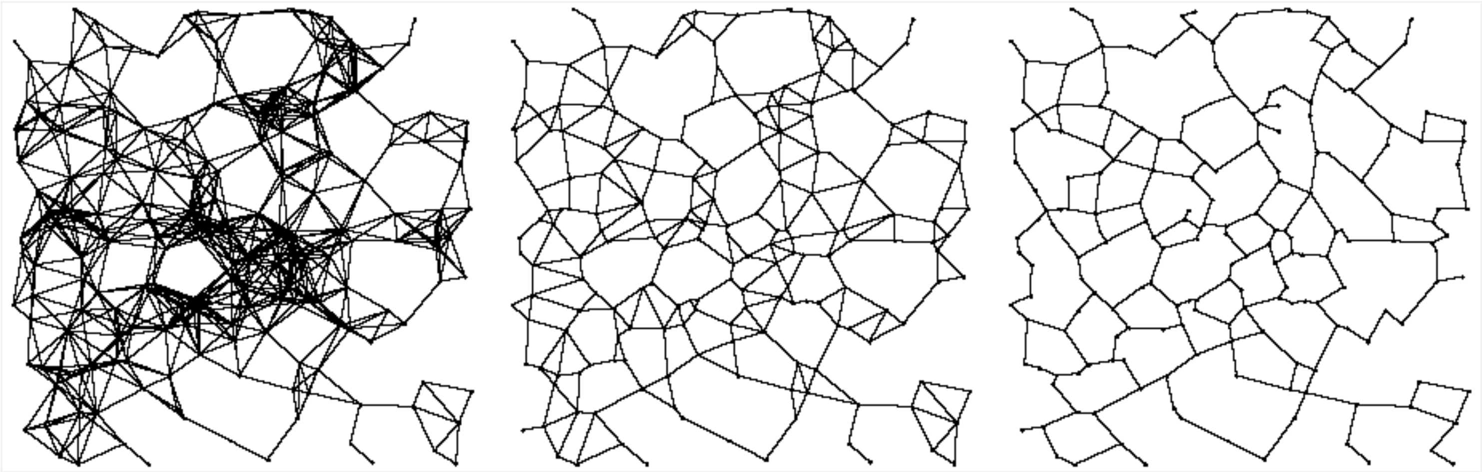
Face Routing

- **Problem**
  - Greedy distance protocols are simple and powerful
  - But: Packets can get stuck in local minima
- **Protocol: right hand rule**
  - Can guarantee delivery
  - Route along the perimeter of voids



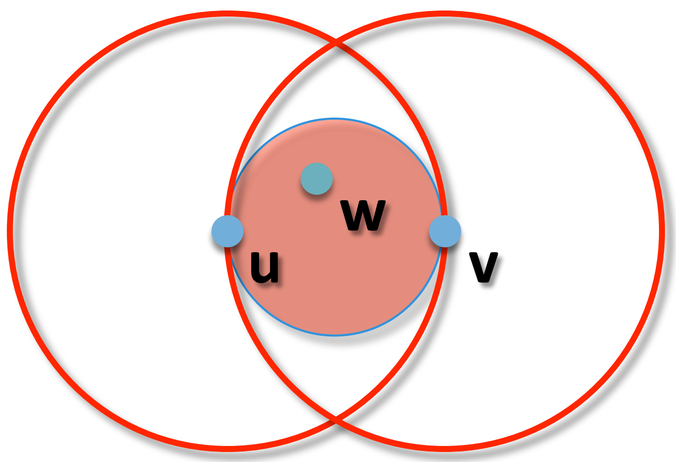
Planarization of a Graph

- **Purpose**
  - Face routing relies on a planarized graph, i.e., no two edges intersect
  - Can be locally computed
  - Gabriel graph & relative neighborhood graph



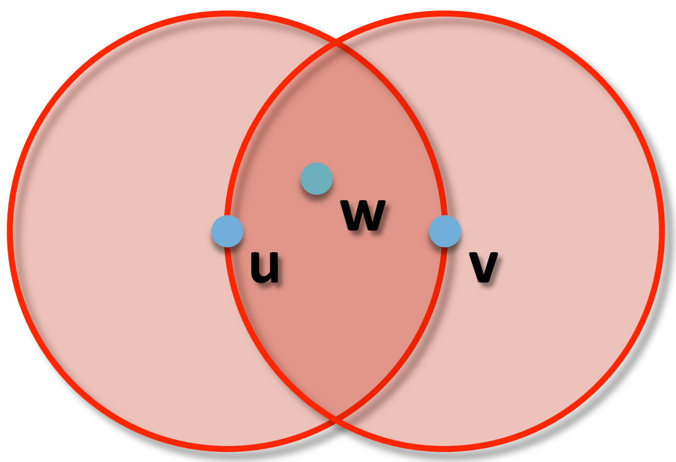
GG: Gabriel Graph

- **Definition**
  - There is an edge (u,v) between two vertices $u, v \in G$ if there is no other vertex $w \in G$ in the circle of diameter <u>uv</u>
  - $\forall w \in G$, $w \neq v$, $w \neq u$: $d(u,v) < (d^2(u,w) + d^2(v,w))^{1/2}$



RNG: Relative Neighborhood Graph

- **Definition**
  - There is an edge (u,v) between two vertices $u, v \in G$ if their distance $d(u,v)$ is less than or equal to the distance of u or v to any other vertex $w \in G$
  - $\forall w \in G$, $w \neq v$, $w \neq u$: $d(u,v) < \max(d(u,w), \mathrm{d}(v,w))$



GPSR

- **GPSR: greedy perimeter stateless routing**
  - Greedy forwarding
  - Use only information of a node's immediate neighbors
  - Compute a planar subgraph of the communication graph (RNG or GG)
- **Perimeter forwarding**
  - If greedy forwarding is not possible
  - Route along the perimeter of a face on the subgraph
  - Use right hand rule
- **Stateless**
  - A router (node) only keeps local topology

GAF I

- **GAF: geographic adaptive fidelity**
  - Location- and emery-aware routing protocol for MANETs
  - Nodes have different roles
- **Hierarchical protocol**
  - Network is partitioned into fixed zones (virtual grid)
  - Each node in a zone is considered equal in terms of cost and position
  - A node in a zone must be able to reach every node of an adjacent zone

GAF II

- **Roles of nodes**
  - In each zone one node will be awake for a period of time while other are asleep
  - The responsible node senses and communicates data for the other nodes in the zone
  - Node have three states: discovering, active, asleep
  - Nodes have to synchronize with other nodes in a zone
- **Orthogonal protocol**
  - GAF can be used in conjunction with another routing protocol

GEAR

- **Geographic and Energy Aware Routing**
  - Heuristics to select neighbors depending on energy levels and distance
- **Nodes use two cost functions**
  - Estimated cost: combination of residual energy and distance to goal
    - N_i is a node and R is the target region
    - $d(N_i, R)$: normalized distance from N_i to the centroid of R
  - e_i is the normalized energy already consumed by node N_i
    - $c(N_i, R) = \alpha\, d(N_i, R) + (1-\alpha)\, e(N_i)$
  - Learned cost: refined cost that accounts for holes in the network

Challenges in WSNs

- **Localization errors**
  - Imprecise measurements
  - Lack of updates
  - Rapid position changes
- **(Self-) Localization techniques**
  - GPS is not available indoors
- **Rapid topology changes**
  - Location service is costly
- **Global behavior from local knowledge**
  - Achieve desired global goal using adaptive localized algorithms
- **Sparse sensor networks**
  - Many algorithms do not perform well if the network is not dense