15-441: Computer Networking

Lecture 27: Ad-Hoc Networks

Scenarios and Roadmap

- Point to point wireless networks
  - Example: Your laptop to CMU wireless
  - Challenges:
    - Poor and variable link quality (makes TCP unhappy)
    - Many people can hear when you talk
  - Pretty well defined.
- Ad hoc networks (wireless++)
  - Rooftop networks (multi-hop, fixed position) (MESH)
  - Mobile ad hoc networks (MANET)
  - Adds challenges: routing, mobility
  - Some deployment + some research
- Sensor networks (ad hoc++)
  - Scatter 100s of nodes in a field / bridge / etc.
  - Adds challenge: Serious resource constraints
  - Current, popular, research.

Wireless Challenges (review)

- Need to share airwaves rather than wire
  - Don't know what hosts are involved
  - Host may not be using same link technology
  - No fixed topology of interconnection
  - Interference
    - Other hosts: collisions, capture, interference
    - The environment (e.g., microwaves + 802.11)
- Mobility -> Things change often
  - Environmental changes do too
  - How do microwaves work? Relate to 802.11 absorption.
- Other characteristics of wireless
  - Noisy → lots of losses
  - Slow
  - Multipath interference

Ad Hoc Networks

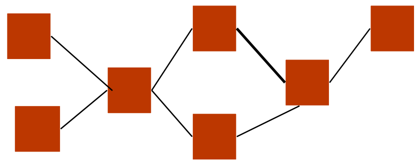
- All the challenges of wireless, plus some of:
  - No fixed infrastructure
  - Mobility (on short time scales)
  - Chaotically decentralized (:-)
  - Multi-hop!
- Nodes are both traffic sources/sinks and forwarders
- The big challenge: Routing

Ad Hoc Routing

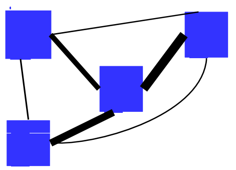
- Find multi-hop paths through network
  - Adapt to new routes and movement / environment changes
  - Deal with interference and power issues
  - Scale well with # of nodes
  - Localize effects of link changes

Traditional Routing vs Ad Hoc

- Traditional network:
  - Well-structured
  - ~O(N) nodes & links
  - All links work ~= well



- Ad Hoc network
  - N^2 links - but many stink!
  - Topology may be really weird
    - Reflections & multipath cause strange interference
  - Change is frequent



Problems using DV or LS

- DV loops are very expensive
  - Wireless bandwidth << fiber bandwidth…
- LS protocols have high overhead
- N^2 links cause very high cost
- Periodic updates waste power
- Need fast, frequent convergence

Proposed protocols

- Basic Taxonomy:
  - Reactive (on-demand)
  - Proactive (table driven)
    - Source routing
    - Hop-by-hop routing
- Destination-Sequenced Distance Vector (DSDV)
- Dynamic Source Routing (DSR)
- Ad Hoc On-Demand Distance Vector (AODV)

- Let's look at DSR first

DSR

- Source routing
  - Intermediate nodes can be out of date
- On-demand route discovery
  - Don't need periodic route advertisements
- (Design point: on-demand may be better or worse depending on traffic patterns…)

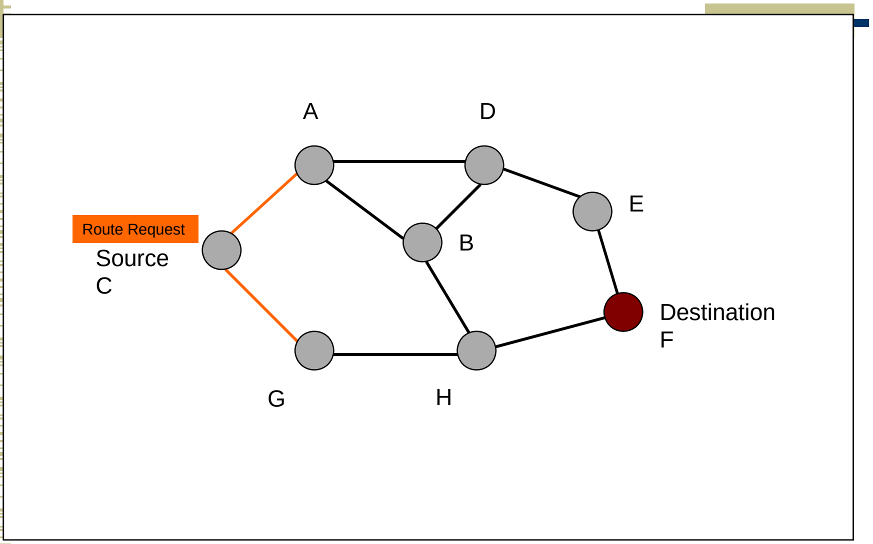
DSR Components

- Route discovery
  - The mechanism by which a sending node obtains a route to destination
- Route maintenance
  - The mechanism by which a sending node detects that the network topology has changed and its route to destination is no longer valid

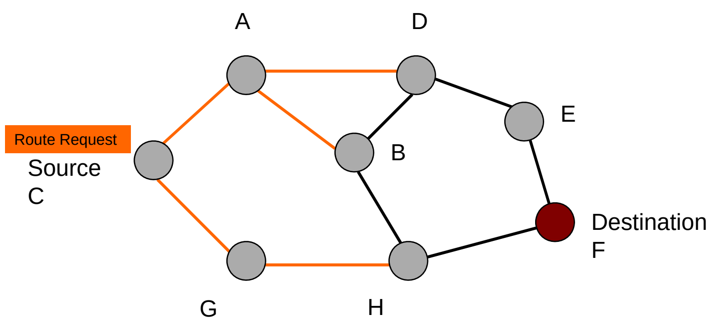
DSR Route Discovery

- Route discovery - basic idea
  - **Source** broadcasts route-request to **Destination**
  - Each node forwards request by adding own address and re-broadcasting
  - Requests propagate outward until:
    - Target is found, or
    - A node that has a route to Destination is found

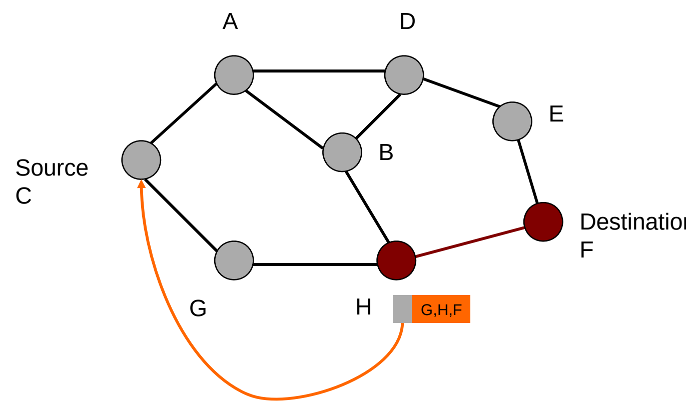
C Broadcasts Route Request to F



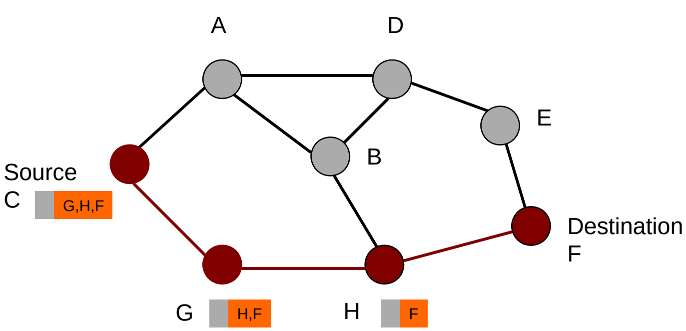
C Broadcasts Route Request to F



H Responds to Route Request



C Transmits a Packet to F



Forwarding Route Requests

- A request is forwarded if:
  - Node is not the destination
  - Node not already listed in recorded source route
  - Node has not seen request with same sequence number
  - IP TTL field may be used to limit scope
- Destination copies route into a Route-reply packet and sends it back to **Source**

Route Cache

- All source routes learned by a node are kept in Route Cache
  - Reduces cost of route discovery
- If intermediate node receives RR for destination and has entry for destination in route cache, it responds to RR and does not propagate RR further
- Nodes overhearing RR/RP may insert routes in cache

Route cache: Issues?

Sending Data

- Check cache for route to destination
- If route exists then
  - If reachable in one hop
    - Send packet
  - Else insert routing header to destination and send
- If route does not exist, buffer packet and initiate route discovery

Discussion

- Source routing is good for on demand routes instead of a priori distribution
- But, high packet overhead ← Why esp important?
- Route discovery protocol used to obtain routes on demand
  - Caching used to minimize use of discovery
- No Periodic messages
- But, need to buffer packets

AODV

- On-demand protocol
- Table-driven, distance-vector routing
- Similar to DSR in finding routes, but
  - Uses sequence numbers on route updates
  - Has an idea of freshness of a route
- RREQ includes normal stuff plus
  - src-seq, Broadcast-seq, dest-seq, hop-count

RREQ/RREP

- On RREQ
  - REPLY
    If my dest-seq >= received dest-seq OR
    I am destintation
  - DISCARD
    If src-adr & broadcast-seq were seen
  - Re-broadcast
    otherwise

Route Maintanience

- Can update routing table when they get information that improves on the routing metric:
  - A smaller hop-count with a larger dest-seq number
- Broken links cause unsolicited RREP to be sent (may cause new RREQ with higher dest-seq number)
- Eavesdrop and periodic hellos

Discussion

- Can use on-demand discovery and do hop-by-hop routing
  - Only 1 route per destination
- Route discovery protocol used to obtain routes on demand
  - Routing tables & dest-seq keep routes fresh and reduce packet overhead
  - In changing networks, has fast convergence
- Periodic messages so more network overhead than DSR

Forwarding Packets is expensive

- Throughput of 802.11b =~ 11Mbits/s
  - In reality, you can get about 5.
- What is throughput of a chain?
  - A -> B -> C ?
  - A -> B -> C -> D ?
  - Assume minimum power for radios.
- Routing metric should take this into account

Routing Metrics

- Used to determine best route to take
- What does "best" mean?
- What should it take into account?

Routing Metrics

- Used to determine best route to take
- What does "best" mean?
- What should it take into account?
  - Route stability
  - Good performance for minimum weight paths
  - Low overhead to find min weight paths
  - Loop-free routing

Route Stability

- Unstable paths can cause ?
- Causes of route instability
  - Mobility
  - Link failures
  - Link capacity
  - Changes in network traffic

Mobility, link failures, link capacity: Topology

Changes in network traffic: Load-sensitive

Route Stability

- Unstable paths can cause ?
- Causes of route instability
  - Mobility
  - Link failures
  - Link capacity
  - Changes in network traffic
- Size of ad-hoc networks (compared to wired networks) implies load-sensitive metrics likely to cause lots of changes.

Mobility, link failures, link capacity: Topology

Changes in network traffic: Load-sensitive

Performance

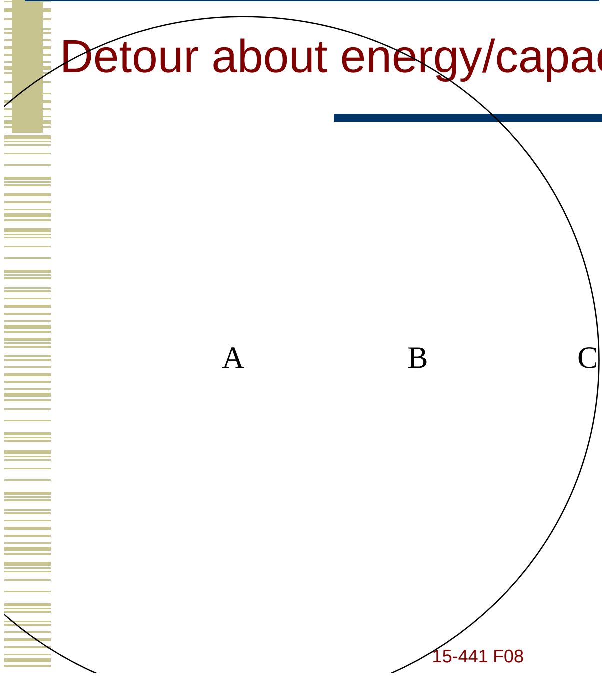
- Goal: route packets through min-weight paths
  - High-throughput
  - Low latency
  - ?
- What characteristics lead to good performance?

Performance

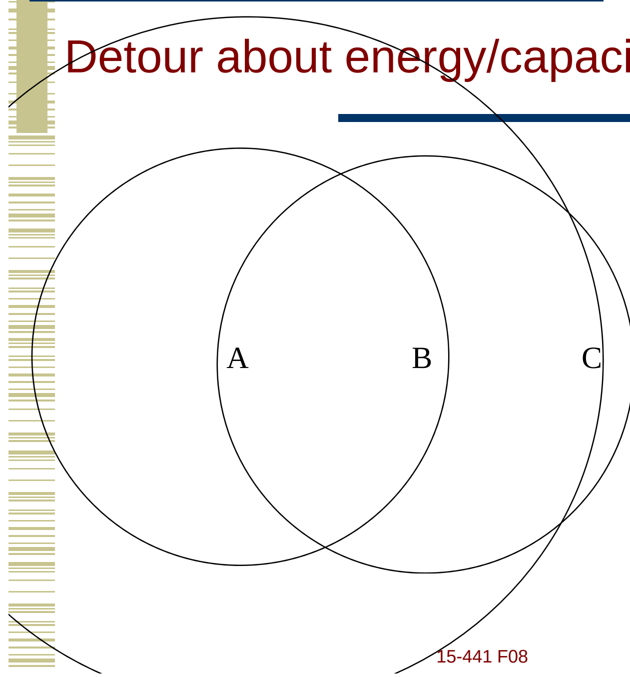
- Goal: route packets through min-weight paths: i.e., high-throughput, low-latency, …
- What characteristics lead to good performance?
  - Path length
  - Link capacity

Distance impacts capacity in several ways

Detour about energy/capacity



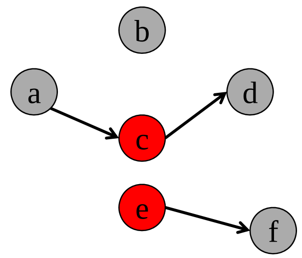
Detour about energy/capacity



Min path length can be at odds with improving capacity or improving energy use

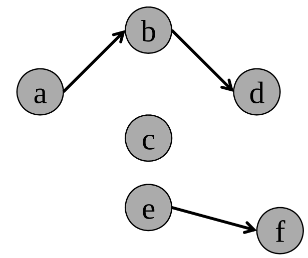
Performance

- Goal: route packets through min-weight paths: i.e., high-throughput, low-latency, …
- What characteristics lead to good performance?
  - Path length
  - Link capacity
  - Packet loss ratios
  - Interference
    - Inter-flow
    - Intra-flow



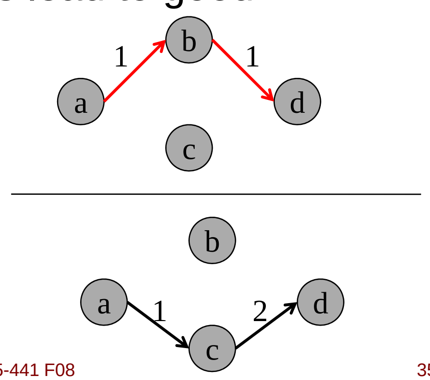
Performance

- Goal: route packets through min-weight paths: i.e., high-throughput, low-latency, …
- What characteristics lead to good performance?
  - Path length
  - Link capacity
  - Packet loss ratios
  - Interference
    - Inter-flow
    - Intra-flow



Performance

- Goal: route packets through min-weight paths: i.e., high-throughput, low-latency, …
- What characteristics lead to good performance?
  - Path length
  - Link capacity
  - Packet loss ratios
  - Interference
    - Inter-flow
    - Intra-flow



Example Metrics

- Hop-count
- Expected Transmission Count (ETX)
- Expected Transmission Time (ETT)
- Weighted Cumulative ETT (WCETT)
- …

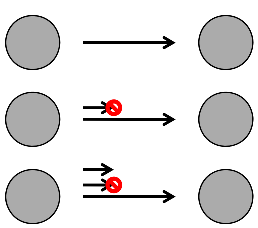
Hop Count

- Used in DSR and AODV
- Smaller path lengths are prefered
- Doesn't take into account link quality
  - Transmission rates
  - Packet loss
  - Interference
- But, simple

ETX

- Essentially captures expected number of MAC-layer transmissions needed to deliver a packet on a wireless link
- Weight of path = Sum of links
- Captures both path length and lossyness of path
- Misses interference and transmission rates

ETX ≈ 1/throughput



Packet loss	Link ETX	Throughput
0%	1	100%
50%	2	50%
66%	3	33%

ETX

- Measure each link's delivery probability with broadcast probes (& measure reverse)
- P(delivery) = 1 / (df * dr)   (ACK must be delivered too)
- Link ETX = 1 / P(delivery)
- Route ETX = sum of link ETX
- (Assumes all hops interfere - not true, but seems to work okay so far)

ETT

- Improves on ETX by considering transmission rates
- ETT = ETX*(packet-size/link-bandwidth)
- Captures all but interference
  - No channel diversity for example

Capacity of multi-hop network

- Assume N nodes, each wants to talk to everyone else. What total throughput (ignore previous slide to simplify things)
  - O(n) concurrent transmissions. Great! But:
  - Each has length O(sqrt(n)) (network diameter)
  - So each Tx uses up sqrt(n) of the O(n) capacity.
  - Per-node capacity scales as 1/sqrt(n)
    - Yes - it goes down! More time spent Tx'ing other peoples packets…
- But: If communication is local, can do much better, and use cool tricks to optimize
  - Like multicast, or multicast in reverse (data fusion)
  - Hey, that sounds like … a sensor network!

Sensor Networks - smart devices

- First introduced in late 90's by groups at UCB/UCLA/USC
- Small, resource limited devices
  - CPU, disk, power, bandwidth, etc.
- Simple scalar sensors – temperature, motion
- Single domain of deployment
  - farm, battlefield, bridge, rain forest
- for a targeted task
  - find the tanks, count the birds, monitor the bridge
- Ad-hoc wireless network

Sensor System Types – Smart-Dust/Motes

- Hardware
  - UCB motes
  - 4 MHz CPU
  - 4 kB data RAM
  - 128 kB code
  - 50 kb/sec 917 Mhz radio
  - Sensors: light, temp.,
    - Sound, etc.,
  - And a battery.




Sensors and power and radios

- Limited battery life drives most goals
- Radio is most energy-expensive part.
- 800 instructions per bit. 200,000 instructions per packet. (!)
- That's about one message per second for ~2 months if no CPU.
- Listening is expensive too. :(

Sensor nets goals

- Replace communication with computation
- Turn off radio receiver as often as possible
- Keep little state (4 KB isn't your pentium 4 ten bazillion gigahertz with five ottabytes of DRAM).

Power

- Which uses less power?
  - Direct sensor -> base station Tx
    - Total Tx power: distance^2
  - Sensor -> sensor -> sensor -> base station?
    - Total Tx power: n * (distance/n) ^2 =~ d^2 / n
  - Why? Radios are omnidirectional, but only one direction matters. Multi-hop approximates directionality.
- Power savings often makes up for multi-hop capacity
  - These devices are *very* power constrained!
- Reality: Many systems don't use adaptive power control. This is active research, and fun stuff.

Example: Aggregation

- Find avg temp in 8th floor of Wean.
- Strawman:
  - Flood query, let a collection point compute avg.
    - Huge overload near the CP. Lots of loss, and local nodes use lots of energy!
- Better:
  - Take local avg. first, & forward that.
    - Send average temp + # of samples
  - Aggregation is the key to scaling these nets.
- The challenge: How to aggregate.
  - How long to wait?
  - How to aggregate complex queries?
  - How to program?

Programmable Matter (or, Seth's interest in ad-hoc networks)