



Mobile and Sensor Systems

Lecture 5: Ad Hoc and Delay Tolerant Routing

Dr. Cecilia Mascolo

What's in this lecture

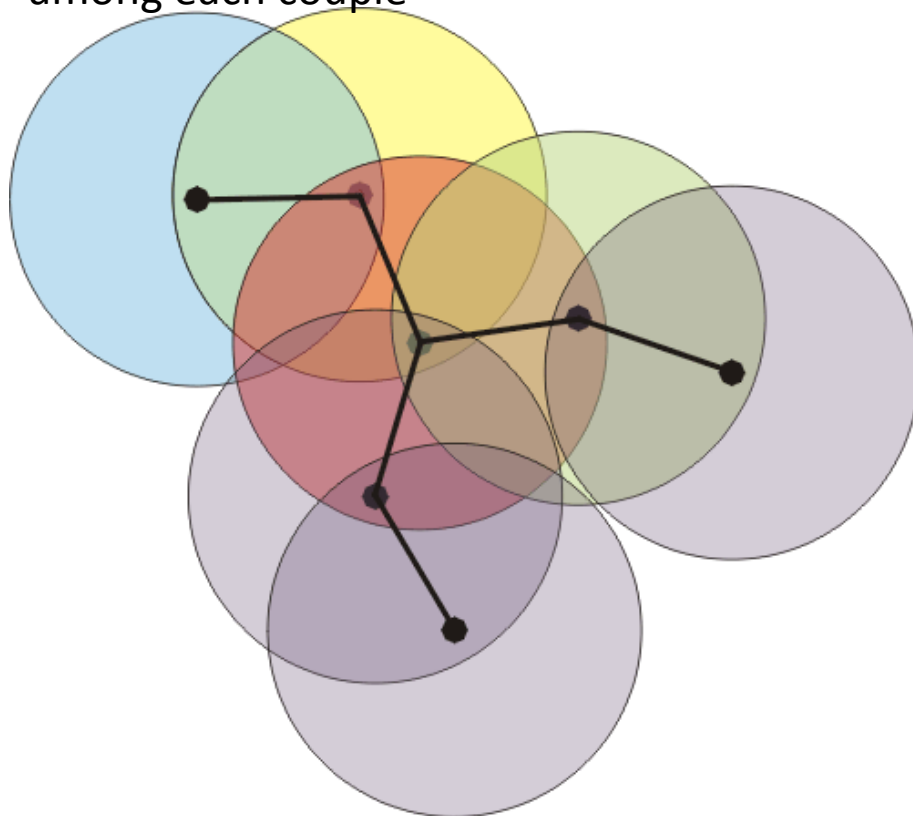


-
- We will describe ad hoc routing protocols
 - We will introduce delay tolerant networks which are disconnected ad hoc networks
 - We will describe delay tolerant routing protocols
 - We will illustrate examples of geographical routing

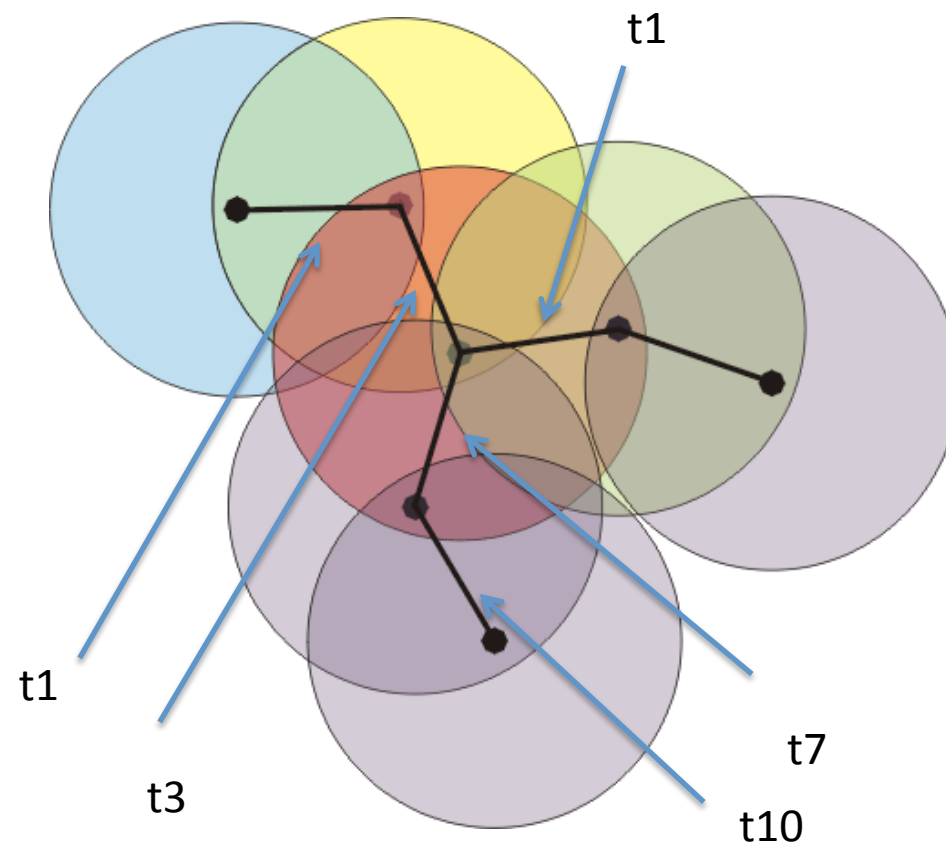
Connected vs Disconnected Ad Hoc Networks



Connected: there is a connected path among each couple



Disconnected: there is no connected path, just sometimes some temporal ones



Routing in Wired/Wireless Networks



- Link State
 - Each node sends its link information to all nodes in the network
 - Small vector to all large number of nodes
 - Dijkstra for shortest path
- Distance Vector
 - Each node sends its table to its neighbours
 - Large vector to small number of nodes
 - Bellman Ford for shortest path

OLSR: Optimized Link State Routing Protocol

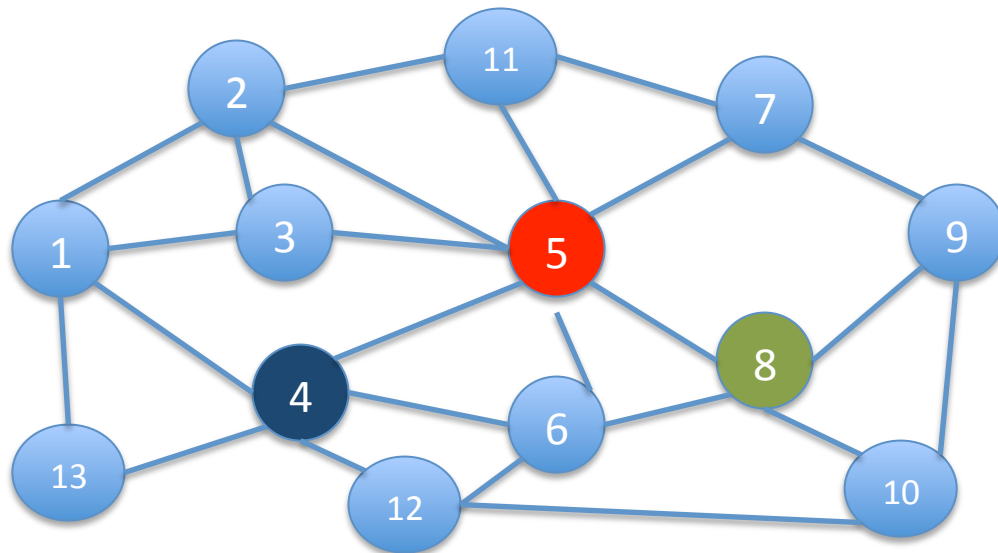


- Proactive
- Hello messages from a node to its neighbours with bidirectional links and list of known neighbours -> learning 2 hop neighbourhood
- Ask a subset of neighbours to forward a node's link state (subset=MPR, Multipoint Relay)
- If node X is in your MPR you are in X's MPR Selector
- Each MPR has a set of MPR Selectors
- Each node sends LS to all its neighbours
- MPR forwards LS of MPR's selectors
- Nodes use this info for routing tables but do not forward

OLSR Example



- Node 5 has selected 4 and 8 as MPR and sends LS to 2,3,4,6,7,8,11
- Nodes 2,3,6,7,11 use this info but do not forward
- Node 4 forwards to 1,6,12,13
- Node 8 forwards to 6,9,10



How are MPR Selected?

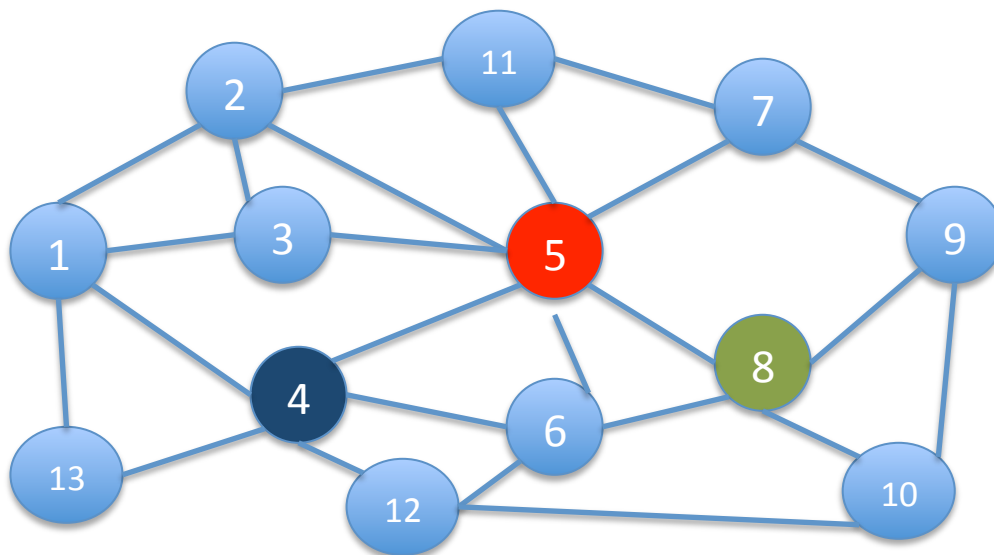


- MPR are arbitrarily selected
- A node can put all its neighbours into a MPR but
 - Not optimized -> lots of duplication
 - Optimal: min set such that all 2-hop neighbours get node's LS
 - Finding optimal MPR is NP complete
 - Heuristics
 - $N1(x)$ =1-hop neighbours
 - $N2(x)$ =2-hop neighbours not covered
 - $MPR(x)$ = empty
 - From $N1(x)$ - $MPR(x)$, select node A that has max connectivity to uncovered nodes (and update $N2(x)$)
 - Add A to $MPR(x)$

Link State forwarding



- Each node maintains a routing table with
 - Node id, next hop, distance
- The table is never forwarded
- Updates on links are forwarded when there is a topology change



Drawbacks of OLSR (and partly of ad hoc protocols)



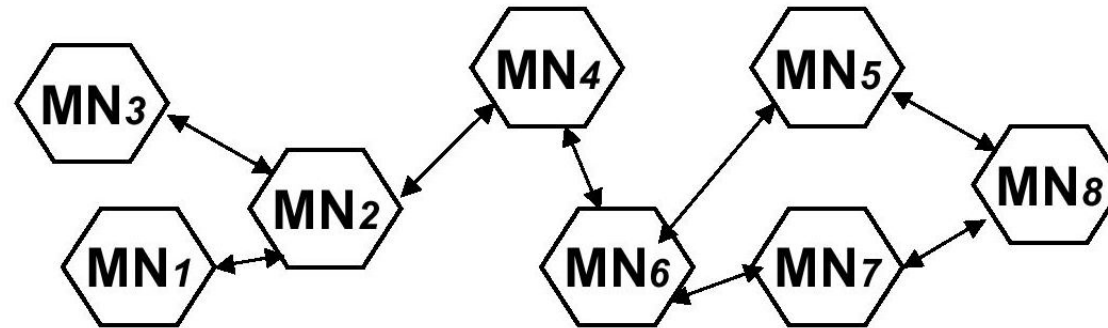
- Assumes a connected network
- Assumes bidirectional links
 - Extensions have been proposed to consider link quality and bidirectionality
- Being proactive means it consumes a lot of resources

Destination Sequenced Distance Vector Routing



- Proactive
- Each node maintains a table with a route to every node
- Each entry of the table has a sequence number assigned by the destination
 - Sequence number, Destination, hops required, next hop

DSDV --- Routing Table at MN4



Dest	NextHop	Metric	DestSequence	InstallTime
MN1	MN2	2	406	
MN2	MN2	1	128	
MN3	MN2	2	564	
MN4	MN4	0	710	
MN5	MN6	2	392	
MN6	MN6	1	076	
MN7	MN6	2	128	
MN8	MN6	3	050	

DSDV routing updates



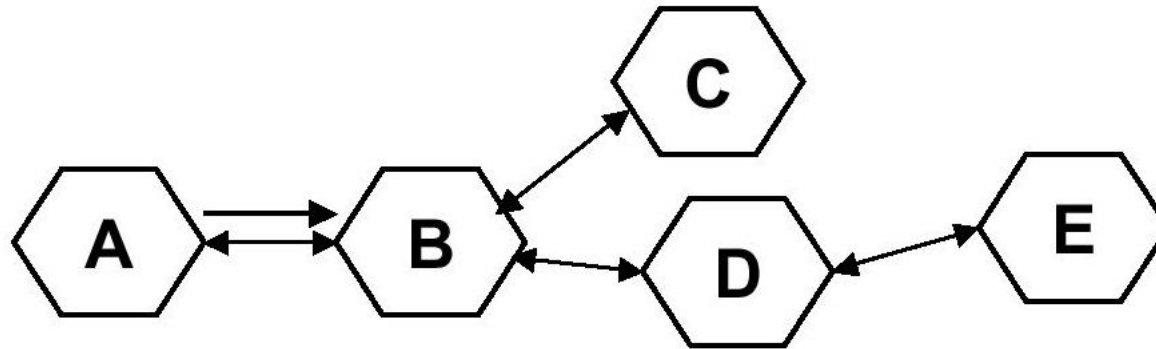
- Each node periodically transmits updates
 - Includes its own sequences number, routing table updates
- Nodes also send routing table updates for important link changes
- When two routes to a destination received from two different neighbors
 - Choose the one with greatest destination sequence number
 - If equal, choose the smaller metric (hop count)

DSDV --- full dump



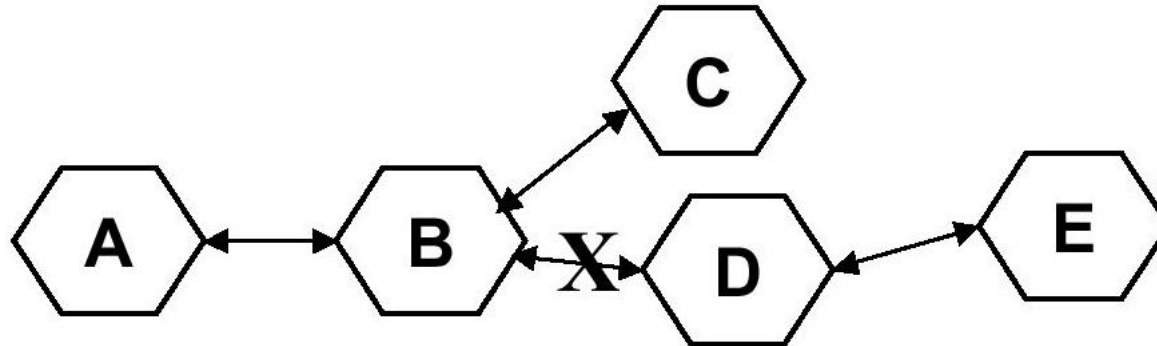
- Full Dumps
 - Carry all routing table information
 - Transmitted relatively infrequently
- Incremental updates
 - Carry only information changed since last full dump
 - Fits within one network protocol data unit
 - If can't, send full dump

DSDV --- link additions



- When A joins network
 - Node A transmits routing table: $\langle A, 101, 0 \rangle$
 - Node B receives transmission, inserts $\langle A, 101, A, 1 \rangle$
 - Node B propagates new route to neighbors $\langle A, 101, 1 \rangle$
 - Neighbors update their routing tables: $\langle A, 101, B, 2 \rangle$ and continue propagation of information

DSDV --- link breaks



- Link between B and D breaks
 - Node B notices break
 - Update hop count for D and E to be infinity
 - Increments sequence number for D and E
 - Node B sends updates with new route information
 - <D, 203, infinite>
 - <E, 156, infinite>

Dynamic Source Routing



- On Demand routing-> Reactive= construct a route only when needed
- Source route=list of routers along the path
- A node S wanting to send checks if it knows the route to the destination D
- If S does not know route to D and sends route request with its ID
- Each node adds itself to the request (compiling a “route record) and forwards to neighbours
- If a node knows the route it appends it and replies back with a route reply [in the worst case D replies back]

Dynamic Source Routing



- The node issuing the route reply places the route record in the reply as well as its known route to destination
- If this node has a path to the source this is followed, or the symmetric path can be followed (if bidirectional links are supported)
- Route Maintenance:
 - When a hop cannot be followed [due to link error] that hop is deleted from the cached route from that node

Issues with Reactive Routing



- If there is high node mobility, reactive routing is not performing well
 - Why?
- Routing overhead is proportional to path length

Zone Routing Protocols (ZRP)



- Zone routing is a hybrid protocol which combines proactive with reactive approaches
- A zone around node N is maintained where routes are collected proactively
- Beyond the zone an inter zone protocol is responsible to determine the routes in a reactive way

Delay Tolerant Routing



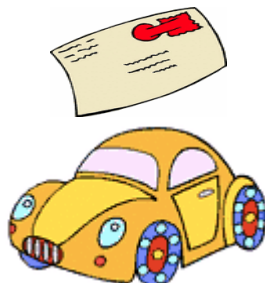
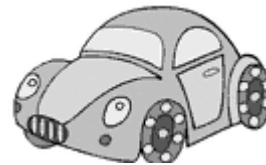
- When the network is not connected ad hoc routing protocols are unable to deliver the messages
- The strategy that works exploits **the fact that nodes can store the messages and forward them later**

Epidemic Routing

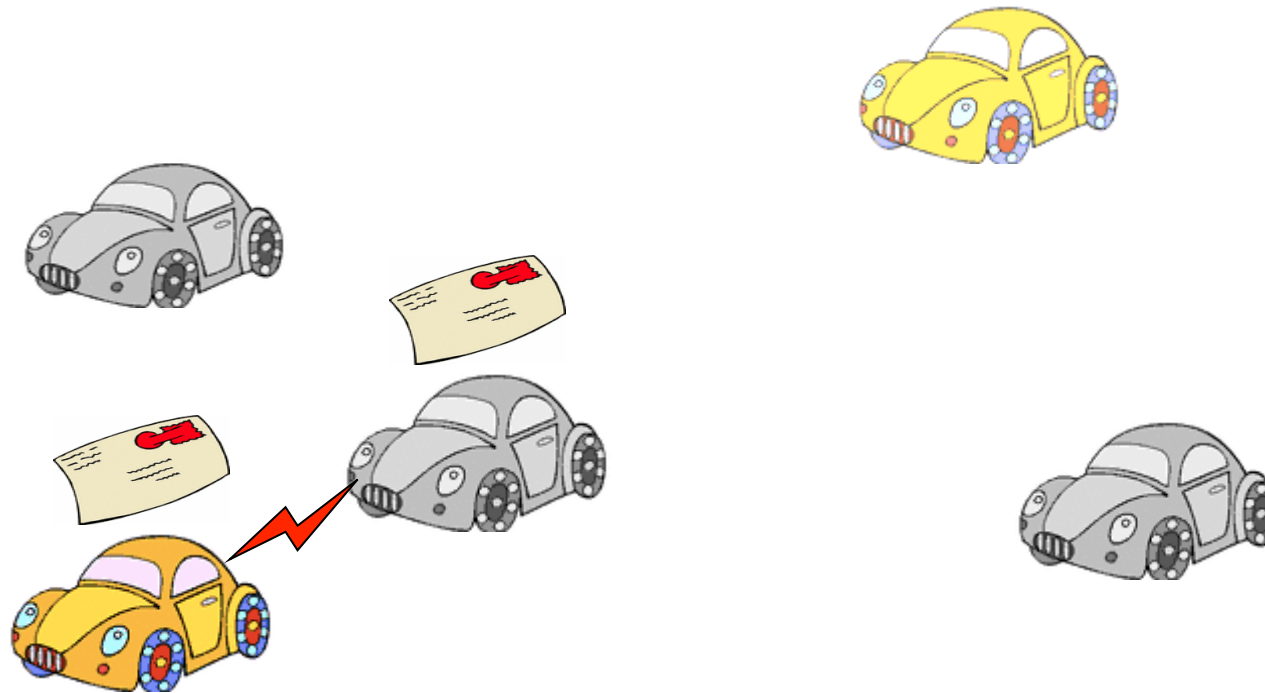


- Vahdat and Becker
- Utilize physical motion of devices to transport data
- *Store-carry-forward* paradigm
 - Nodes buffer and carry data when disconnected
 - Nodes exchange data when met
 - data is replicated throughout the network
- Robust to disconnections
- Scalability and resource usage problems

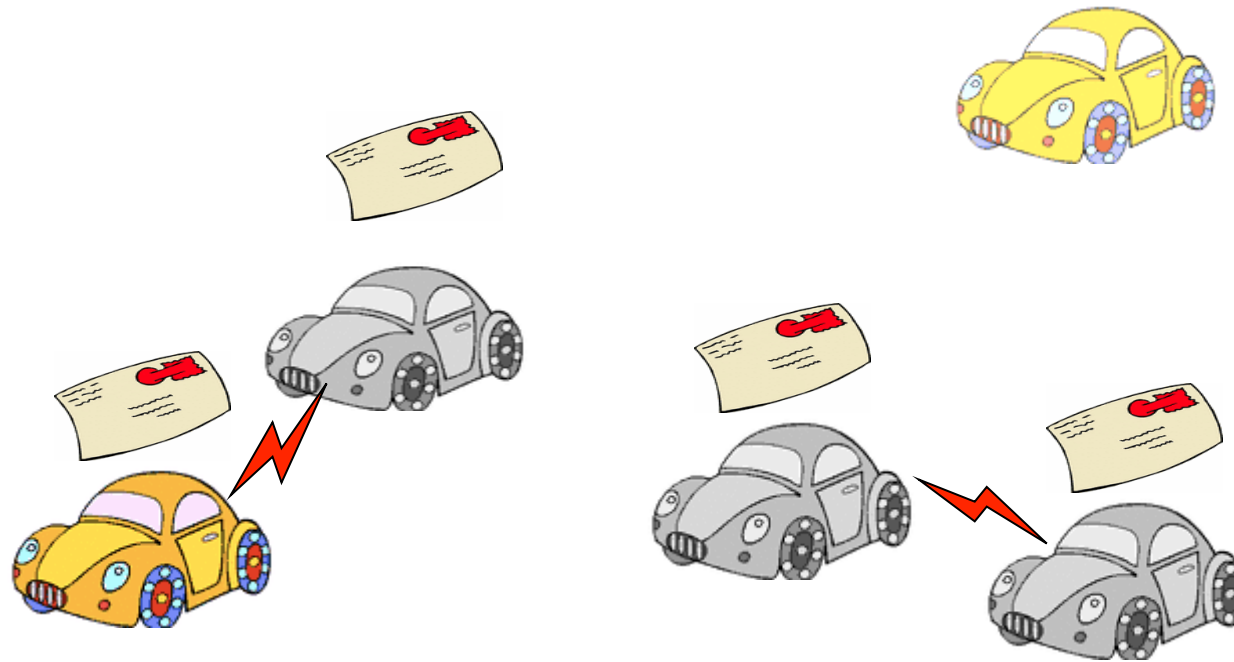
Epidemic Routing



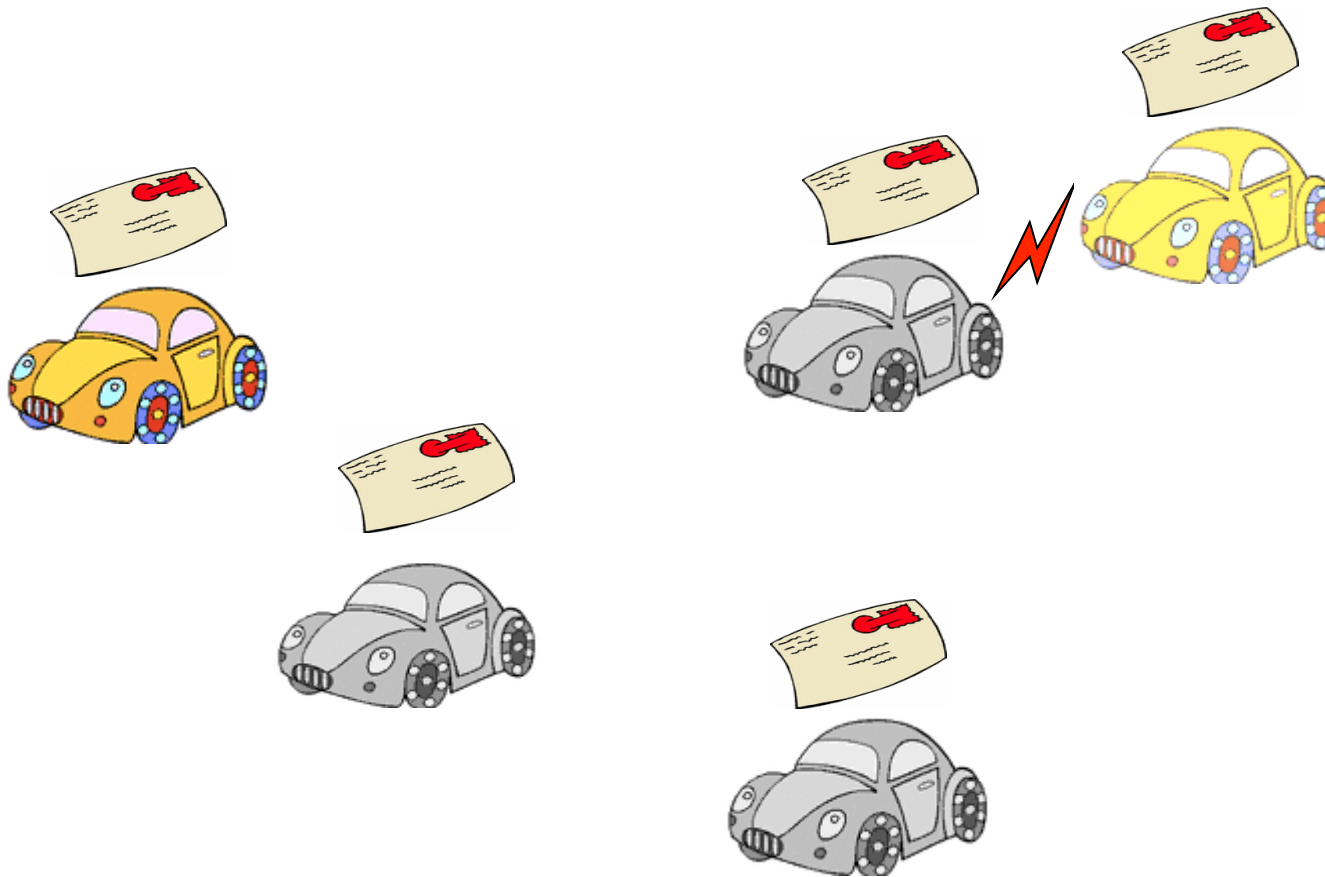
Epidemic Routing



Epidemic Routing



Epidemic Routing



The Trouble with ER



- High delivery but also high overhead!
- Lots of duplications and collision
- Can we do better than this?
 - ...with infinite buffers and bandwidth, not in terms of the delivery but only in terms of overhead!
 - With finite buffers and/or limited bandwidth, if we send less messages around we can do much better!

How can we improve from Epidemic Routing



- Exploit the knowledge on the mobility of the nodes
- Is the mobility deterministic (ie. Always on the same path at same times like busses)? Maybe we can even control the mobility of some of the nodes!
- If not fixed, is it at least predictable?
- If not predictable, random...

What when you do not know the mobility?



- Prediction of mobility techniques need to be applied unless you want to use “epidemic”
- Instead of blindly forwarding packets to all or some neighbors, intermediate nodes estimate the chance, for each outgoing link, of eventually reaching the destination.
- Based on this estimation, the intermediate nodes decide whether to store the packet and wait for a better chance, or decide to which nodes (and the time) to forward.

Context Aware Routing

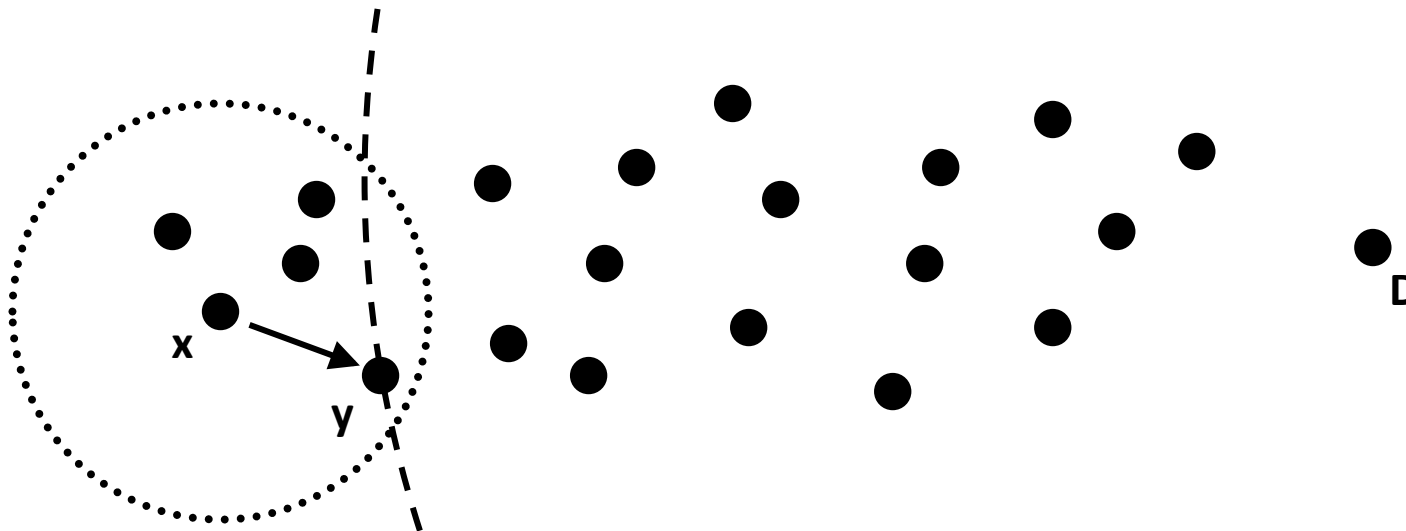


- When a node wants to send a message to another known node in the network it forwards it to the best (carrier) neighbour
- How is the best carrier neighbour chosen?
 - Host mobility
 - Host collocation with destination node
 - Battery?
 - A utility function which weights these various aspects
 - Kalman Filter is used to predict future host collocation with destination based on previous history
- The approach is based on local knowledge only

Greedy Perimeter Stateless Routing (GPSR)



- The algorithm consists of two methods for forwarding packets:
- *greedy forwarding*, which is used wherever possible, and
- *perimeter forwarding*, which is used in the regions greedy forwarding cannot be.



Greedy Forwarding



- Under GPSR, packets are marked by their originator with their destinations' locations.
- As a result, a forwarding node can make a locally optimal, greedy choice in choosing a packet's next hop.
- Specifically, if a node knows its radio neighbors' positions, the locally optimal choice of next hop is the neighbor geographically closest to the packet's destination.
- Forwarding in this regime follows successively closer geographic hops, until the destination is reached.

Greedy Forwarding



- A simple beaconing algorithm provides all nodes with their neighbors' positions: periodically, each node transmits a beacon to the broadcast MAC address, containing only its own identifier (e.g., IP address) and position.
- Position is encoded as two four-byte floating point quantities, for x and y coordinate values.

Greedy Forwarding

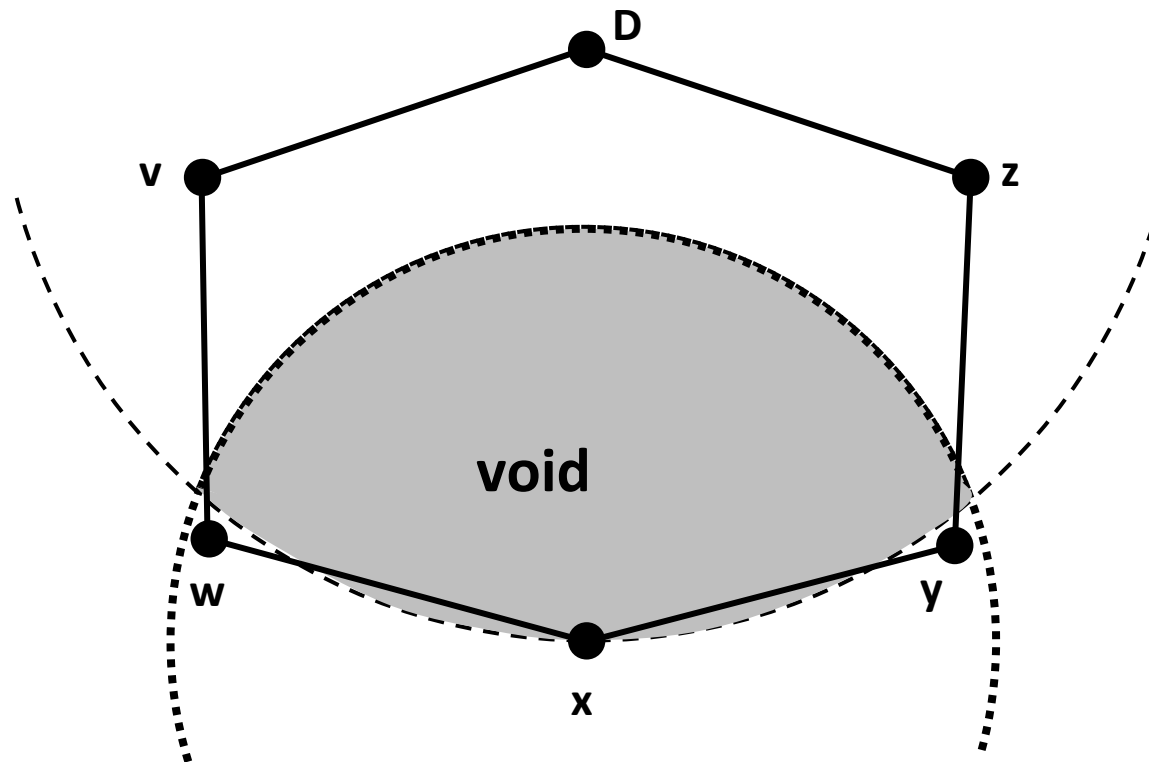


- Upon not receiving a beacon from a neighbor for longer than timeout interval T , a GPSR router assumes that the neighbor has failed or gone out-of-range, and deletes the neighbor from its table.
 - The 802.11 MAC layer also gives direct indications of link-level retransmission failures to neighbors; algorithm interprets these indications identically.

Greedy Forwarding Failure



Greedy forwarding not always possible! Consider:



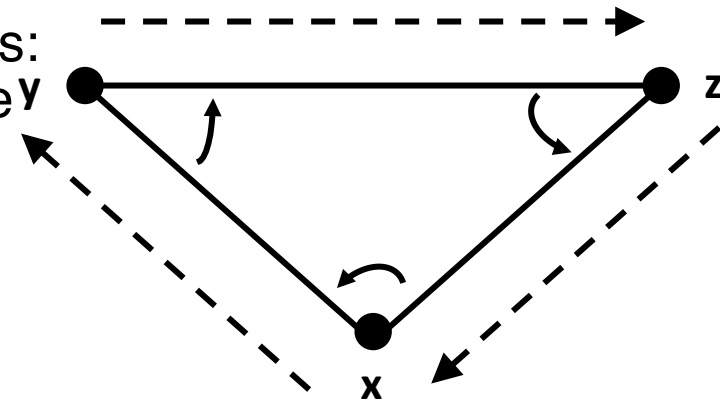
Void Traversal: The Right-hand Rule



Well-known graph traversal: **right-hand rule**

Requires **only neighbors' positions**

- Mapping perimeters by sending packets on tours of them, using the right-hand rule. The state accumulated in these packets is cached by nodes, which recover from local maxima in greedy forwarding by routing to a node on a cached perimeter closer to the destination
- This approach requires the no-crossing heuristic, to force the right-hand rule to find perimeters that enclose voids in regions where edges of the graph cross.
- Caveat: if the graph has cross cutting edges:
 - Remove those with a specific procedure

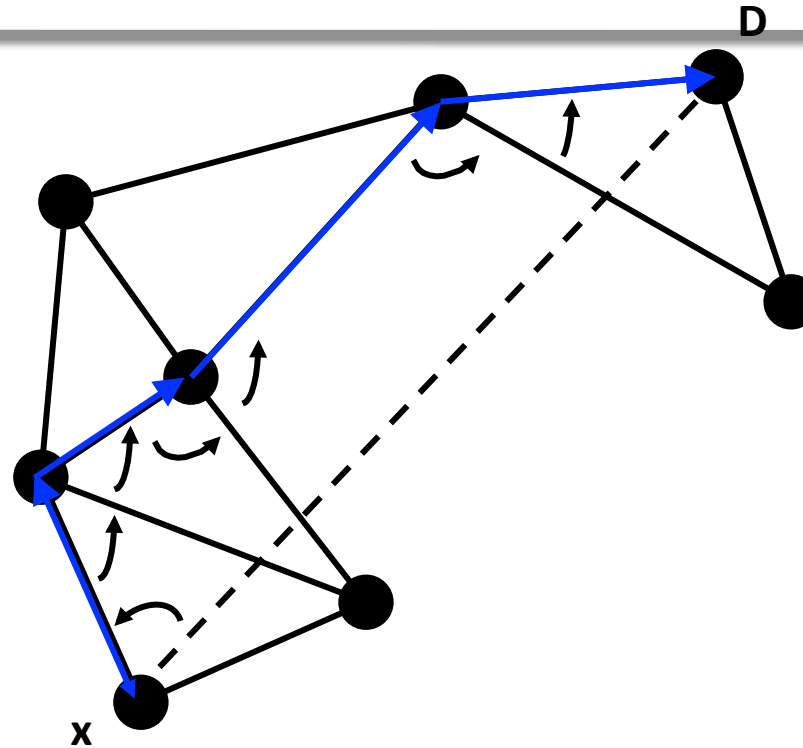


Full Greedy Perimeter Stateless Routing



- All packets begin in greedy mode
- Greedy mode uses full graph
- Upon greedy failure, node marks its location in packet, marks packet in perimeter mode
- Perimeter mode packets follow simple planar graph traversal:
 - Forward along successively closer faces by right-hand rule, until reaching destination
 - Packets return to greedy mode upon reaching node closer to destination than perimeter mode entry point

Perimeter Mode Forwarding Example



- Traverse face closer to D along \overline{xD} by right-hand rule, until crossing \overline{xD}
- Repeat with next-closer face etc.

Summary



-
- We have described ad hoc routing protocols for connected networks
 - When there is no connected path between nodes delay tolerant routing protocols should be applied
 - When geographical position of the nodes is known and the network is connected geographical routing protocols can be used

References



- A. Vahdat and D. Becker, Epidemic routing for partially-connected ad hoc networks, 2000.
- M. Musolesi and C. Mascolo. CAR: Context-aware Adaptive Routing for Delay Tolerant Mobile Networks. In IEEE Transactions on Mobile Computing. Vol. 8(2). pp. 246-260. February 2009.
- Karp, B. and Kung, H.T., GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, in Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), Boston, MA, August, 2000, pp. 243-254