Mobile Ad-Hoc Networks

Mads Darø Kristensen
Niels Olof Bouvin

What is Ad-hoc Networking?

- **Enable sharing/collaboration using portable devices, typically over radio connections**
- **No centralised components and no fixed infrastructure (e.g., base stations, access point, or wired links)**
- **Challenges:**
  - Limited communication range, bandwidth, processing power, storage capacity, and battery life
  - Portable devices are moved about – network topology can be very transient

Why Use Ad-hoc Networking?

- **Sometimes there may be no network infrastructure available**
  - Remote areas, unplanned meetings, extending existing infrastructure
  - Emergency relief personnel deployed into an area.
  - Military where infrastructure has been destroyed or is untrusted.
- **Sometimes users will not or cannot use the available infrastructure**
  - Time to register and access the service.
  - Prohibitive cost, performance, or capacity of the service.

MANETs and Pervasive Computing

- We are almost always working with wireless networking in pervasive computing
- Often these access the network (perhaps the Internet) through a managed Wi-Fi connection, or
  - they may simply use point-to-point Bluetooth connections.
- In many of these cases it may make sense to apply MANET techniques instead; which would for example enable multi-hop routing.
- A sensor network is a good example of MANET

Wireless Links – Challenges

- **Communication is based on radio waves.**
- **Signal strength diminishes with distance and may vary significantly and seemingly unpredictable**
- **Radio waves may be blocked or absorbed by objects such as buildings, mountains, and rain.**
- **Can be unidirectional – e.g., A can hear B, but B cannot hear A.**
- **Signals are broadcast – others can listen in…**
  - while a security problem, this can also be used to boost performance!

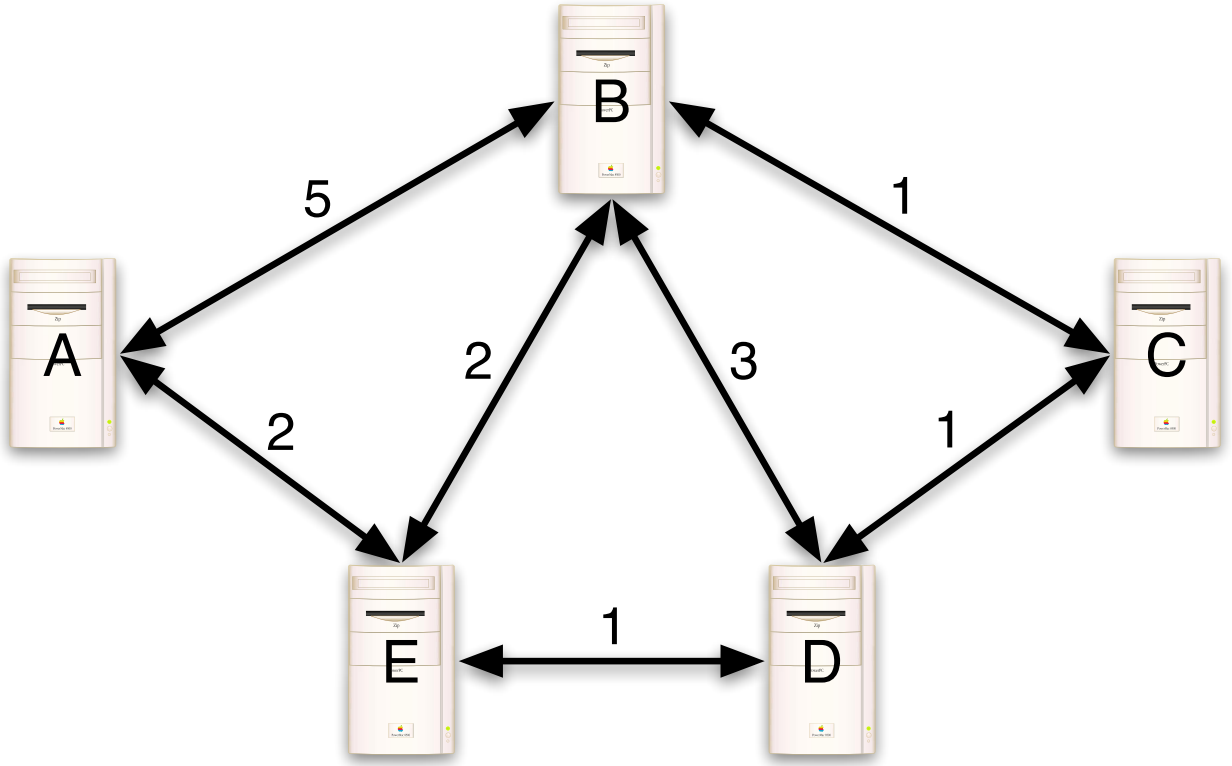
Overview

- **Routing basics**
- **MANET routing**
- **Energy efficient MANET routing**

Structure of this talk

- **What is network routing?**
- **Classical approaches**
  - Link state (Dijkstra)
  - Distance vector (Bellman-Ford)
- **Summary**

What is routing?



Properties of wired networks

- **Nearly static configuration**
  - Nodes stay in the same place for a long time.
  - Interconnecting links exist for an extended period of time.
- **Small fluctuations in link quality**
  - Mostly link quality only fluctuates when very high traffic rates are experienced.
- **High traffic rates**
  - It is often the case in wired networks that all nodes are participating in communication at the same time.

Classical routing approaches

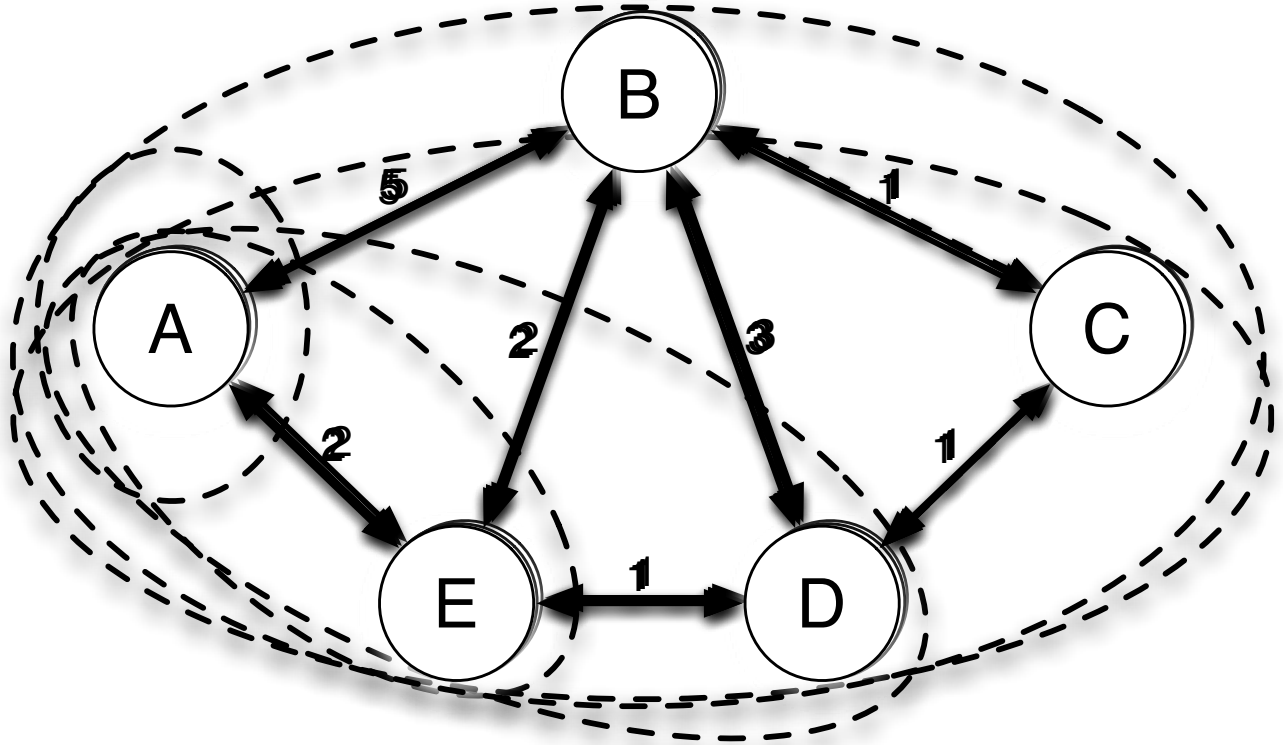
- **There are two main approaches towards generating routing tables in static networks:**
  - Link State protocols.
  - Distance Vector protocols.
- **Both of these approaches build full routing tables including information about all reachable nodes in the network for each node in the network.**

Link state

- **In an LS algorithm it is assumed that the network topology and all link costs are known**
  - Given this information routing becomes a simple case of finding the single-source shortest path
- **This means that global information is needed, and that this information must be communicated throughout the network periodically**
- **All nodes will have a routing table that for each node in the network contains an entry with the distance to the node and the id of the next-hop neighbour on the path to the node**

Dijkstra's algorithm

(seen from A)



D(B), p(B)	D(C), p(C)	D(D), p(D)	D(E), p(E)
4, E	4, E	3, E	2, A

Dijkstra's algorithm

- **The preceding slides only gave some intuition as to how Dijkstra's single-source shortest path algorithm works**
  - http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm for a more in-depth description

Properties of link state

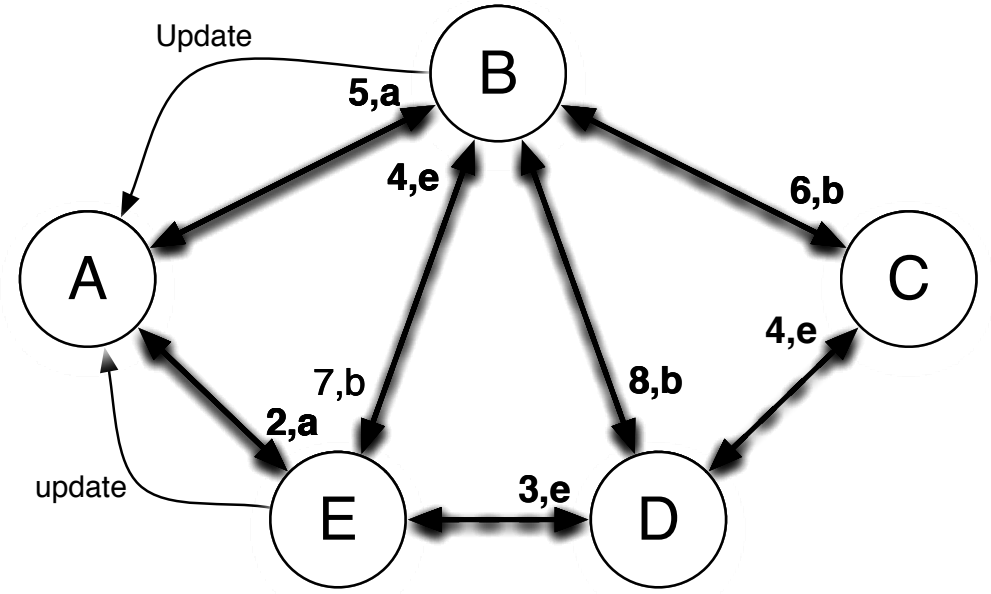
- Periodically (and on changes in link cost) nodes broadcast information about their outgoing links to the entire network.
- This does not scale very well – LS is therefore best suited for relatively small, stable networks.
- Upon changes the entire routing table is recomputed. This is an O(m + n log n) operation, where n = #nodes and m = #edges, which requires some processing power.

Distance vector

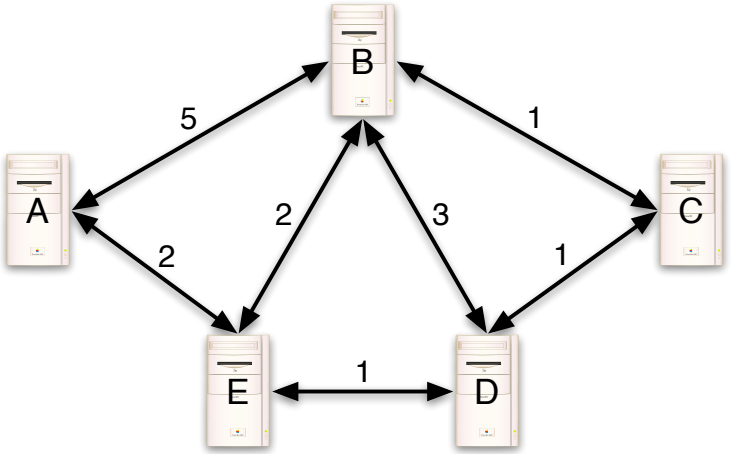
- **A DV algorithm is iterative, asynchronous, and distributed**
  - Information about the network is only exchanged between neighbours
  - When such updates arrive only the affected parts of the routing table are updated
- **When updates mean that changes are made to the routing table these changes are propagated to neighbour nodes**
  - The algorithm is self-terminating – eventually a quiescent state is reached

Bellman-Ford's algorithm

(seen from A)



D^A	B	E
D^A	B	E
B	5	4
B	5	4
C	6	∞
E	∞	3
D	8	3
E	7	2



Bellman-Ford's algorithm

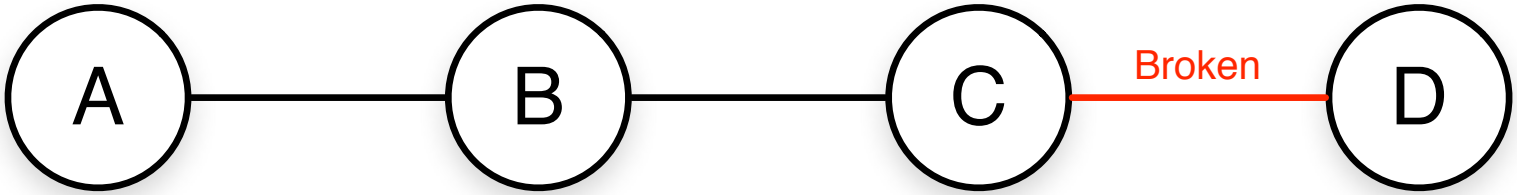
- **Again, the preceding description only provided an intuition as to how the algorithm works**
  - for more information see e.g., http://en.wikipedia.org/wiki/Bellman-Ford_algorithm

Properties of distance vector

- **In stable networks it quickly enters a quiescent state – just like LS.**
- **But, in contrast to LS, it also works well in unstable networks.**
  - Routing tables are updated incrementally.
  - Updates are only sent to neighbours – and they only propagate further if a new shortest path is found.
- **But, the simple DV algorithm here has some problems.**
  - Count-to-infinity – can be solved by using 'poisoned reverse' or 'split-horizon'.

Count-to-infinity

(what is the distance to D?)



Time	A	B	C
0	3,B	2,C	1,D
1	3,B	2,C	3,B
2	3,B	4,C	3,B
3	5,B	4,C	5,B
...	...	...	...

Summary

- **Routing is**
  - finding the cheapest path between two nodes in a network graph
    - cheapest may be according to any metric (bandwidth, latency, cost, or simply routing hops)
- **Most classical routing protocols are based on either**
  - link state, exemplified by Dijkstra's algorithm
  - distance vector, exemplified by Bellman-Ford's algorithm
- **Link state uses global state and is most efficient in small, stable networks**
- **Distance vector builds its routing tables incrementally and is thus more suited in larger, unstable networks**

Overview

- Routing basics
- **MANET routing**
- **Energy efficient MANET routing**

Structure of this talk

- Desirable properties of MANET
- Proactive vs. reactive routing
- Routing protocols
  - Destination sequenced distance vector (DSDV)
  - Ad-hoc on-demand distance vector (AODV)
  - Dynamic source routing (DSR)
- Summary

MANET properties

- **In a mobile ad-hoc network there can be a high degree of mobility.**
- **Specialised protocols that are geared towards mobility are needed.**
- **If one had to use either LS or DV in a mobile network DV should be the preferred choice.**
  - Only part of the routing table is affected upon changes.
  - Updates are only sent to neighbors, i.e., less control traffic is generated.

Desirable properties

- Minimal control overhead.
- Minimal processing overhead.
- Multi-hop routing capability.
- Dynamic topology maintenance.
- Loop prevention.

Proactive routing

- Nodes periodically exchange routing information and attempt to maintain current routing information of the entire network.
- The exchange of routing information may be done periodically, or may be triggered, when topology changes are detected.
- Ensures efficient routing at the cost of constantly exchanging routing information between peers.
- Relatively high control overhead in low traffic scenarios.
- Using the “fresh” routing tables is a big advantage in high traffic/low latency scenarios.

Reactive routing

- Route establishment is done on-demand. Nodes only try to find a route to a destination when actually needed.
- Nodes do not maintain much state, and there is no constant overhead.
- May save a lot of control overhead because routing information is only sent when needed.
- Routing is more costly and unpredictable.
- Especially effective in low traffic scenarios.

Destination Sequenced Distance Vector

- **A proactive approach.**
- **Distance vector algorithm that:**
  - uses sequence numbers to avoid routing loops.
  - is more bandwidth efficient through the use of incremental updates.
  - delays route advertisements to damp fluctuations.

DSDV routing table

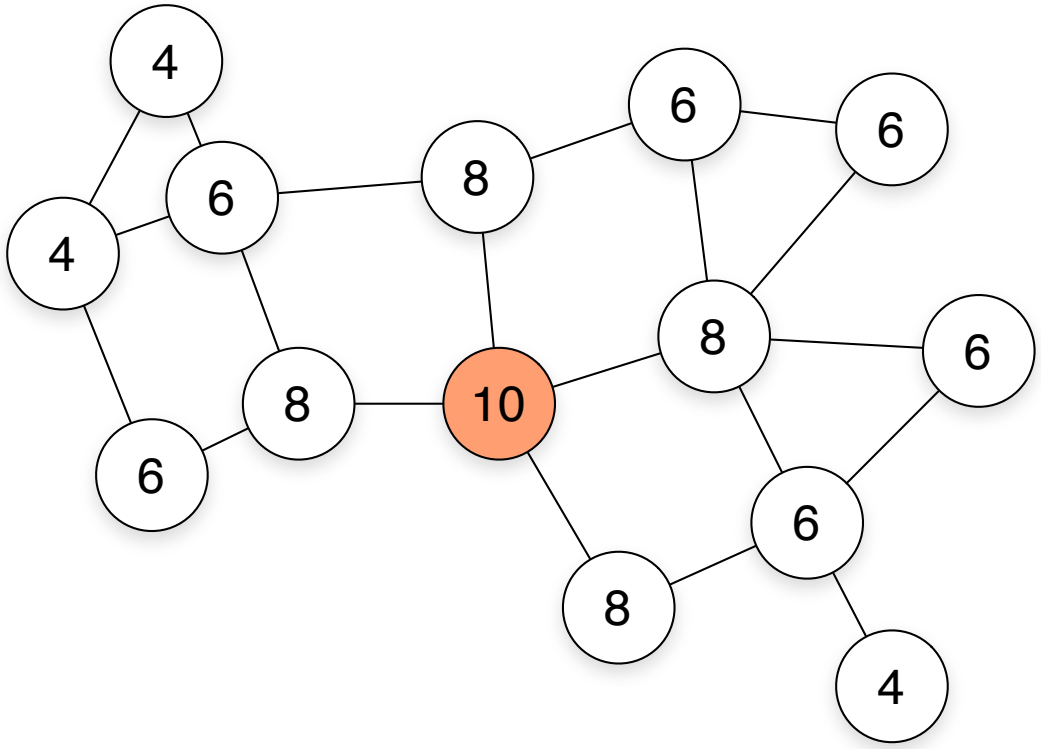
- **Each node maintains a table of:**
  - (Destination, Next hop, metric, sequence number)
- **When reacting to updates the route with the highest (newest) sequence number is always preferred.**
  - If the sequence numbers are identical, the route with the lowest cost is chosen.

Working with sequence numbers

- **A node maintains its own sequence number.**
  - it is initially set to zero, and whenever a route update is sent out, it is incremented by **two**.
  - Sequence numbers set by the node itself will thus always be even numbers.
- **Neighbours monitor each other to see if links break.**
  - When a link breaks, the node to discover it sends out an update with the cost set to ∞ and the sequence number incremented by **one**.
  - Any subsequent update sent by the disappeared and reappeared node will automatically supersede this message, flushing out stale information.

Loop prevention

- Following the simple sequence number rules ensures that DSDV is loop free.



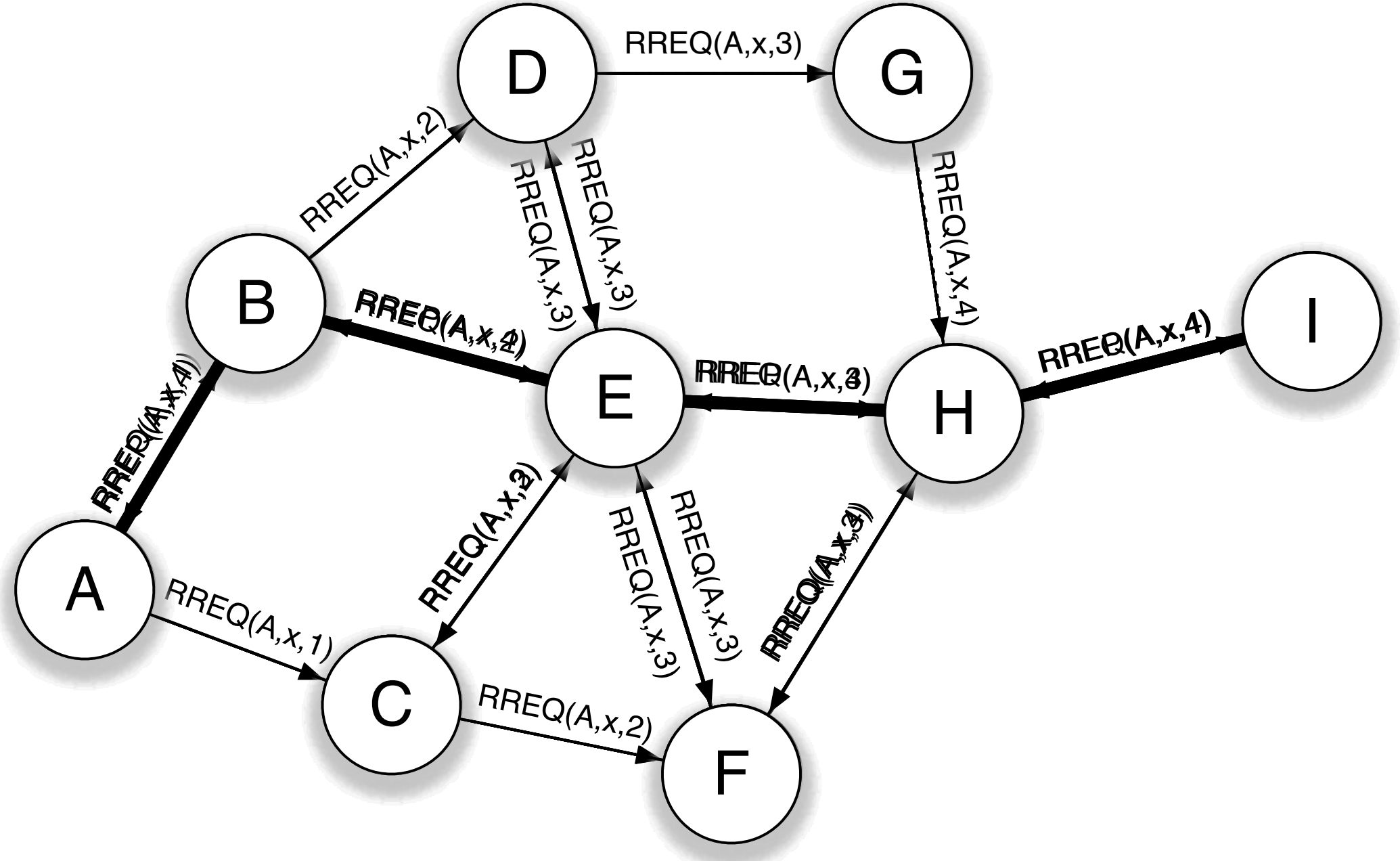
Route updates

- **Updates in DSDV are bundled together to bring down the overhead of sending this control data.**
  - Full updates are sent infrequently.
    - Includes the full routing table – used to bootstrap new neighbours.
  - Incremental updates are sent frequently.
    - Includes only information about changed routes.
    - Must fit within a single packet otherwise a full update is sent.

Ad-Hoc On-Demand Distance Vector

- A re-active approach towards routing.
- Uses destination sequence numbers to avoid loops just like DSDV.
- Routes are discovered by broadcasting route requests (RREQs) containing:
  - source address, source seq. no., broadcast id, destination address, destination seq. no., hop count.
- When the RREQ reaches a node that has a recent route to the destination a route reply (RREP) is sent back to the source using the reverse path.

AODV example



Route replies

- **Note that multiple RREPs may be returned in response to a RREQ.**
  - RREPs are only forwarded by the intermediate nodes if:
    - they have a higher destination sequence number.
    - they have the same seq. no. but a lower hop count.
- **When the RREP is sent back to the source, the intermediate nodes record which nodes they received it from and thus a forward path is built.**

Route maintenance

- **Upon link failure:**
  - Upstream neighbour sends RREP with seq. no. +1 and hop count set to infinity to any active neighbours – that is neighbours that are using the route.
- **Rediscovery:**
  - Uses a higher sequence number (like in DSDV) and thus overwrites the RREP sent out upon link failure.

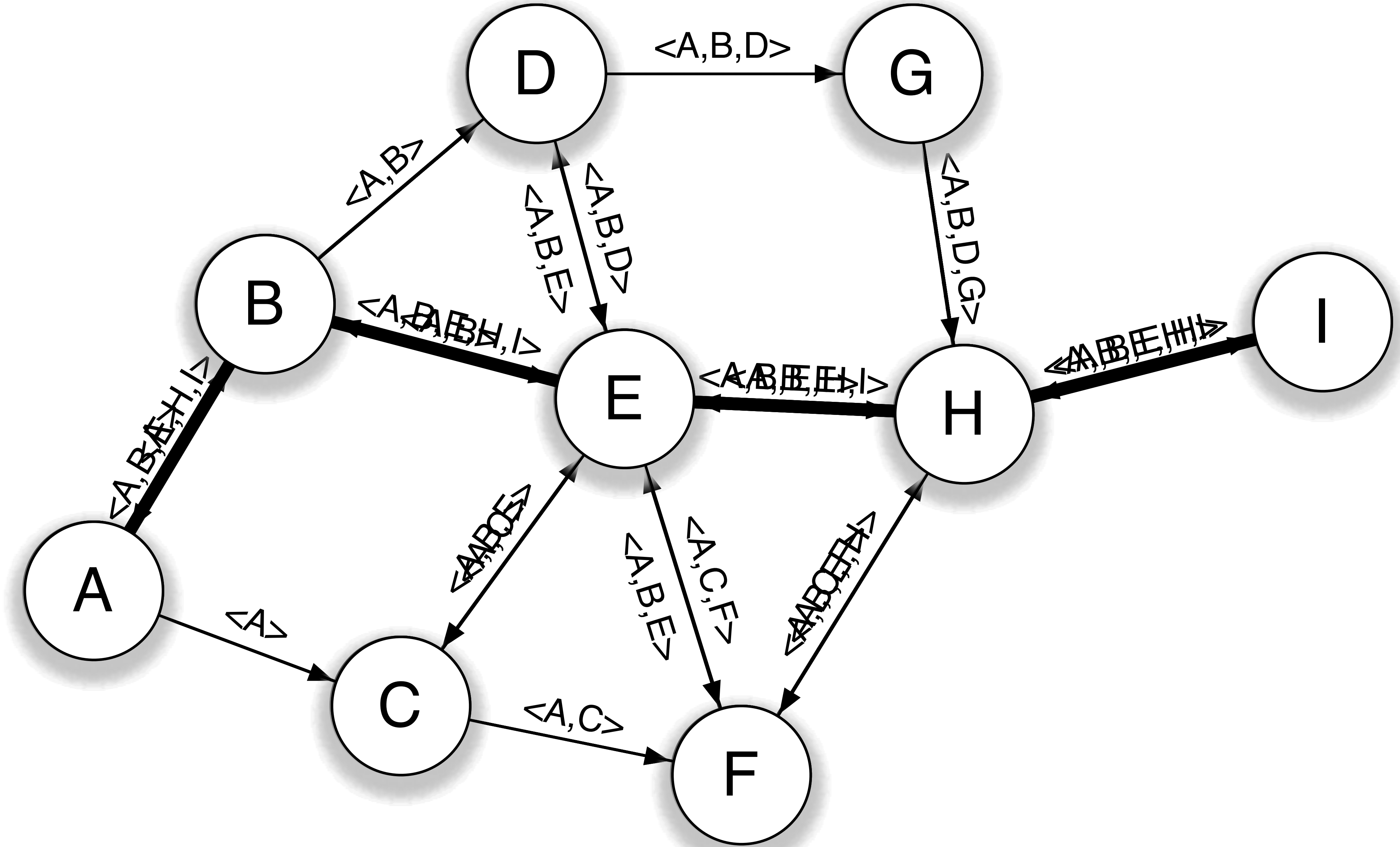
Properties of AODV

- **Minimises control traffic by only maintaining active routes.**
- **Very small processing overhead—only simple table lookups are done.**
- **Functions very well in low traffic scenarios.**
- **The initial cost can be high in high traffic scenarios.**
  - Especially if there is also a high degree of mobility so that routes must be recomputed often.

Dynamic Source Routing

- Re-active routing protocol using a discovery procedure similar to the one used in AODV.
- Route requests (and replies) include the entire routing path, i.e., intermediate nodes need not keep any state while forwarding RREPs and RREQs.
- Does *not* require bi-directional links.

DSR example



Uni-directional links

- **The previous example used the reverse path to return the route reply. This means that bi-directional links are required.**
- **Instead of using the reverse path, a node can send a RREP by:**
  - Using an already known path to the source.
  - Sending out a RREQ to discover a path to the source.
    - The route from source to destination is piggy-backed on this RREQ to avoid looping.

Loop prevention

- Given that the entire route is known (and not just the next hop neighbour and a distance as in AODV), loop prevention becomes trivial.
- Nodes simply do not forward a RREQ that is already on the known path, i.e., that is used earlier on in the route.

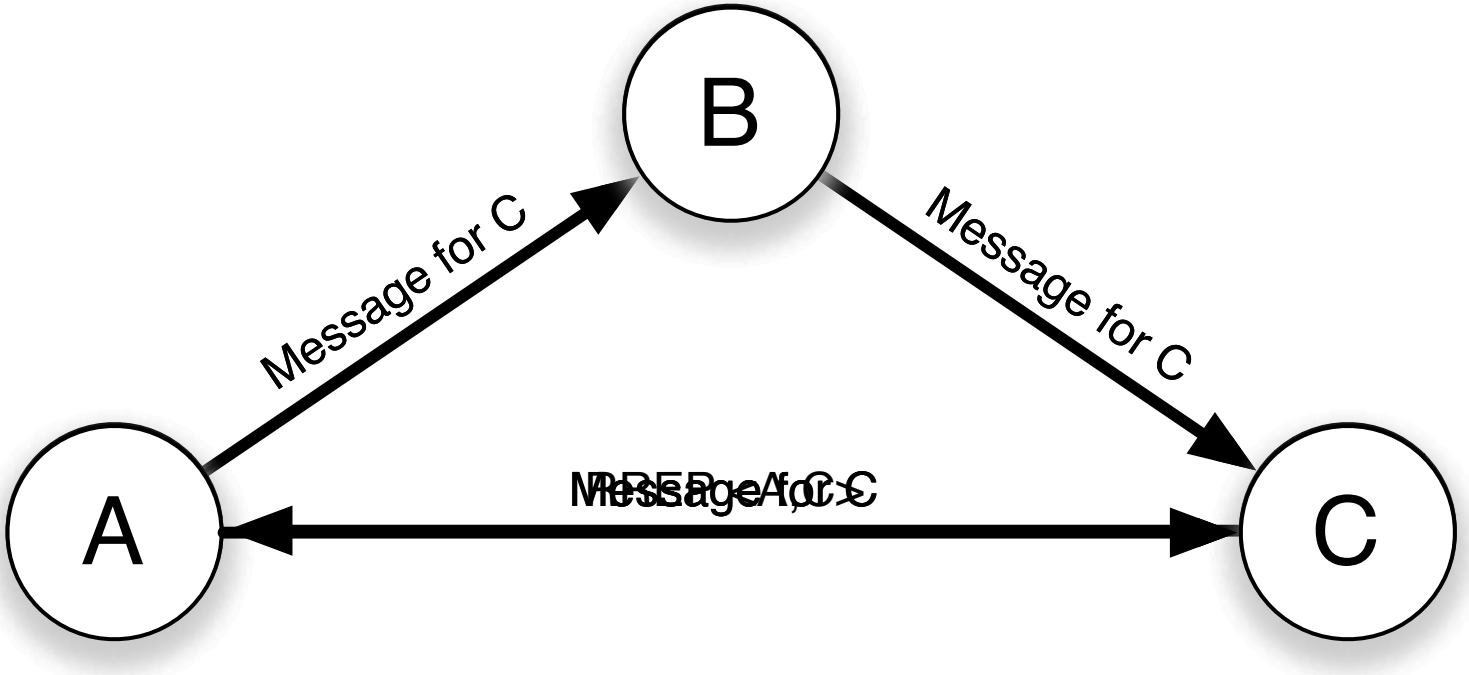
The route cache

- **Instead of a conventional routing table a DSR node maintains a route cache of known routes to destinations.**
  - The cache is in effect a tree rooted at the node.
- **The cache may contain multiple routes to the same destination.**
  - This makes for graceful handling of route errors. If a link fails it may be possible to select another route.
  - If an intermediate node, that is in an active path, discovers a link failure, it will send a RREP to the source with the alternative path.

Promiscuous mode operation

- **Wireless communication is broadcast – everyone can listen in. This can be used in DSR:**
  - *Passive acknowledgements* – observing that a message gets forwarded.
  - *Updating the route cache* – when overhearing a message the overhearing node may update its route cache with the route of the message.
    - If the overheard message is an error message steps can be taken to mend the routes if needed.
  - *Route shortening* – If the destination overhears a message destined for it, it may notify the sender that the intermediate node can be skipped.

Route shortening



Summary

- **MANET are mobile ad-hoc networks**
  - high mobility leads to the need for new routing approaches
  - lack of infrastructure means that participating nodes must route themselves
- **MANET routing protocols fall into two categories**
  - proactive algorithms tries to keep an up-to-date routing table at all time
  - reactive algorithms build routing tables only when needed

Overview

- Routing basics
- MANET routing
- **Energy efficient MANET routing**

Structure of this talk

- Why energy efficiency?
- Power control vs. power save
- An example of a power save approach
- Summary

Why energy efficiency?

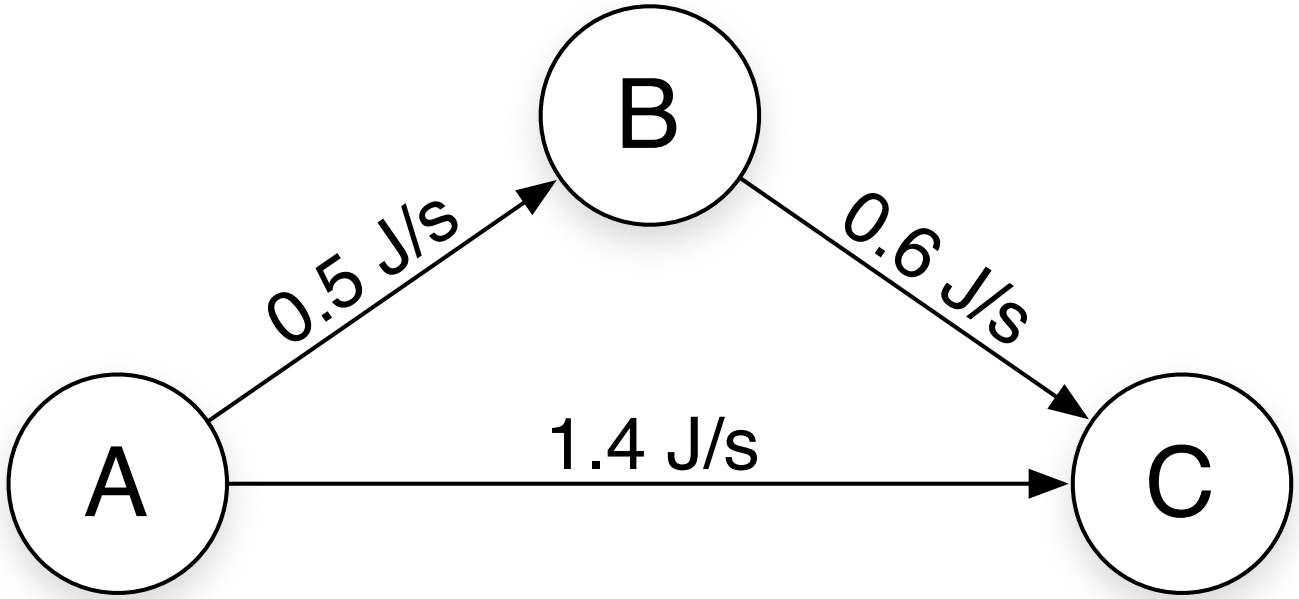
- **Nodes in a MANET are mobile and usually battery powered**
  - and network interfaces consume a large amount of energy
- **Nodes route each others' data in a MANET**
  - i.e., nodes use the network interface continuously to help others, which in the long run may deplete their battery
  - all nodes are needed so that the network does not become fragmented
- **Sensors may be discarded in small sensor networks, when their battery are depleted**
  - thus being energy efficient prolongs the lifetime (and thus utility) of the sensor network

Energy consumption of wireless network interfaces

- **Wireless communication comes at a cost. A study of energy consumption of a particular wi-fi device has shown that:**
  - Transmitting costs 1.4 W
  - Receiving costs 1.0 W
  - Being idle costs 0.83 W
  - Sleeping costs 0.13 W
- **Notice how expensive it is to be idle! And furthermore, when using a broadcast media a lot of time is spent discarding packets, i.e., a lot of time is spent receiving useless packets.**

Power control

- Power-control schemes are based on the fact that there is a non-linear relation between transmission range and energy used to transmit.



Power control

- **Power-control schemes thus use less energy if possible – leading to a shorter transmission range.**
- **The benefits of this are:**
  - The total energy used for sending a packet is decreased.
  - The noise in the network is brought down by reducing the transmission range, i.e., only close neighbours are disturbed when a message is sent.

Power save

- **Another approach towards energy efficiency is power-save.**
  - Based on the fact that, in most MANETs, idle time dominates. That is, most nodes spend most of their time idling, and therefore most of their energy is spent while doing nothing.
  - Utilises the sleep states of the wireless interfaces to put these nodes to sleep.

Challenge

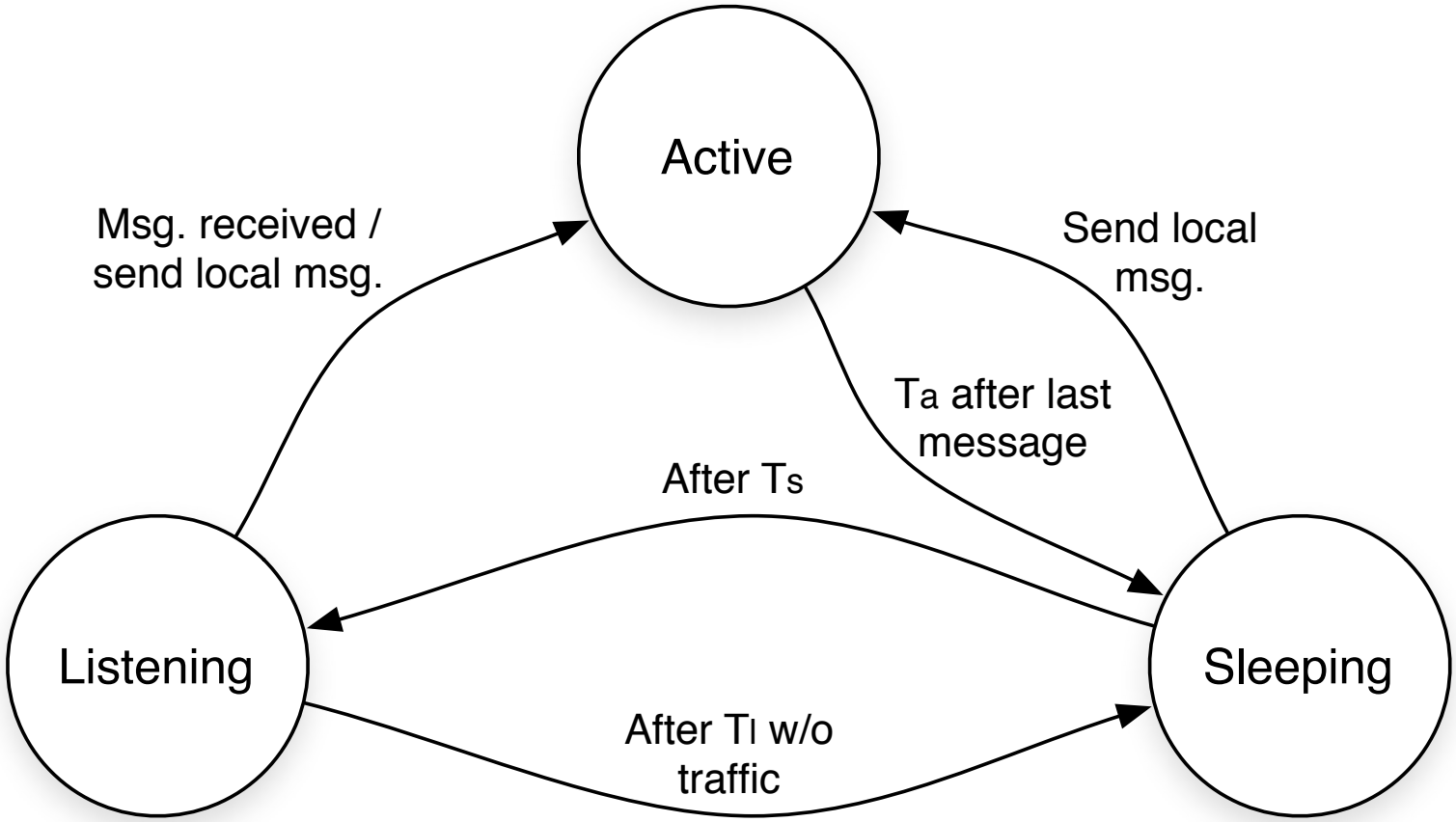
- **How do we maintain connectivity when some of the nodes are sleeping?**
  - By using retransmissions.
  - Or by making sure that enough nodes are kept awake at all times.
  - Or a combination of the two…

BECA / AFECA

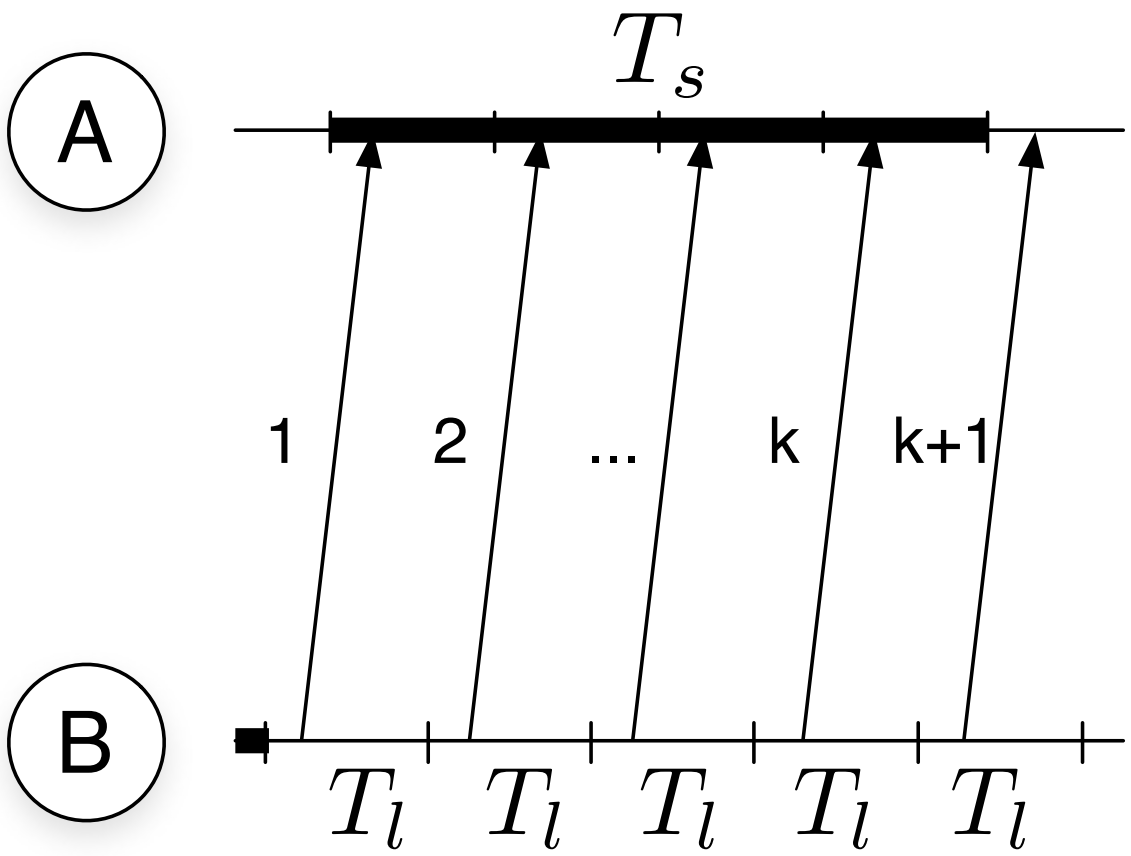
- **Simple power-save approach based on retransmissions and fixed sleep/wake intervals.**

Param	Description
T_l	Listen interval ($T_l = T_o$)
T_S	Sleep interval ($T_s = kT_o$)
T_a	Active interval ($\approx 60s$)
T_o	Retransmission interval
R	No. of retransmissions ($R \geq k + 1$)

BECA states and transitions



BECA retransmissions



Effects of BECA

- Potentially a power saving of $\frac{k}{k+1}$ can be obtained
- The energy savings comes at the expense of large discovery latencies. When nodes on the path to a destination node are sleeping path discovery can be very slow
  - In the worst case this delay will be kT_o for every routing hop
- When nodes are actively communicating, no further delays are experienced

AFECA – a more "advanced" BECA

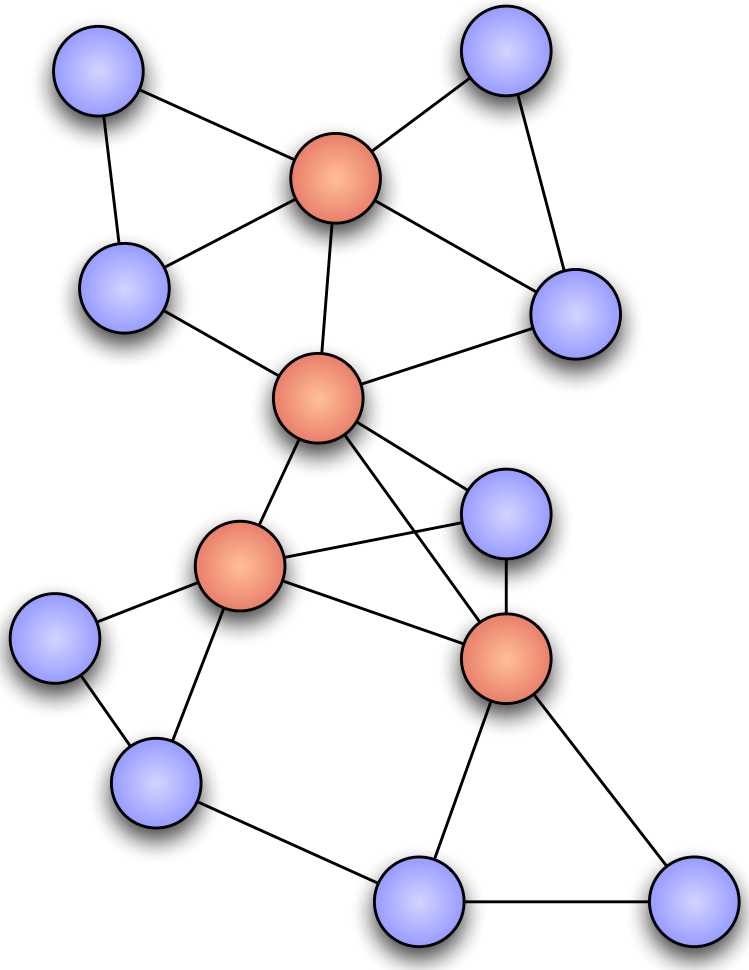
- **AFECA extends BECA by taking node density into consideration.**
- **It makes the observation that, when the node density is high, there is a high probability that some node in the neighbourhood is awake to perform routing tasks.**
  - Therefore the individual nodes may sleep for a longer period of time.
  - The new sleep interval is therefore calculated as:

$$T_{sa} = Random(1, N) \times T_s$$

Span

- **Span is not really a power saving approach – it is rather an elegant way to calculate a Connected Dominating Set (CDS) in a distributed fashion.**
- **Some nodes are chosen to be coordinators. These coordinators are members of the CDS.**
- **A CDS is "a connected subset S of a graph G such that every vertex u in G is either in S or adjacent to some vertex v in S.".**
  - That is: from the CDS all nodes in the network can be reached in one hop

The connected dominating set of coordinator nodes



Using the connected dominating set

- Because of the properties of a CDS, it is ideal to use as a *routing backbone*
- The idea is that only coordinators are used for routing purposes—non-coordinator nodes may thus spend a large part of their time sleeping.
  - Coordinators on the other hand must never sleep.
- To be fair, and to extend the lifetime of the entire network, nodes take turns being coordinators.

Coordinator selection

- **By letting neighbours exchange information about their one-hop neighbours they gain knowledge of their entire two-hop neighbourhood.**
  - This information includes knowledge about who is currently operating as coordinators.
- **Periodically non-coordinator nodes check to see if they should become coordinators.**
  - They do so by calculating a back-off delay based on the information they have got about their two-hop neighbourhood.

Coordinator selection

- **After waiting for the calculated amount of time two things may have happened:**
  - Another node in the vicinity may have announced that it wants to become a coordinator – in which case this node does not have to.
  - No other announcements have been heard and the node thus announces that it is now a coordinator.
- **Periodically coordinator nodes try to withdraw from the coordinator role…**

Combining Span and BECA/AFECA

- **Coordinator nodes are not allowed to enter the sleep state of BECA.**
- **Non-coordinator nodes can sleep for longer periods of time because the coordinators are always available for doing routing.**
- **When Span is utilised no retransmissions are needed between coordinator nodes = big savings on retransmissions!**
  - In fact retransmissions are *only* needed at the last hop of a path.

Summary

- **Energy efficiency is of high importance in MANETs because:**
  - Nodes are mobile and thus battery powered.
  - Nodes are performing routing on each others behalf, which has the potential to deplete a nodes energy resources without the node itself being “active”.
- **Energy efficiency schemes generally fall into two categories:**
  - Power-save, utilising the sleep states of wireless interfaces.
  - Power-control, controlling the transmission power of the wireless interface