Mobile Ad Hoc Networks: Routing, MAC and Transport Issues

Nitin H. Vaidya
University of Illinois at Urbana-Champaign

nhv@uiuc.edu
http://www.crhc.uiuc.edu/~nhv

Note

Due to the time constraints, only a subset of the slides will be discussed during the INFOCOM 2006 tutorial presentation

Notes

Names in brackets, as in [Xyz00], refer to a document in the list of references

The handout may not be as readable as the original slides, since the slides contain colored text and figures

- Note that different colors in the colored slides may look identical in the black-and-white handout

Statutory Warnings

Only most important features of various schemes are typically discussed, *i.e, features I consider as being important*

- Others may disagree

Most schemes include many more details, and optimizations

- Not possible to cover all details in this tutorial

Be aware that some protocol specs have changed several times, and the slides may not reflect the most current specifications

Jargon used to discuss a scheme may occasionally differ from that used by the proposers

Coverage

Not intended to be exhaustive

Many interesting papers not covered in the tutorial due to lack of time

Tutorial Outline

Introduction

Unicast routing

Medium Access Control

Performance of UDP and TCP

Selected security issues

Implementation Issues

Standards activities

Open problems

Mobile Ad Hoc Networks (MANET)

Introduction and Generalities

Mobile Ad Hoc Networks

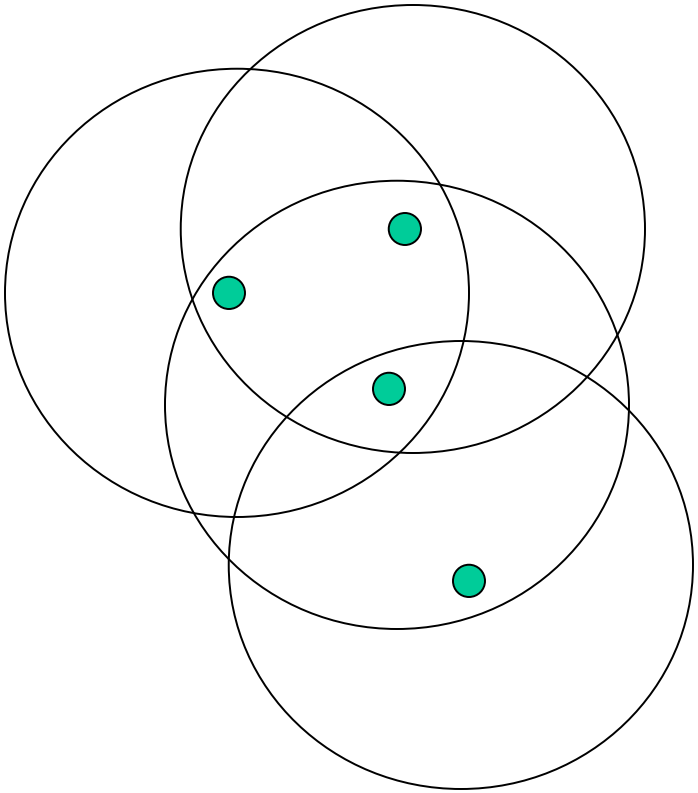
Formed by wireless hosts which may be mobile

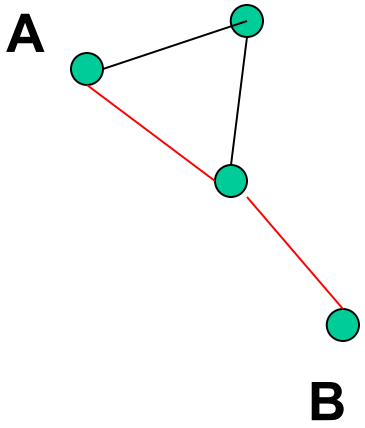
Without (necessarily) using a pre-existing infrastructure

Routes between nodes may potentially contain multiple hops

Mobile Ad Hoc Networks

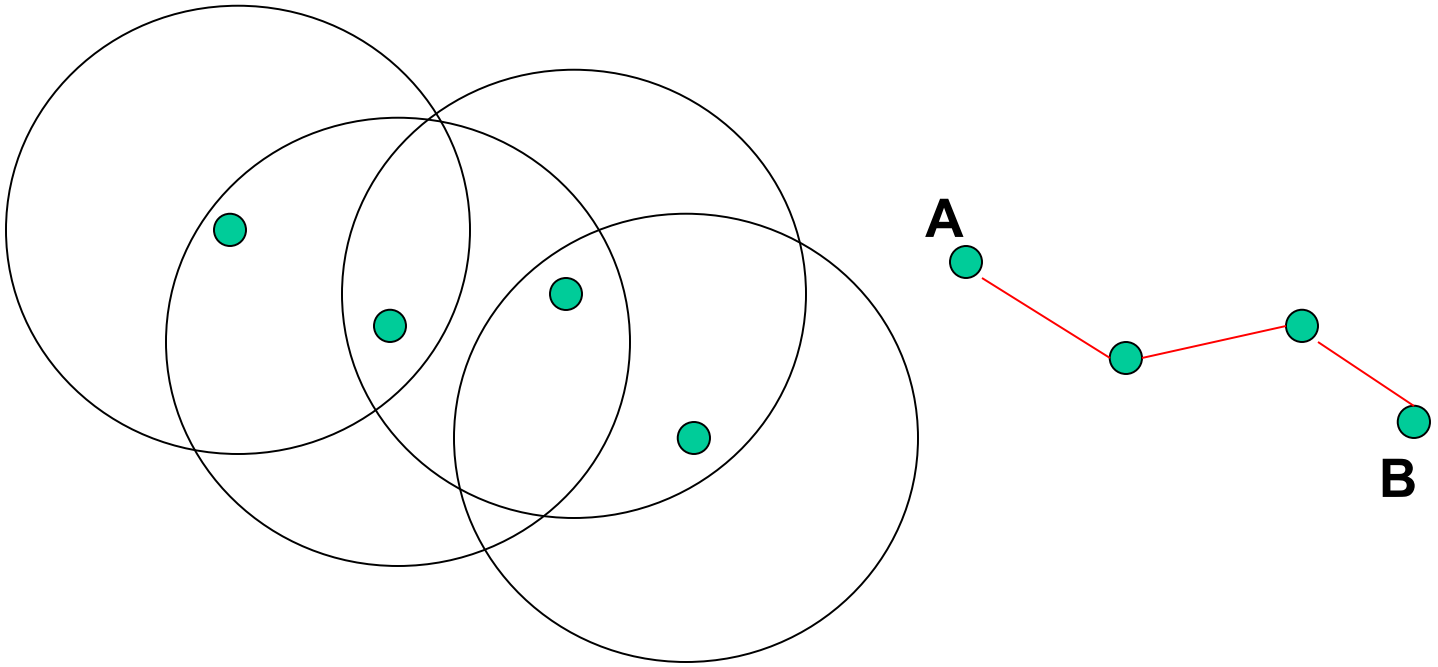
May need to traverse multiple links to reach a destination





Mobile Ad Hoc Networks (MANET)

Mobility causes route changes



Why Ad Hoc Networks ?

Ease of deployment

Speed of deployment

Decreased dependence on infrastructure

Many Applications

- Personal area networking
  - cell phone, laptop, ear phone, wrist watch
- Military environments
  - soldiers, tanks, planes
- Civilian environments
  - Mesh networks
  - taxi cab network
  - meeting rooms
  - sports stadiums
  - boats, small aircraft
- Emergency operations
  - search-and-rescue
  - policing and fire fighting

Many Variations

Fully Symmetric Environment

- all nodes have identical capabilities and responsibilities

Asymmetric Capabilities

- transmission ranges and radios may differ
- battery life at different nodes may differ
- processing capacity may be different at different nodes
- speed of movement

Asymmetric Responsibilities

- only some nodes may route packets
- some nodes may act as leaders of nearby nodes (e.g., cluster head)

Many Variations

Traffic characteristics may differ in different ad hoc networks

- bit rate
- timeliness constraints
- reliability requirements
- unicast / multicast / geocast
- host-based addressing / content-based addressing / capability-based addressing

May co-exist (and co-operate) with an infrastructure-based network

Many Variations

Mobility patterns may be different

- people sitting at an airport lounge
- New York taxi cabs
- kids playing
- military movements
- personal area network

Mobility characteristics

- speed
- predictability
  - direction of movement
  - pattern of movement
- uniformity (or lack thereof) of mobility characteristics among different nodes

Challenges

Limited wireless transmission range

Broadcast nature of the wireless medium

- Hidden terminal problem (see next slide)

Packet losses due to transmission errors

Mobility-induced route changes

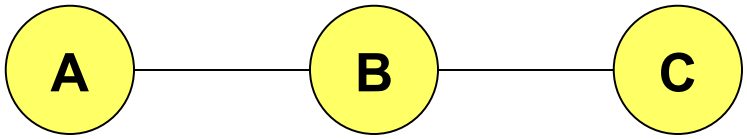
Mobility-induced packet losses

Battery constraints

Potentially frequent network partitions

Ease of snooping on wireless transmissions (security hazard)

Hidden Terminal Problem



Nodes A and C cannot hear each other

Transmissions by nodes A and C can collide at node B

Nodes A and C are hidden from each other

Research on Mobile Ad Hoc Networks

Variations in capabilities & responsibilities

X

Variations in traffic characteristics, mobility models, etc.

X

Performance criteria (e.g., optimize throughput, reduce energy consumption)

+

Increased research funding

=

Significant research activity

The Holy Grail

A one-size-fits-all solution

- Perhaps using an adaptive/hybrid approach that can adapt to situation at hand

Difficult problem

Many solutions proposed trying to address a sub-space of the problem domain

Assumption

Unless stated otherwise, fully symmetric environment is assumed implicitly

- all nodes have identical capabilities and responsibilities

Unicast Routing
in
Mobile Ad Hoc Networks

Why is Routing in MANET different ?

Host mobility

- link failure/repair due to mobility may have different characteristics than those due to other causes

Rate of link failure/repair may be high when nodes move fast

New performance criteria may be used

- route stability despite mobility
- energy consumption

Unicast Routing Protocols

Many protocols have been proposed

Some have been invented specifically for MANET

Others are adapted from previously proposed protocols for wired networks

No single protocol works well in all environments

- some attempts made to develop adaptive protocols

Routing Protocols

Proactive protocols

- Determine routes independent of traffic pattern
- Traditional link-state and distance-vector routing protocols are proactive

Reactive protocols

- Maintain routes only if needed

Hybrid protocols

Trade-Off

Latency of route discovery

- Proactive protocols may have lower latency since routes are maintained at all times
- Reactive protocols may have higher latency because a route from X to Y will be found only when X attempts to send to Y

Overhead of route discovery/maintenance

- Reactive protocols may have lower overhead since routes are determined only if needed
- Proactive protocols can (but not necessarily) result in higher overhead due to continuous route updating

Which approach achieves a better trade-off depends on the traffic and mobility patterns

Overview of Unicast Routing Protocols

Flooding for Data Delivery

Sender S broadcasts data packet P to all its neighbors

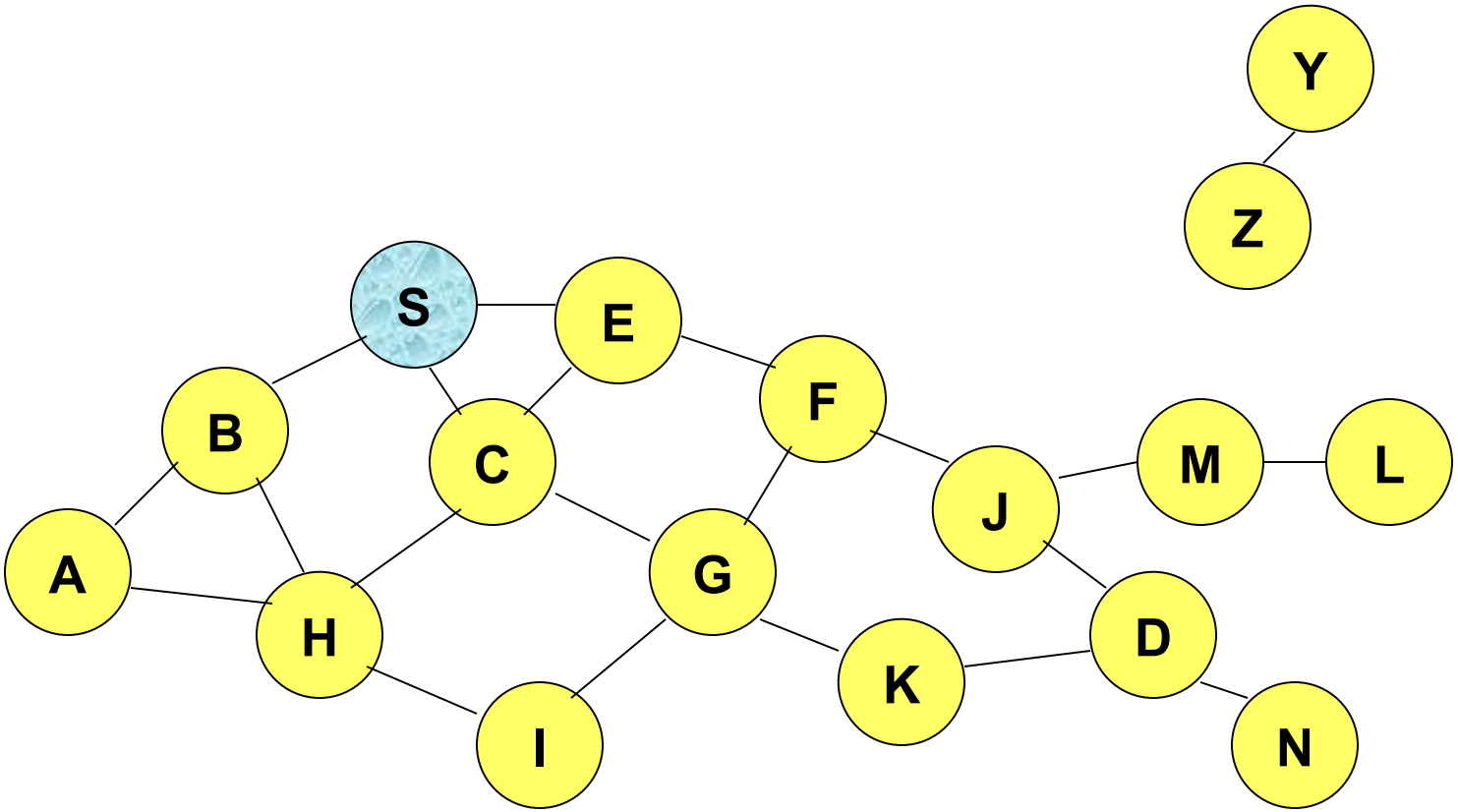
Each node receiving P forwards P to its neighbors

Sequence numbers used to avoid the possibility of forwarding the same packet more than once

Packet P reaches destination D provided that D is reachable from sender S

Node D does not forward the packet

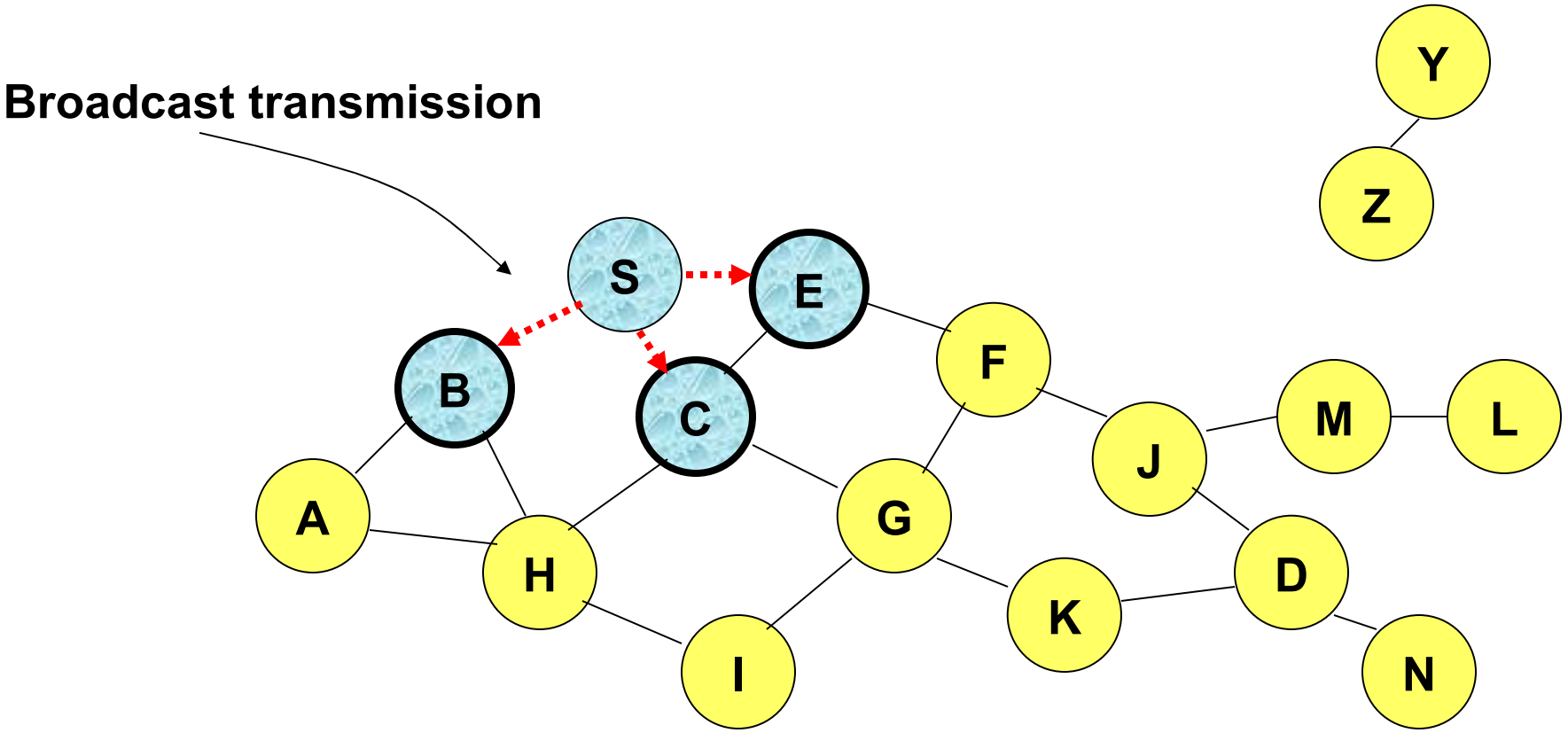
Flooding for Data Delivery



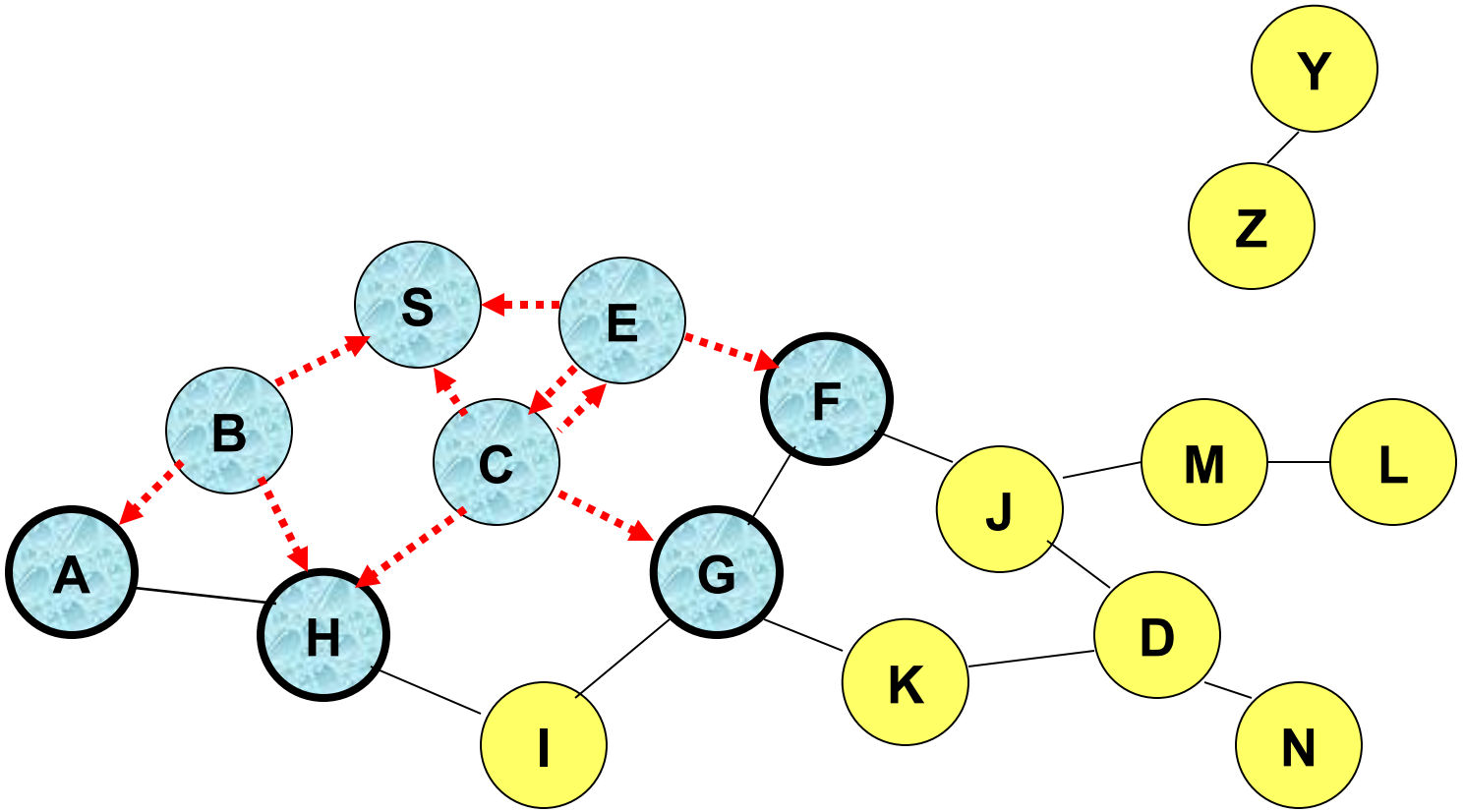
Represents a node that has received packet P

Represents that connected nodes are within each other's transmission range

Flooding for Data Delivery

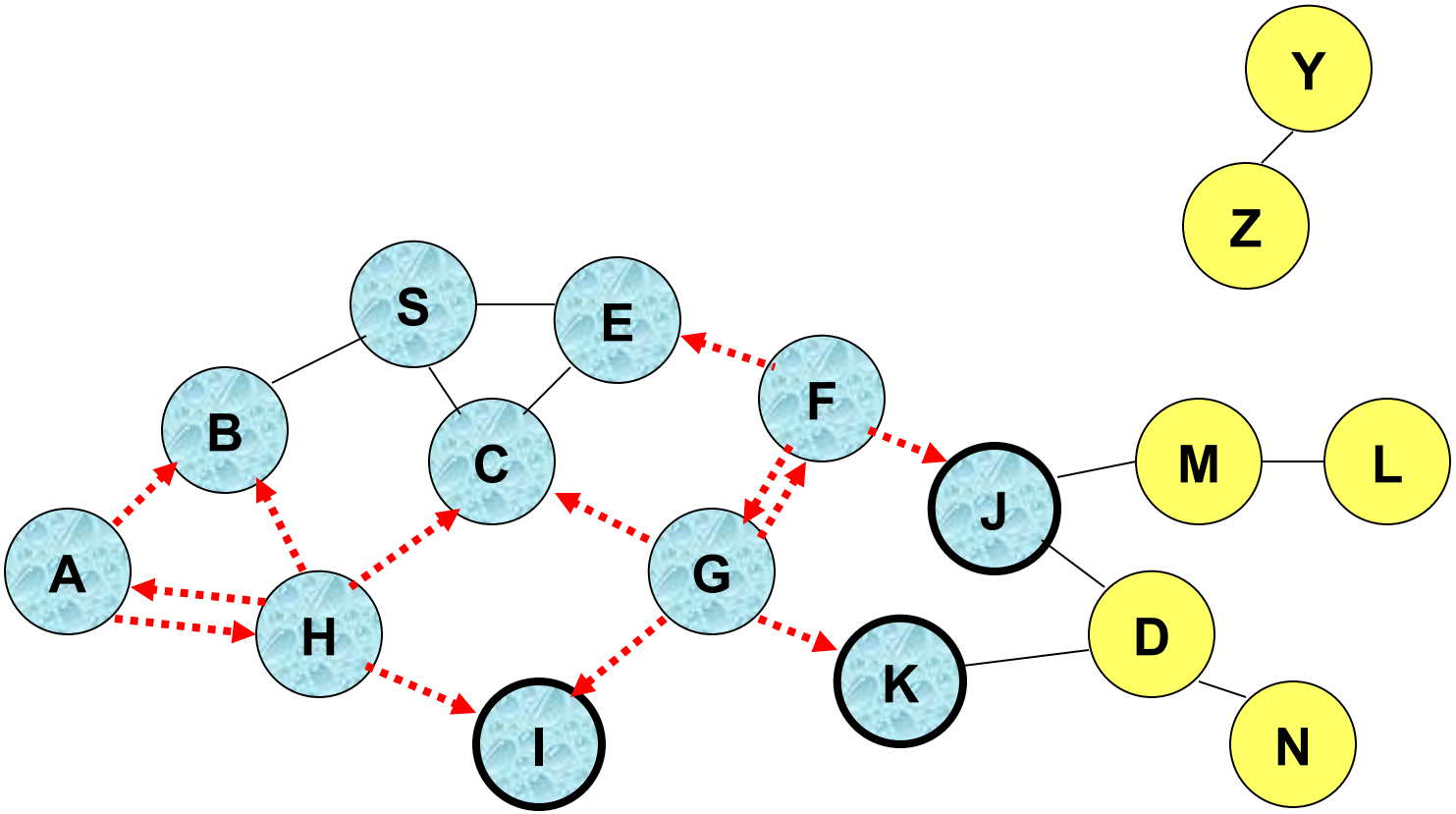


Flooding for Data Delivery



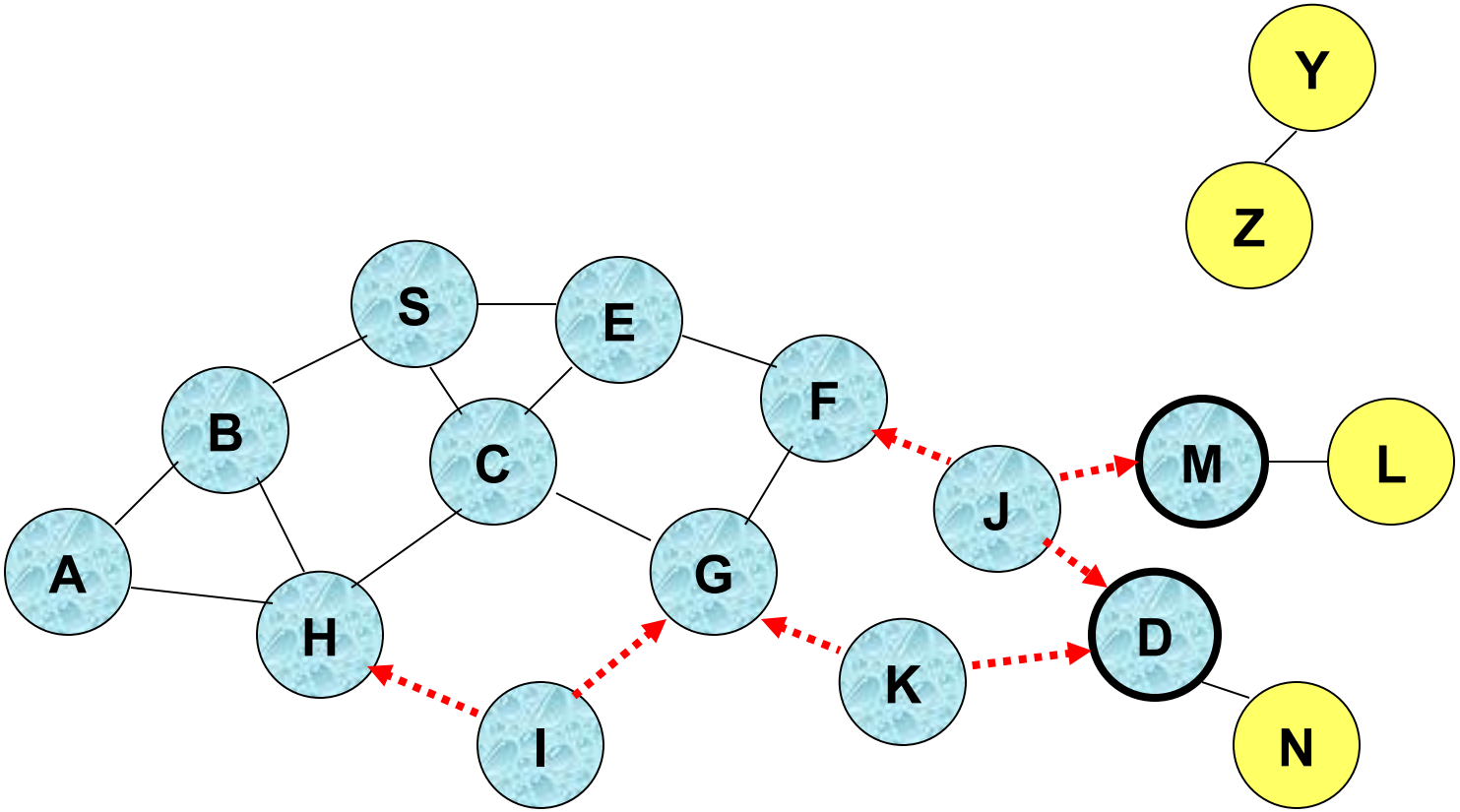
- **Node H receives packet P from two neighbors:**
  potential for collision

Flooding for Data Delivery



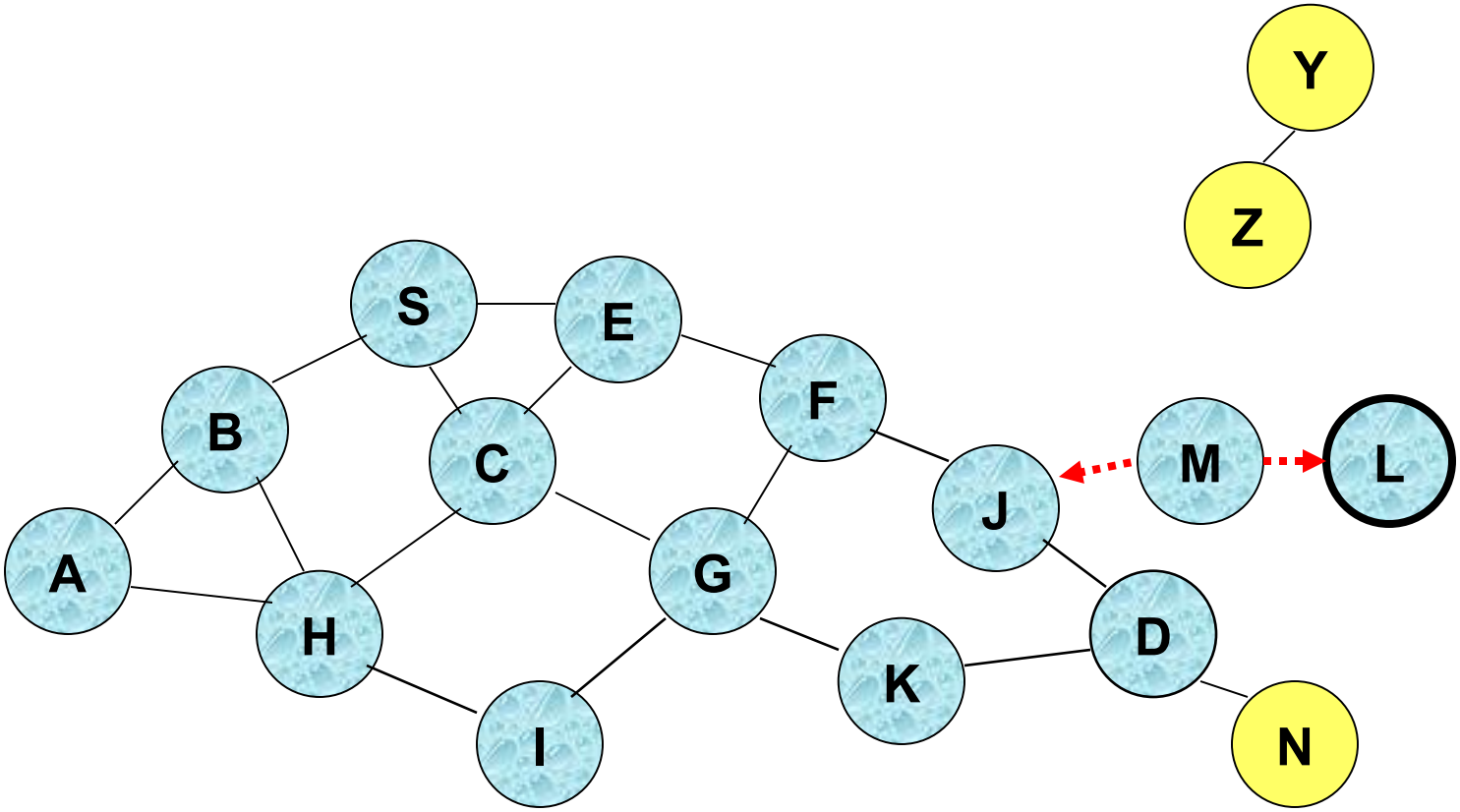
- **Node C receives packet P from G and H, but does not forward it again, because node C has already forwarded packet P once**

Flooding for Data Delivery



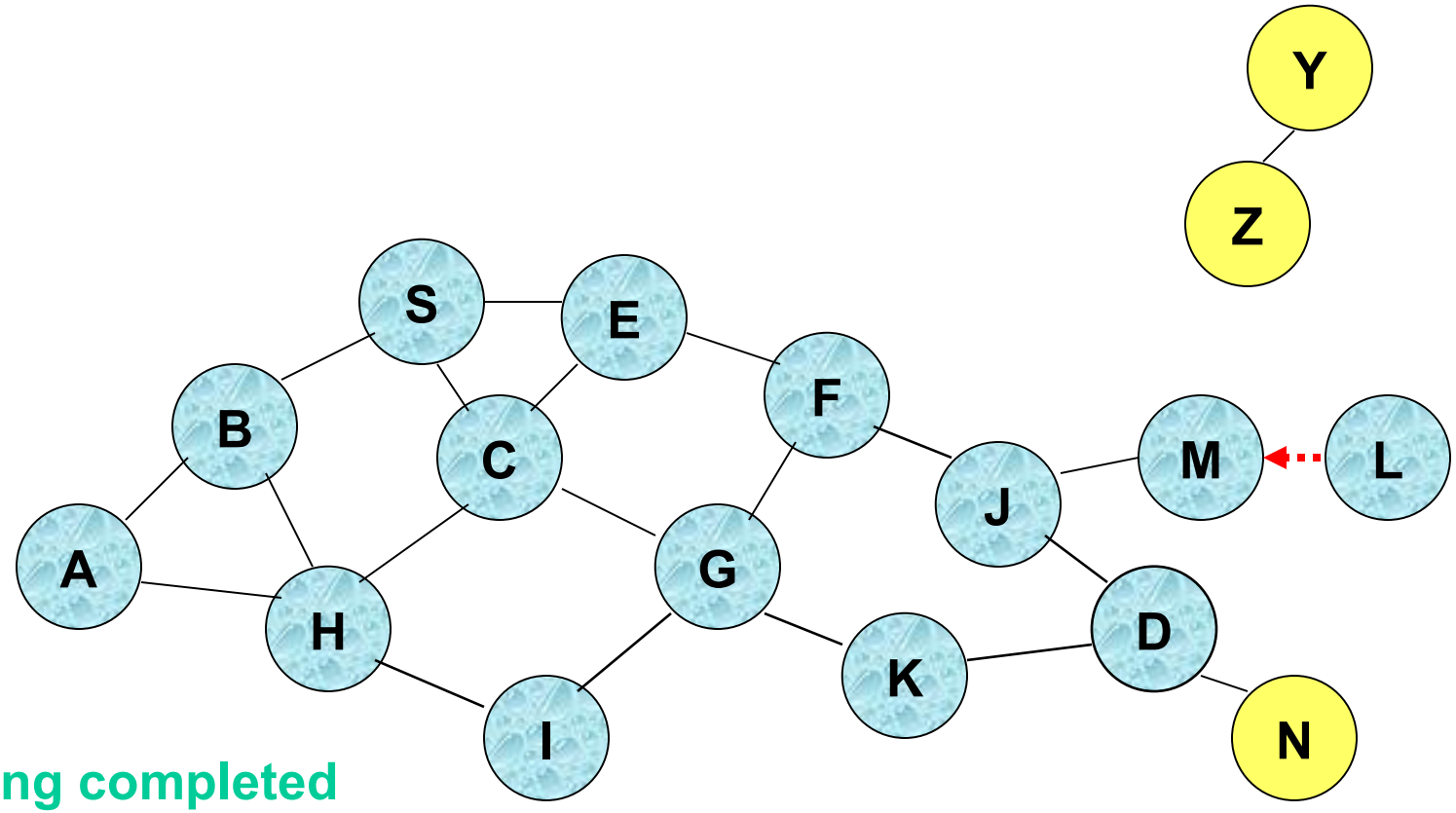
- Nodes J and K both broadcast packet P to node D
- Since nodes J and K are hidden from each other, their transmissions may collide
  - => Packet P may not be delivered to node D at all, despite the use of flooding

Flooding for Data Delivery



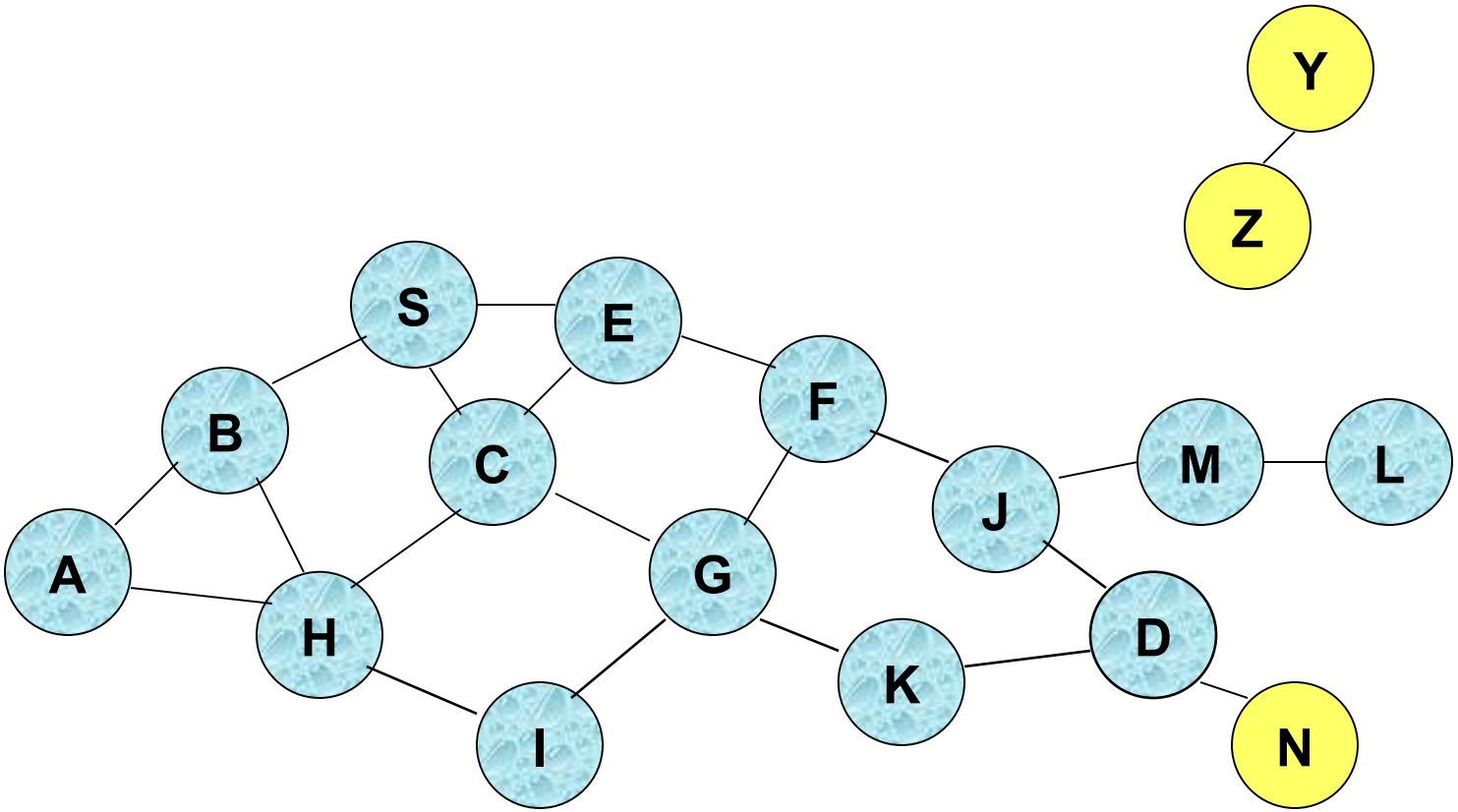
- **Node D does not forward packet P, because node D is the intended destination of packet P**

Flooding for Data Delivery



- **Flooding completed**
- **Nodes unreachable from S do not receive packet P (e.g., node Z)**
- **Nodes for which all paths from S go through the destination D also do not receive packet P (example: node N)**

Flooding for Data Delivery



- **Flooding may deliver packets to too many nodes (in the worst case, all nodes reachable from sender may receive the packet)**

Flooding for Data Delivery: Advantages

Simplicity

May be more efficient than other protocols when rate of information transmission is low enough that the overhead of explicit route discovery/maintenance incurred by other protocols is relatively higher

- this scenario may occur, for instance, when nodes transmit small data packets relatively infrequently, and many topology changes occur between consecutive packet transmissions

Potentially higher reliability of data delivery

- Because packets may be delivered to the destination on multiple paths

Flooding for Data Delivery: Disadvantages

Potentially, very high overhead

- Data packets may be delivered to too many nodes who do not need to receive them

Potentially lower reliability of data delivery

- Flooding uses broadcasting -- hard to implement reliable broadcast delivery without significantly increasing overhead
  - Broadcasting in IEEE 802.11 MAC is unreliable
- In our example, nodes J and K may transmit to node D simultaneously, resulting in loss of the packet
  - in this case, destination would not receive the packet at all

Flooding of Control Packets

Many protocols perform (potentially *limited*) flooding of control packets, instead of data packets

The control packets are used to discover routes

Discovered routes are subsequently used to send data packet(s)

Overhead of control packet flooding is amortized over data packets transmitted between consecutive control packet floods

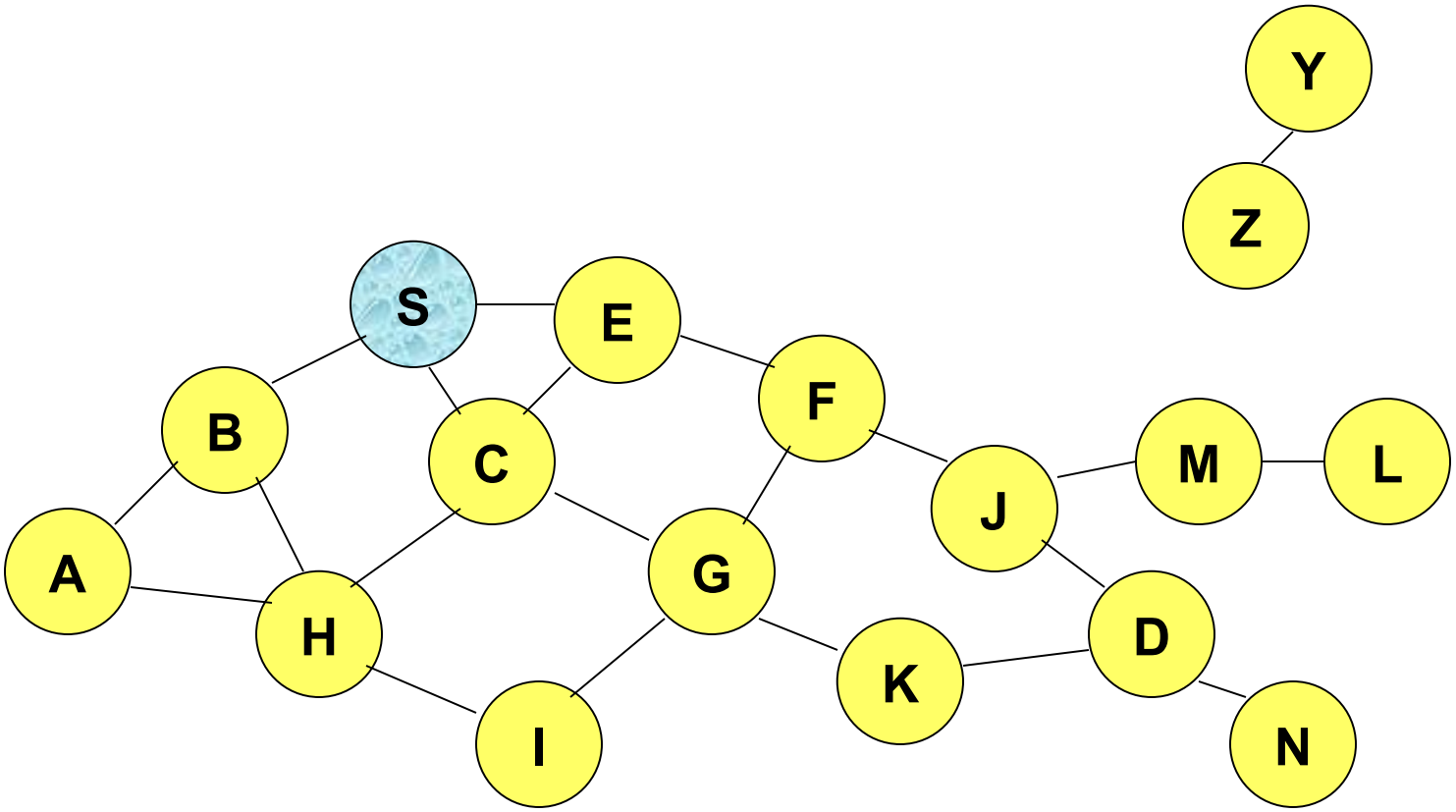
Dynamic Source Routing (DSR) [Johnson96]

When node S wants to send a packet to node D, but does not know a route to D, node S initiates a route discovery

Source node S floods Route Request (RREQ)

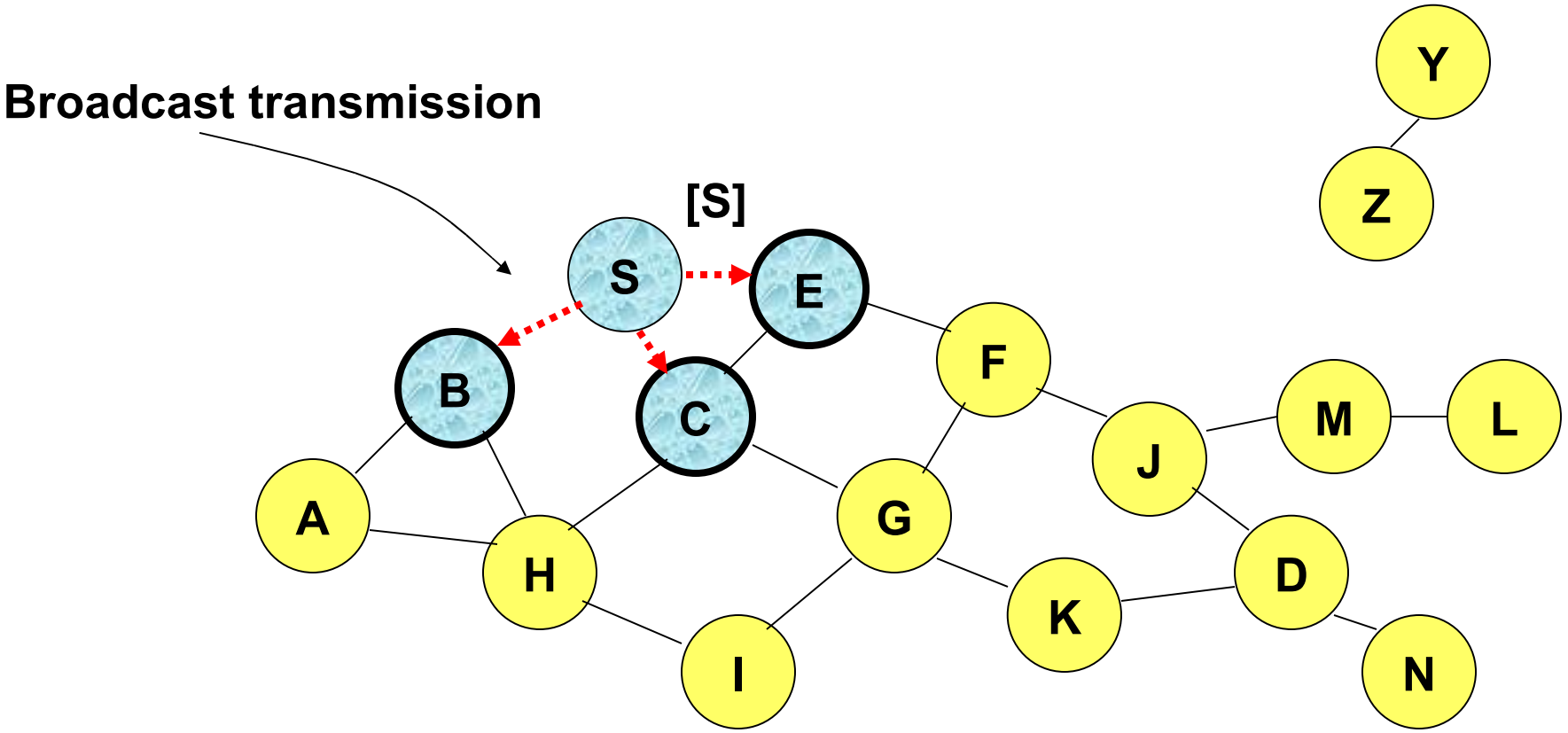
Each node appends own identifier when forwarding RREQ

Route Discovery in DSR



Represents a node that has received RREQ for D from S

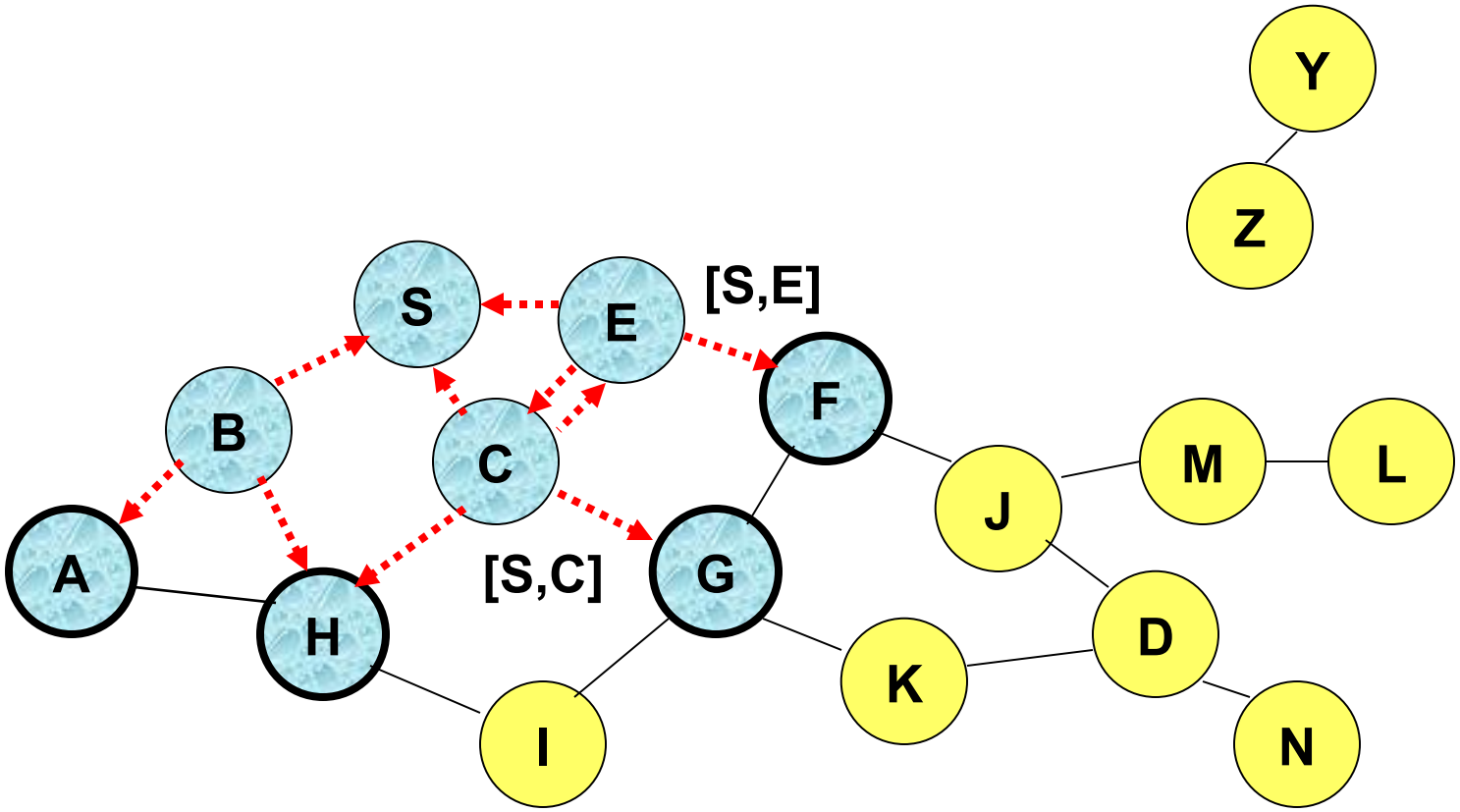
Route Discovery in DSR



Represents transmission of RREQ

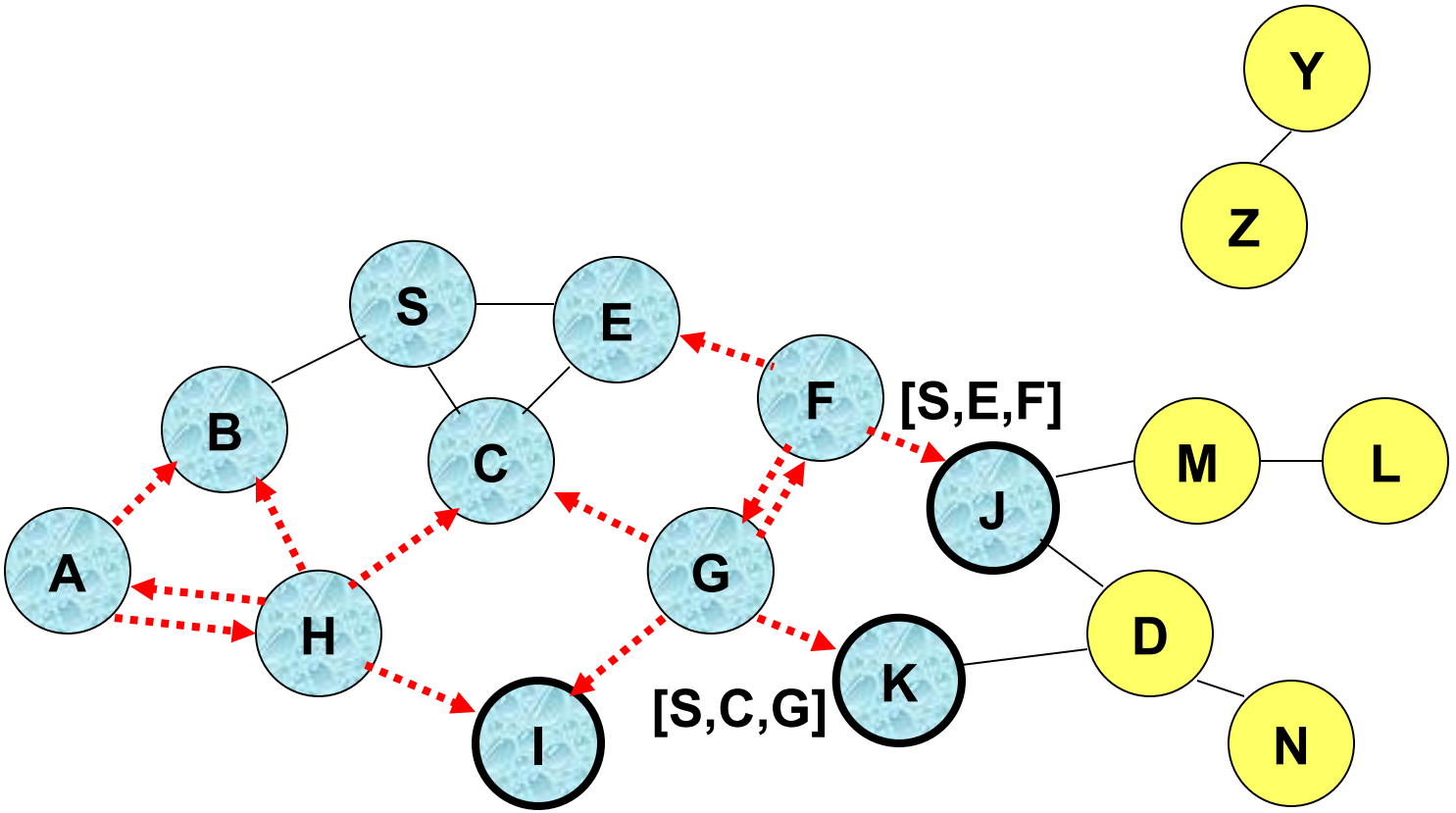
[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR



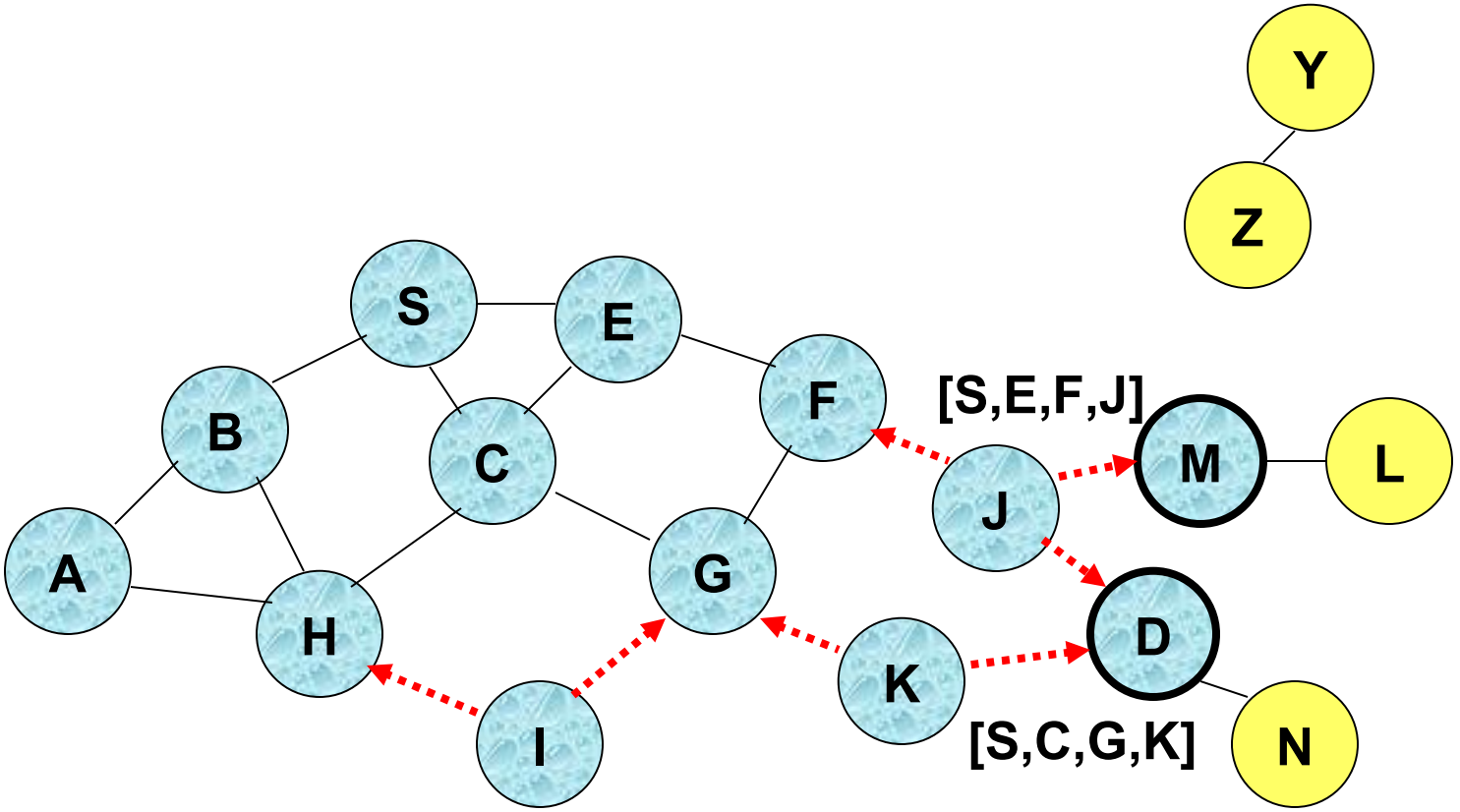
- **Node H receives packet RREQ from two neighbors:**
  potential for collision

Route Discovery in DSR



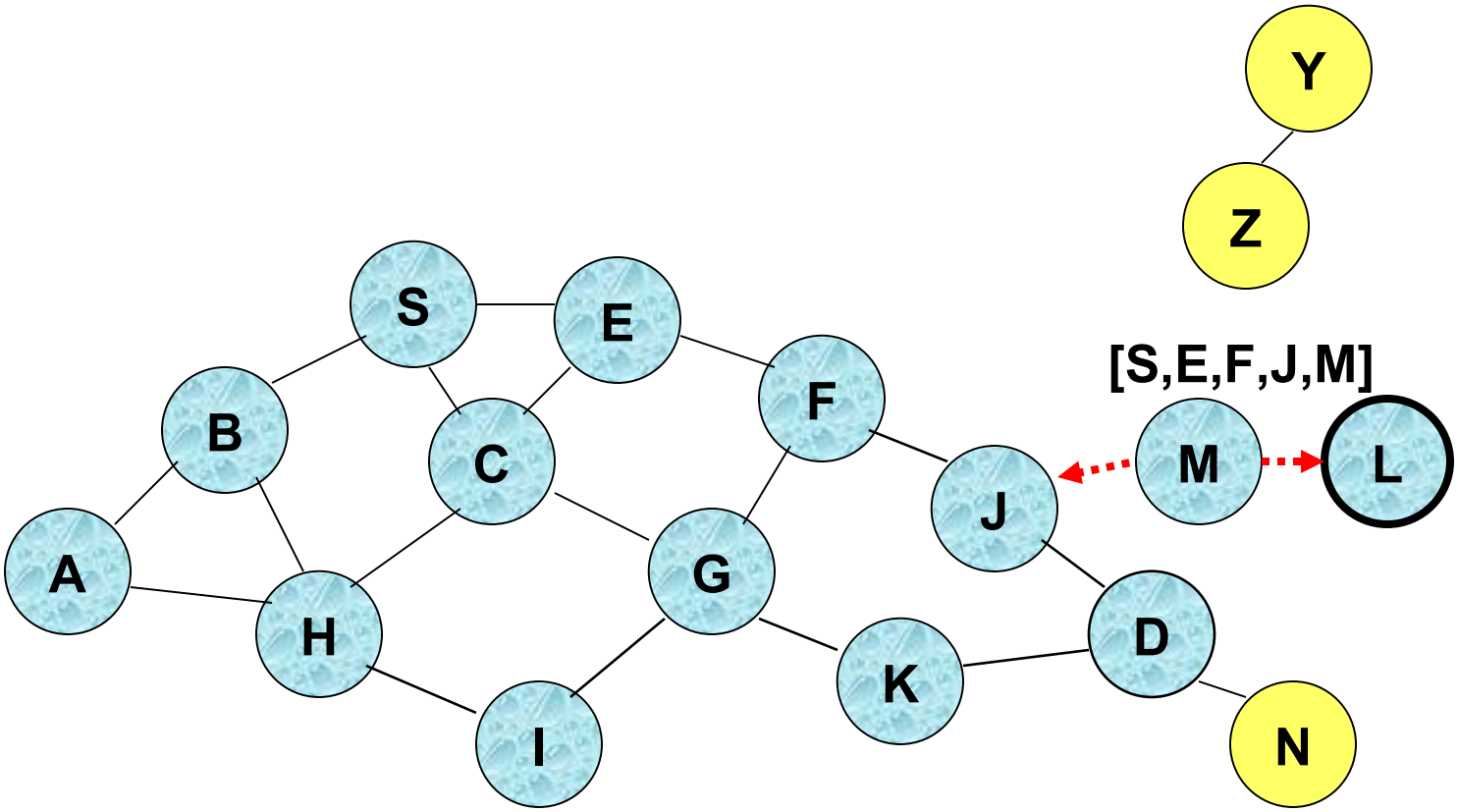
- **Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once**

Route Discovery in DSR



- Nodes J and K both broadcast RREQ to node D
- Since nodes J and K are hidden from each other, their transmissions may collide

Route Discovery in DSR



- Node D **does not forward** RREQ, because node D is the **intended target** of the route discovery

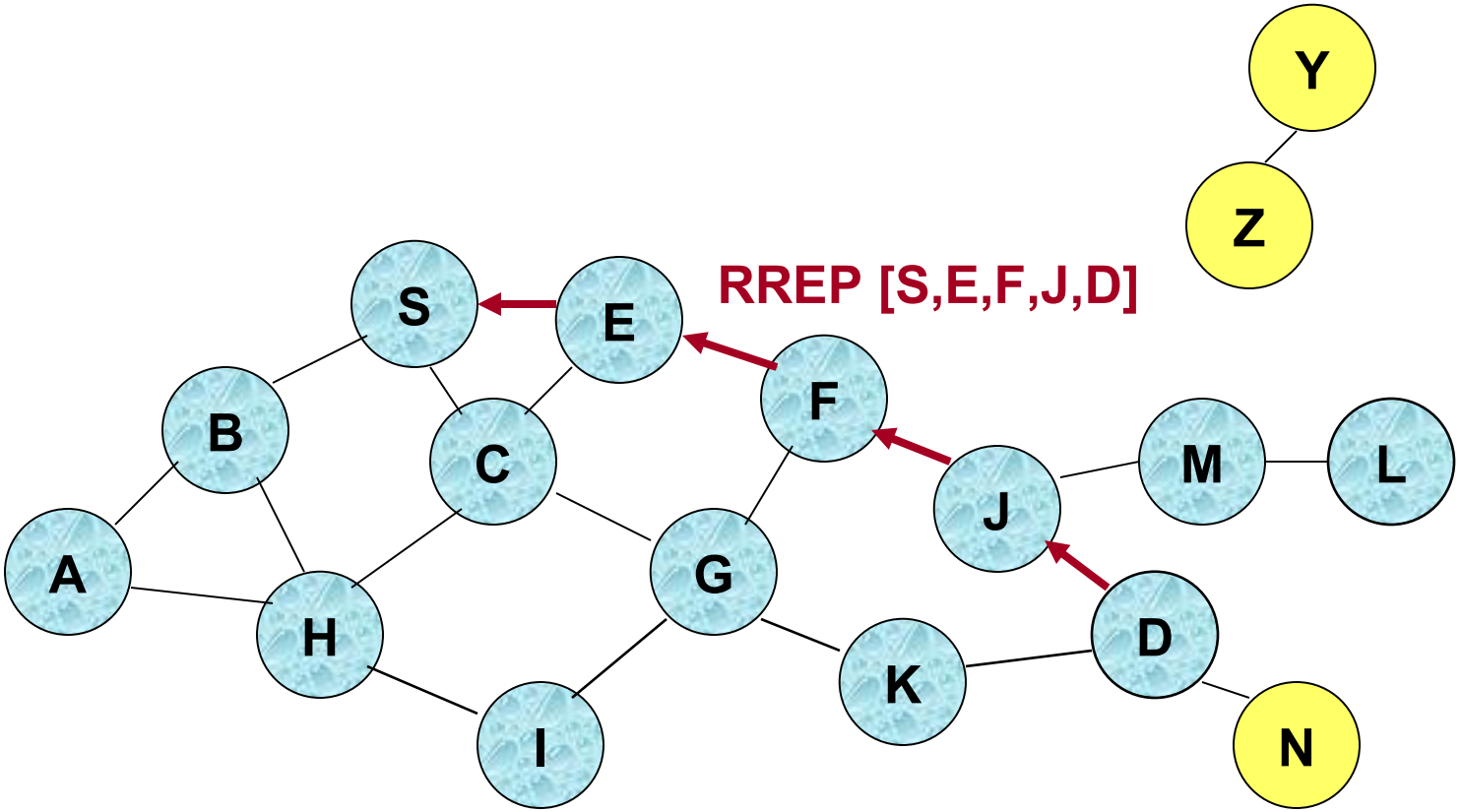
Route Discovery in DSR

Destination D on receiving the first RREQ, sends a Route Reply (RREP)

RREP is sent on a route obtained by reversing the route appended to received RREQ

RREP includes the route from S to D on which RREQ was received by node D

Route Reply in DSR



Represents RREP control message

Route Reply in DSR

Route Reply can be sent by reversing the route in Route Request (RREQ) only if links are guaranteed to be bi-directional

- To ensure this, RREQ should be forwarded only if it received on a link that is known to be bi-directional

If unidirectional (asymmetric) links are allowed, then RREP may need a route discovery for S from node D

- Unless node D already knows a route to node S
- If a route discovery is initiated by D for a route to S, then the Route Reply is piggybacked on the Route Request from D.

If IEEE 802.11 MAC is used to send data, then links have to be bi-directional (since Ack is used)

Dynamic Source Routing (DSR)

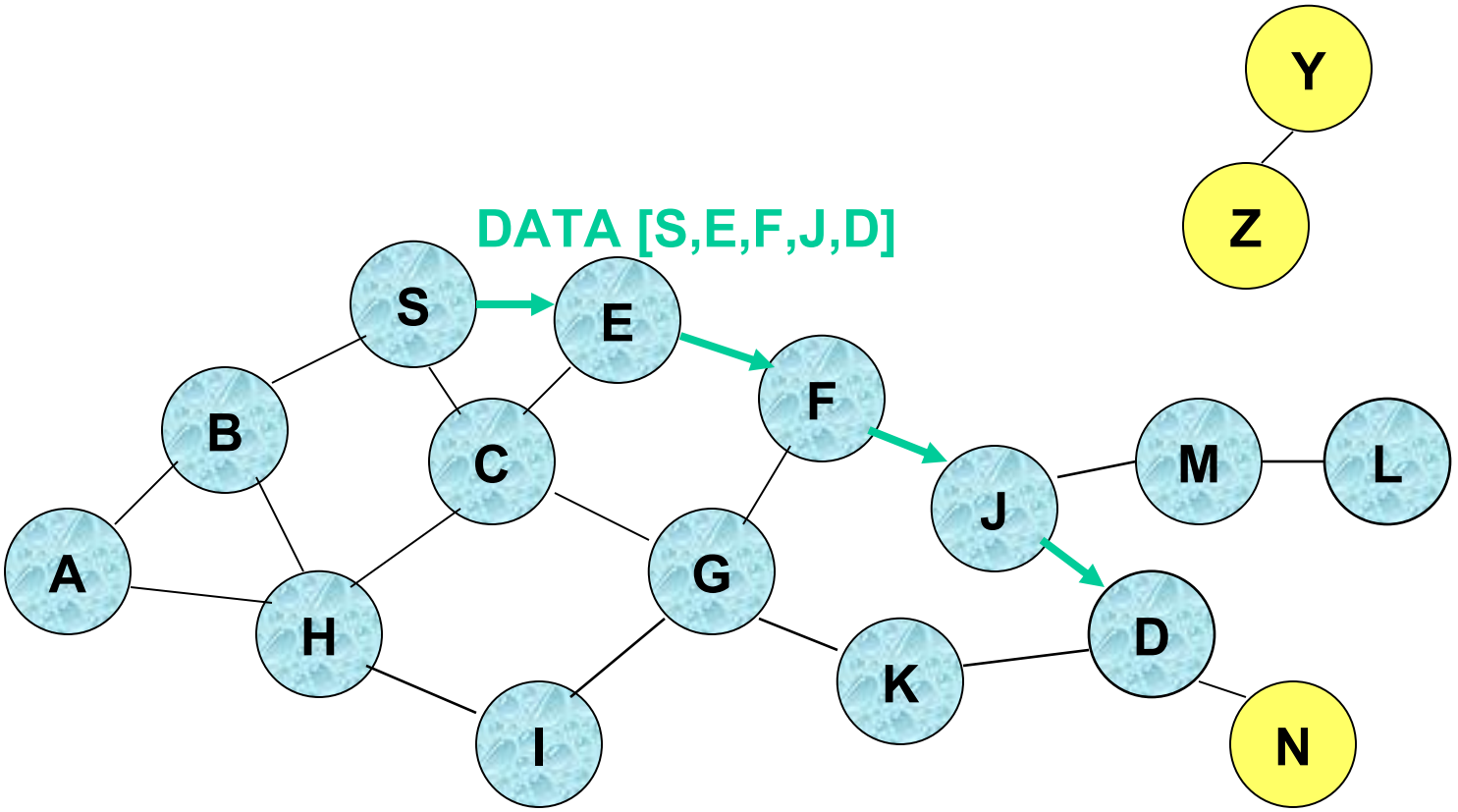
Node S on receiving RREP, caches the route included in the RREP

When node S sends a data packet to D, the entire route is included in the packet header

hence the name source routing

Intermediate nodes use the source route included in a packet to determine to whom a packet should be forwarded

Data Delivery in DSR



Packet header size grows with route length

When to Perform a Route Discovery

When node S wants to send data to node D, but does not know a valid route node D

DSR Optimization: Route Caching

Each node caches a new route it learns by *any means*

When node S finds route [S,E,F,J,D] to node D, node S also learns route [S,E,F] to node F

When node K receives Route Request [S,C,G] destined for node, node K learns route [K,G,C,S] to node S

When node F forwards Route Reply RREP [S,E,F,J,D], node F learns route [F,J,D] to node D

When node E forwards Data [S,E,F,J,D] it learns route [E,F,J,D] to node D

A node may also learn a route when it overhears Data packets

Use of Route Caching

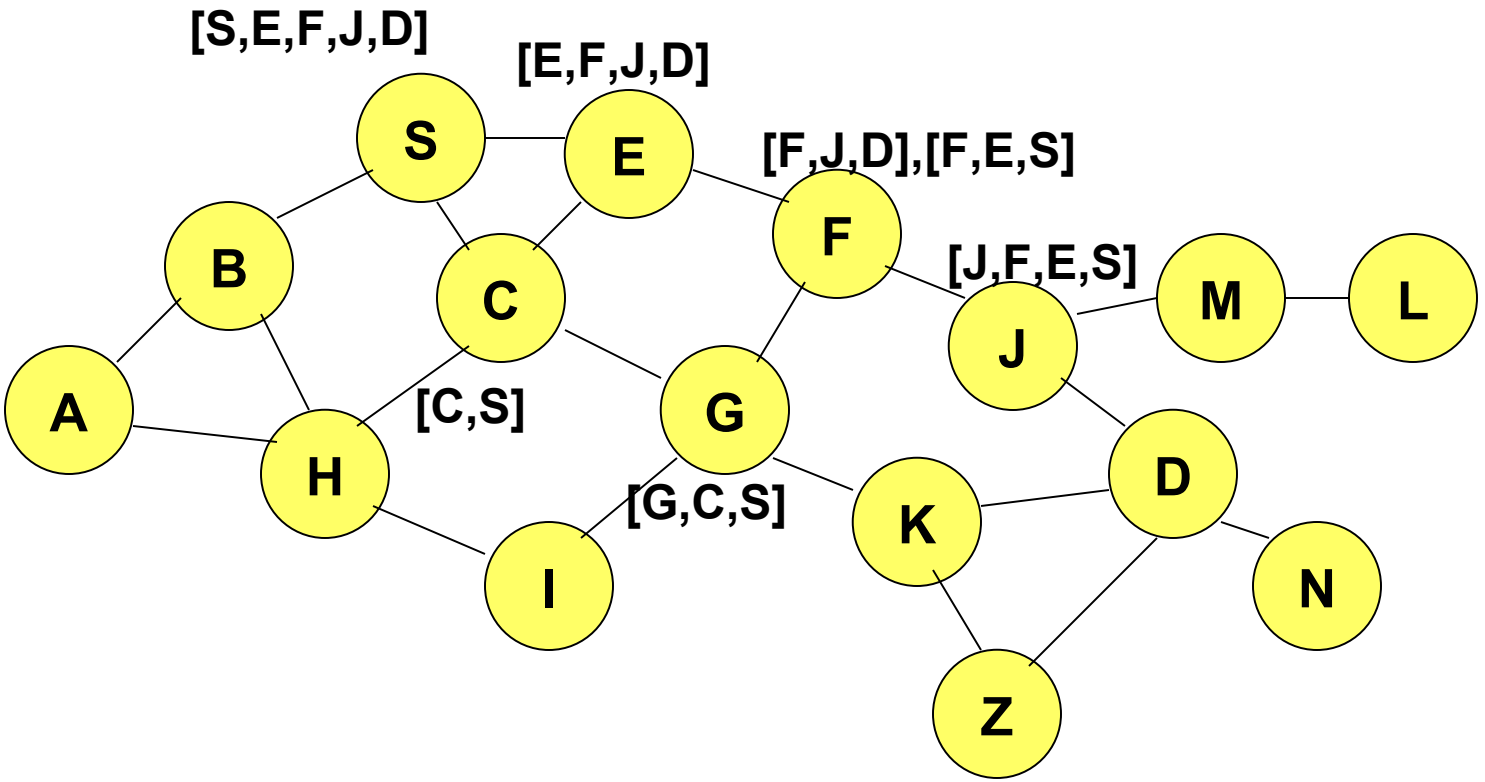
When node S learns that a route to node D is broken, it uses another route from its local cache, if such a route to D exists in its cache. Otherwise, node S initiates route discovery by sending a route request

Node X on receiving a Route Request for some node D can send a Route Reply if node X knows a route to node D

Use of route cache

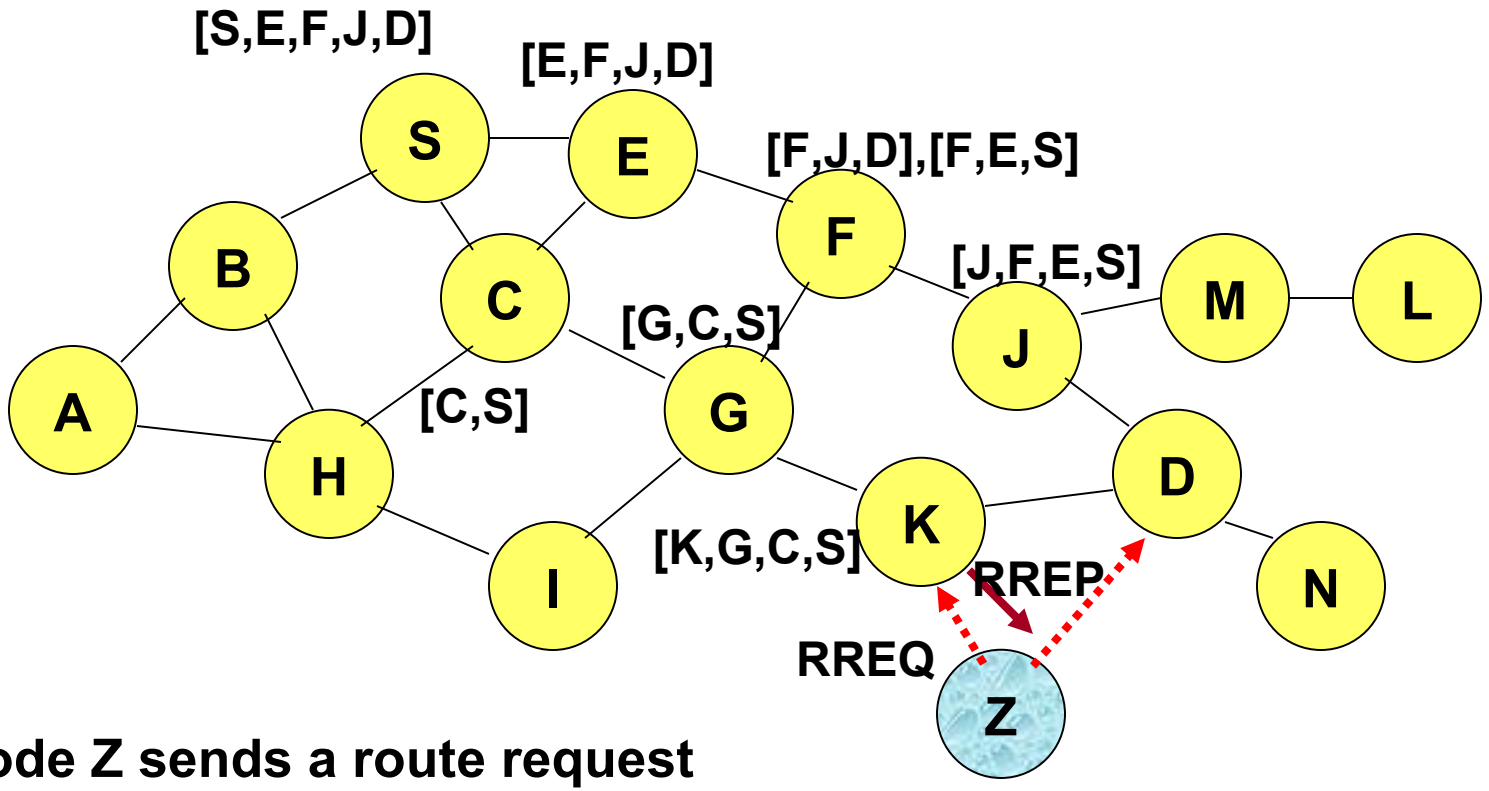
- can speed up route discovery
- can reduce propagation of route requests

Use of Route Caching



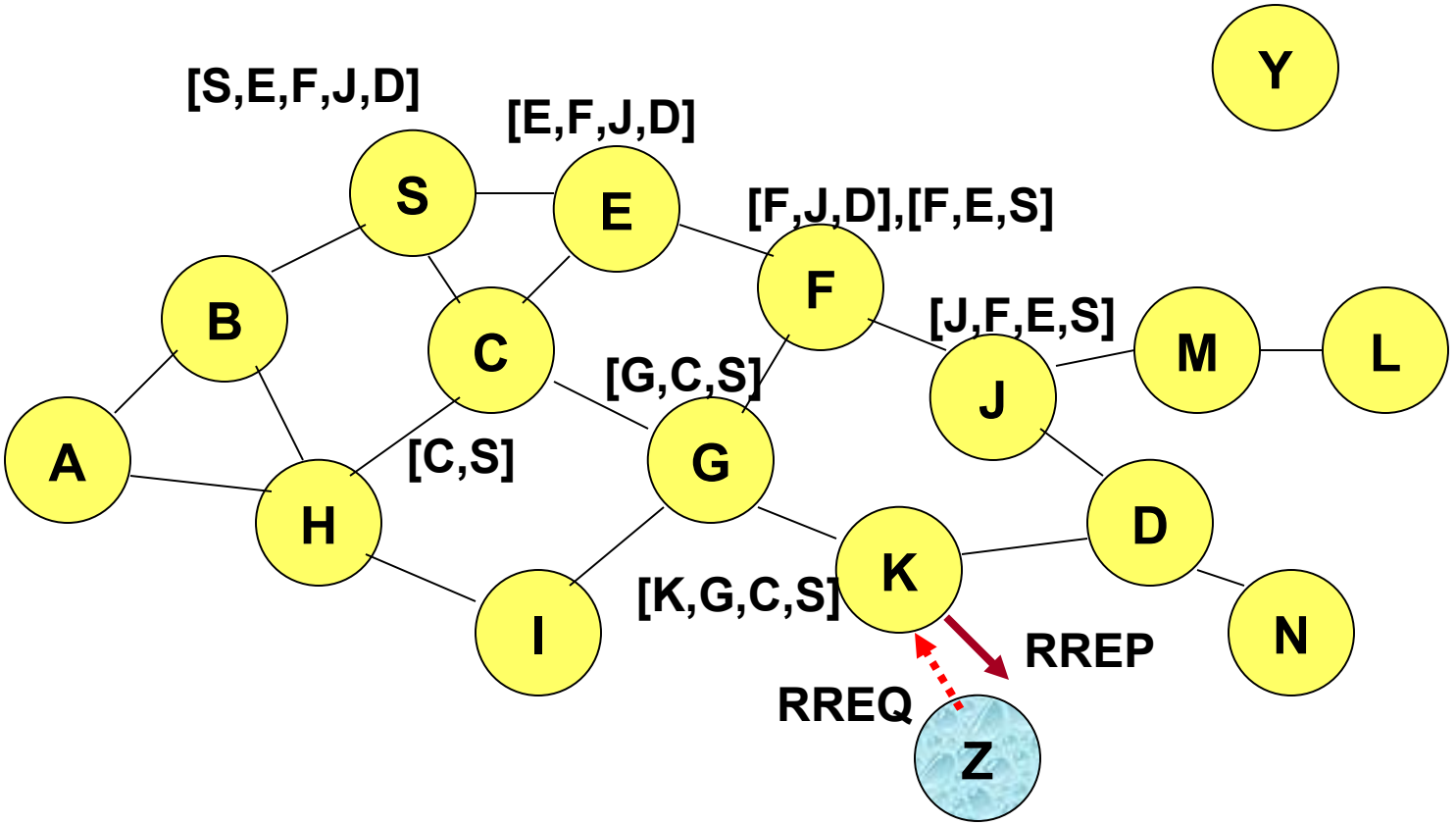
[P,Q,R] Represents cached route at a node
(DSR maintains the cached routes in a tree format)

Use of Route Caching:
Can Speed up Route Discovery



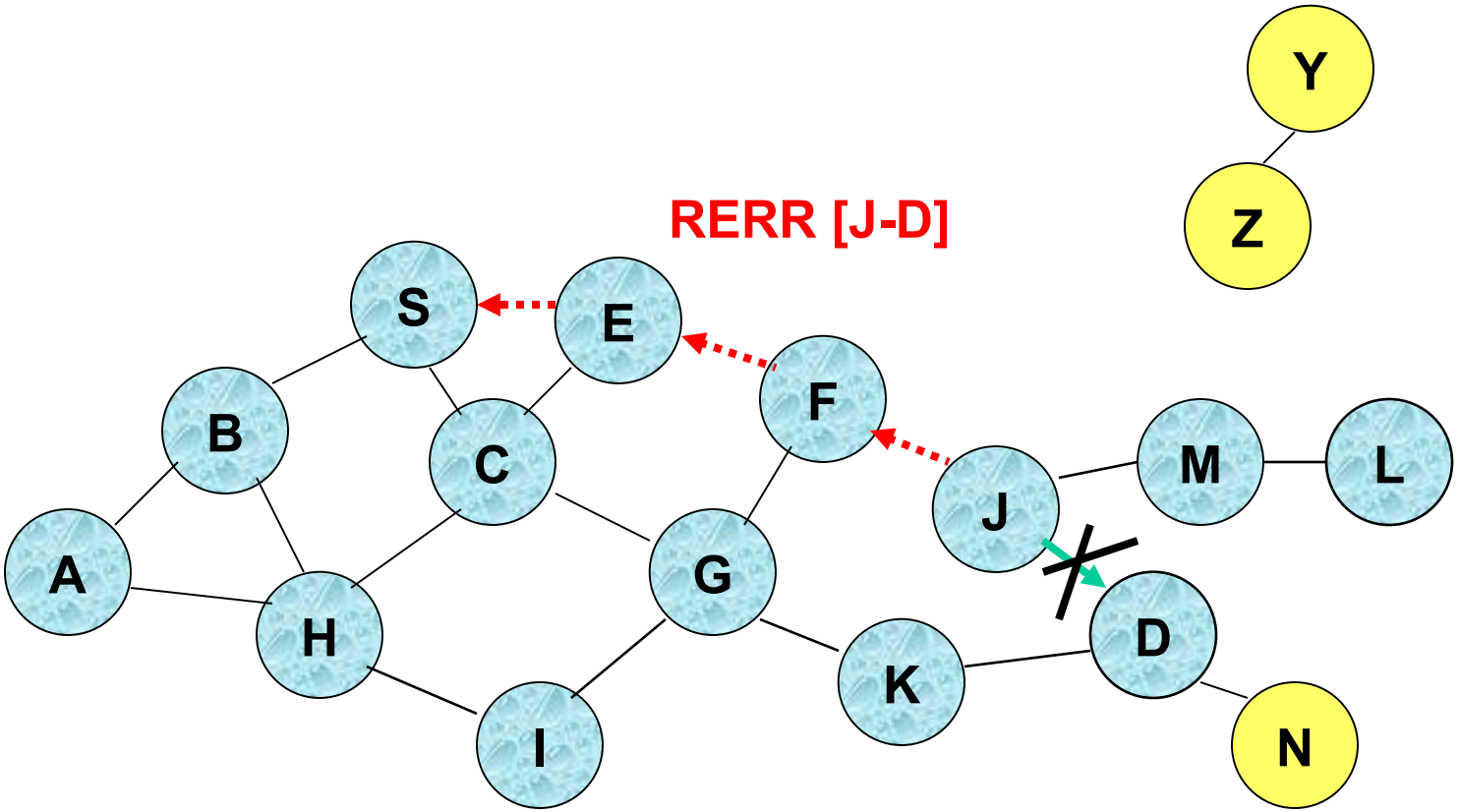
When node Z sends a route request for node C, node K sends back a route reply [Z,K,G,C] to node Z using a locally cached route

Use of Route Caching:
Can Reduce Propagation of Route Requests



Assume that there is no link between D and Z.
Route Reply (RREP) from node K limits flooding of RREQ.
In general, the reduction may be less dramatic.

Route Error (RERR)



J sends a route error to S along route J-F-E-S when its attempt to forward the data packet S (with route SEFJD) on J-D fails

Nodes hearing RERR update their route cache to remove link J-D

Route Caching: Beware!

Stale caches can adversely affect performance

With passage of time and host mobility, cached routes may become invalid

A sender host may try several stale routes (obtained from local cache, or replied from cache by other nodes), before finding a good route

An illustration of the adverse impact on TCP will be discussed later in the tutorial [Holland99]

Dynamic Source Routing: Advantages

Routes maintained only between nodes who need to communicate

- reduces overhead of route maintenance

Route caching can further reduce route discovery overhead

A single route discovery may yield many routes to the destination, due to intermediate nodes replying from local caches

Dynamic Source Routing: Disadvantages

Packet header size grows with route length due to source routing

Flood of route requests may potentially reach all nodes in the network

Care must be taken to avoid collisions between route requests propagated by neighboring nodes

- insertion of random delays before forwarding RREQ

Increased contention if too many route replies come back due to nodes replying using their local cache

- Route Reply *Storm* problem
- Reply storm may be eased by preventing a node from sending RREP if it hears another RREP with a shorter route

Dynamic Source Routing: Disadvantages

An intermediate node may send Route Reply using a stale cached route, thus polluting other caches

This problem can be eased if some mechanism to purge (potentially) invalid cached routes is incorporated.

For some proposals for cache invalidation, see [Hu00Mobicom]

- Static timeouts
- Adaptive timeouts based on link stability

Flooding of Control Packets

How to reduce the scope of the route request flood ?

- LAR [Ko98Mobicom]
- Query localization [Castaneda99Mobicom]

How to reduce redundant broadcasts ?

- The Broadcast Storm Problem [Ni99Mobicom]

Location-Aided Routing (LAR) [Ko98Mobicom]

Exploits location information to limit scope of route request flood

- Location information may be obtained using GPS

Expected Zone is determined as a region that is expected to hold the current location of the destination

- Expected region determined based on potentially old location information, and knowledge of the destination's speed

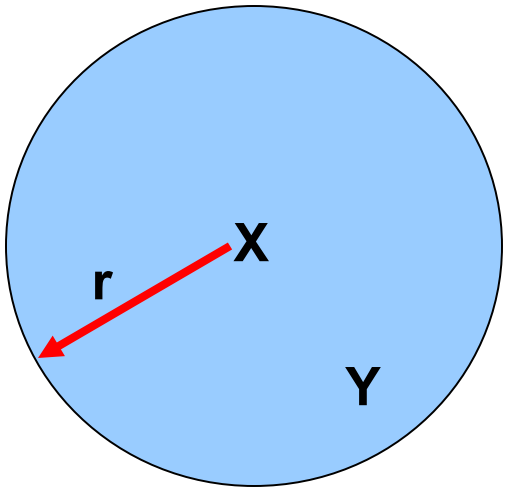
Route requests limited to a *Request Zone* that contains the Expected Zone and location of the sender node

Expected Zone in LAR

X = last known location of node D, at time t0

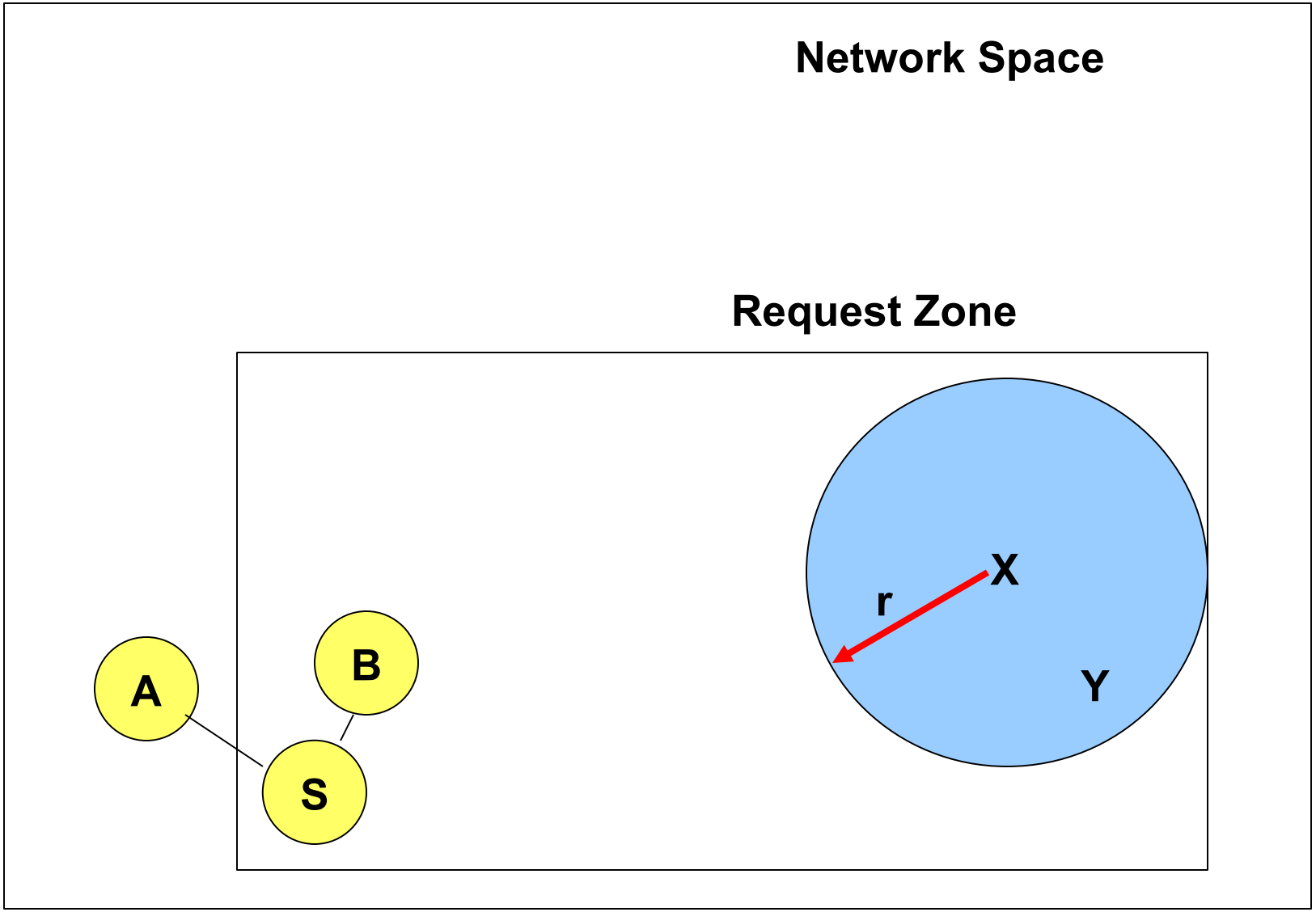
Y = location of node D at current time t1, unknown to node S

r = (t1 - t0) * estimate of D's speed



Expected Zone

Request Zone in LAR



LAR

Only nodes within the request zone forward route requests

- Node A does not forward RREQ, but node B does (see previous slide)

Request zone explicitly specified in the route request

Each node must know its physical location to determine whether it is within the request zone

LAR

Only nodes within the request zone forward route requests

If route discovery using the smaller request zone fails to find a route, the sender initiates another route discovery (after a timeout) using a larger request zone

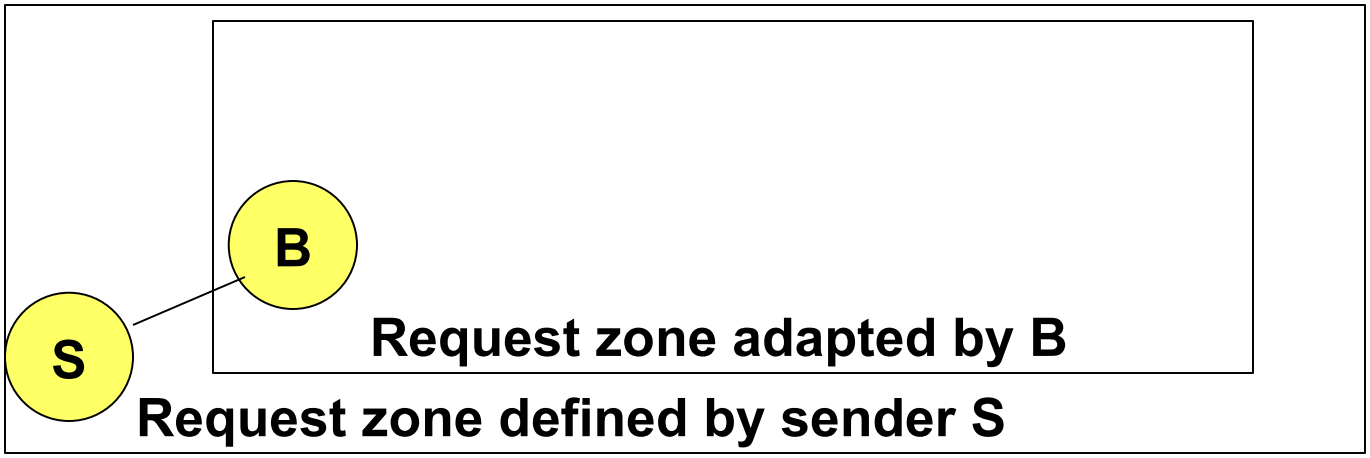
- the larger request zone may be the entire network

Rest of route discovery protocol similar to DSR

LAR Variations: Adaptive Request Zone

Each node may modify the request zone included in the forwarded request

Modified request zone may be determined using more recent/accurate information, and may be smaller than the original request zone



LAR Variations: Implicit Request Zone

In the previous scheme, a route request explicitly specified a request zone

Alternative approach: A node X forwards a route request received from Y if node X is deemed to be closer to the expected zone as compared to Y

The motivation is to attempt to bring the route request physically closer to the destination node after each forwarding

Location-Aided Routing

The basic proposal assumes that, *initially,* location information for node X becomes known to Y only during a route discovery

This location information is used for a future route discovery

Each route discovery yields more updated information which is used for the next discovery

Variations

Location information can also be piggybacked on any message from Y to X

Y may also proactively distribute its location information

Similar to other protocols (e.g., DREAM, GLS)

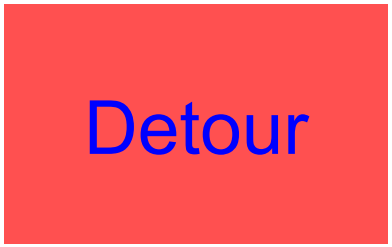
Location Aided Routing (LAR)

Advantages

- reduces the scope of route request flood
- reduces overhead of route discovery

Disadvantages

- Nodes need to know their physical locations
- Does not take into account possible existence of obstructions for radio transmissions



Routing Using Location Information

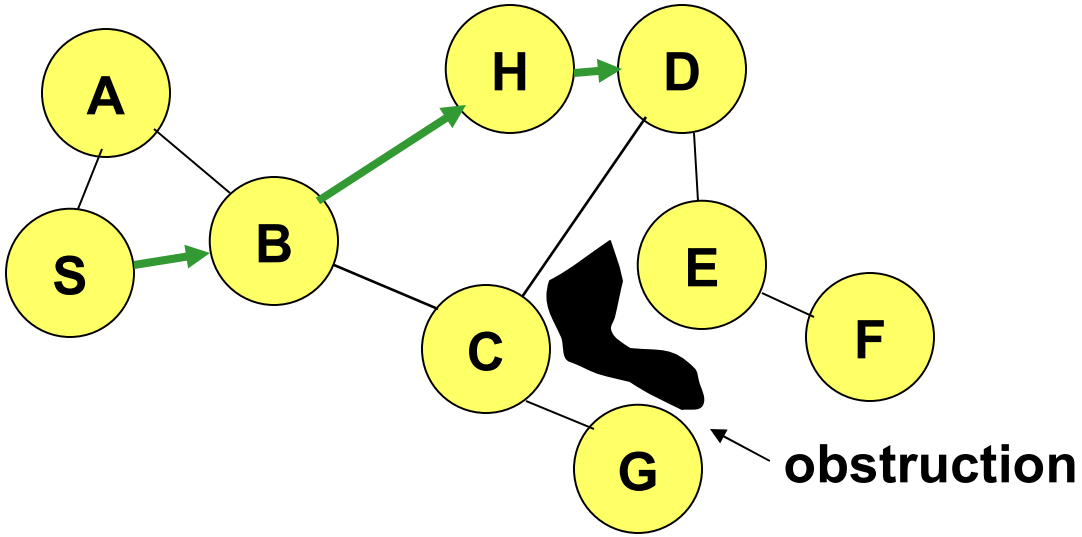
Geographic Distance Routing (GEDIR) [Lin98]

Location of the destination node is assumed known

Each node knows location of its neighbors

Each node forwards a packet to its neighbor closest to the destination

Route taken from S to D shown below

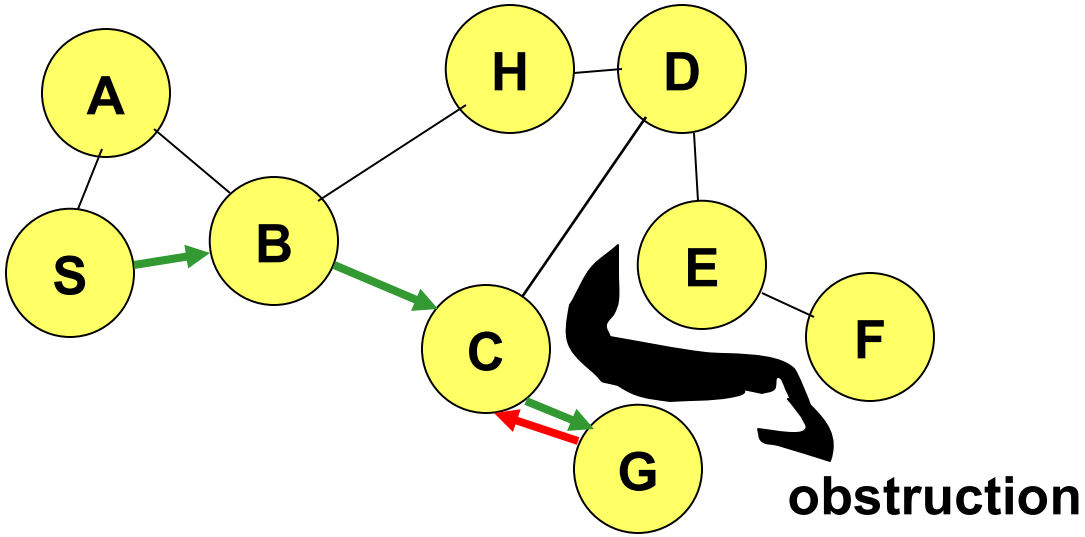


Geographic Distance Routing (GEDIR)
[Stojmenovic99]

The algorithm terminates when same edge traversed twice consecutively

Algorithm fails to route from S to E

Node G is the neighbor of C who is closest from destination E, but C does not have a route to E



Routing with Guaranteed Delivery [Bose99Dialm]

Improves on GEDIR [Lin98]

Guarantees delivery (using location information) provided that a path exists from source to destination

Routes around obstacles if necessary

A similar idea also appears in [Karp00Mobicom]

End of Detour

Back to

Reducing Scope of the Route Request Flood

Query Localization [Castaneda99Mobicom]

Limits route request flood without using physical information

Route requests are propagated only along paths that are *close* to the previously known route

The *closeness* property is defined without using physical location information

Query Localization

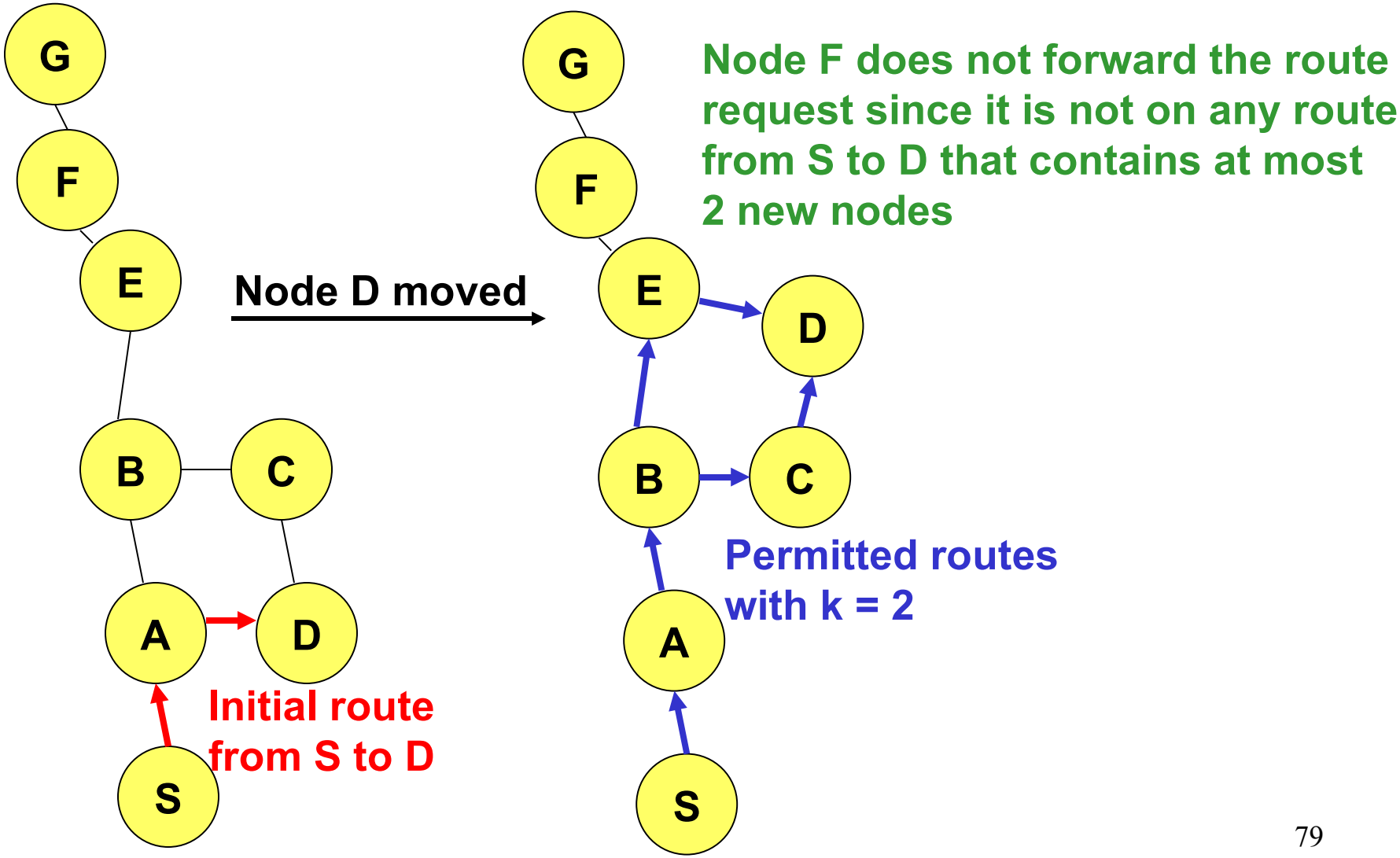
Path locality heuristic: Look for a new path that contains at most *k* nodes that were not present in the previously known route

Old route is piggybacked on a Route Request

Route Request is forwarded only if the accumulated route in the Route Request contains at most *k* new nodes that were absent in the old route

- this limits propagation of the route request

Query Localization: Example



Query Localization

Advantages:

- Reduces overhead of route discovery without using physical location information
- Can perform better in presence of obstructions by searching for new routes in the *vicinity* of old routes

Disadvantage:

- May yield routes longer than LAR
  - (Shortest route may contain more than k new nodes)

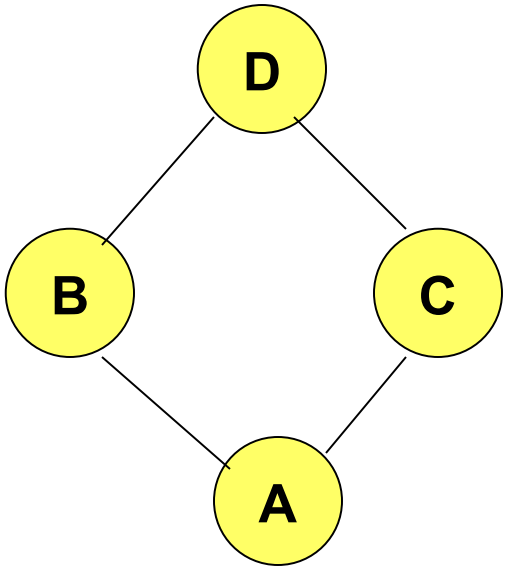
Broadcast Storm Problem [Ni99Mobicom]

When node A broadcasts a route query, nodes B and C both receive it

B and C both forward to their neighbors

B and C transmit at about the same time since they are reacting to receipt of the same message from A

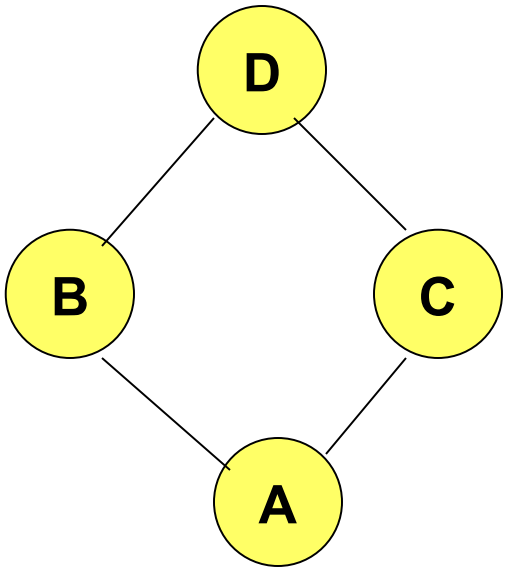
This results in a high probability of collisions



Broadcast Storm Problem

Redundancy: A given node may receive the same route request from too many nodes, when one copy would have sufficed

Node D may receive from nodes B and C both



Solutions for Broadcast Storm

Probabilistic scheme: On receiving a route request for the first time, a node will re-broadcast (forward) the request with probability p

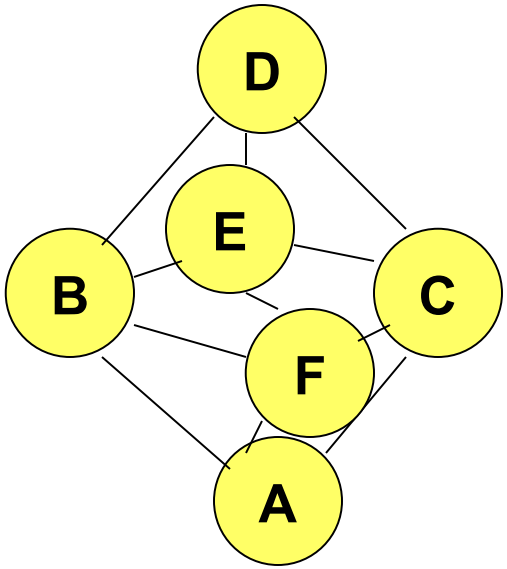
Also, re-broadcasts by different nodes should be staggered by using a collision avoidance technique (wait a random delay when channel is idle)

- this would reduce the probability that nodes B and C would forward a packet simultaneously in the previous example

Solutions for Broadcast Storms

Counter-Based Scheme: If node E hears more than *k* neighbors broadcasting a given route request, before it can itself forward it, then node E will not forward the request

Intuition: *k* neighbors together have probably already forwarded the request to all of E's neighbors

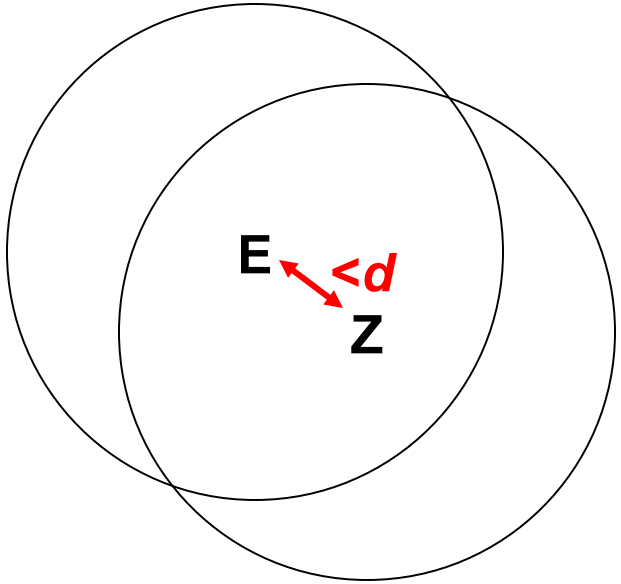


Solutions for Broadcast Storms

Distance-Based Scheme: If node E hears RREQ broadcasted by some node Z within physical distance *d*, then E will not re-broadcast the request

Intuition: Z and E are too close, so transmission areas covered by Z and E are not very different

if E re-broadcasts the request, not many nodes who have not already heard the request from Z will hear the request



Summary: Broadcast Storm Problem

Flooding is used in many protocols, such as Dynamic Source Routing (DSR)

Problems associated with flooding

- collisions
- redundancy

Collisions may be reduced by “jittering” (waiting for a random interval before propagating the flood)

Redundancy may be reduced by selectively re-broadcasting packets from only a subset of the nodes

Ad Hoc On-Demand Distance Vector Routing (AODV) [Perkins99Wmcsa]

DSR includes source routes in packet headers

Resulting large headers can sometimes degrade performance

- particularly when data contents of a packet are small

AODV attempts to improve on DSR by maintaining routing tables at the nodes, so that data packets do not have to contain routes

AODV retains the desirable feature of DSR that routes are maintained only between nodes which need to communicate

AODV

Route Requests (RREQ) are forwarded in a manner similar to DSR

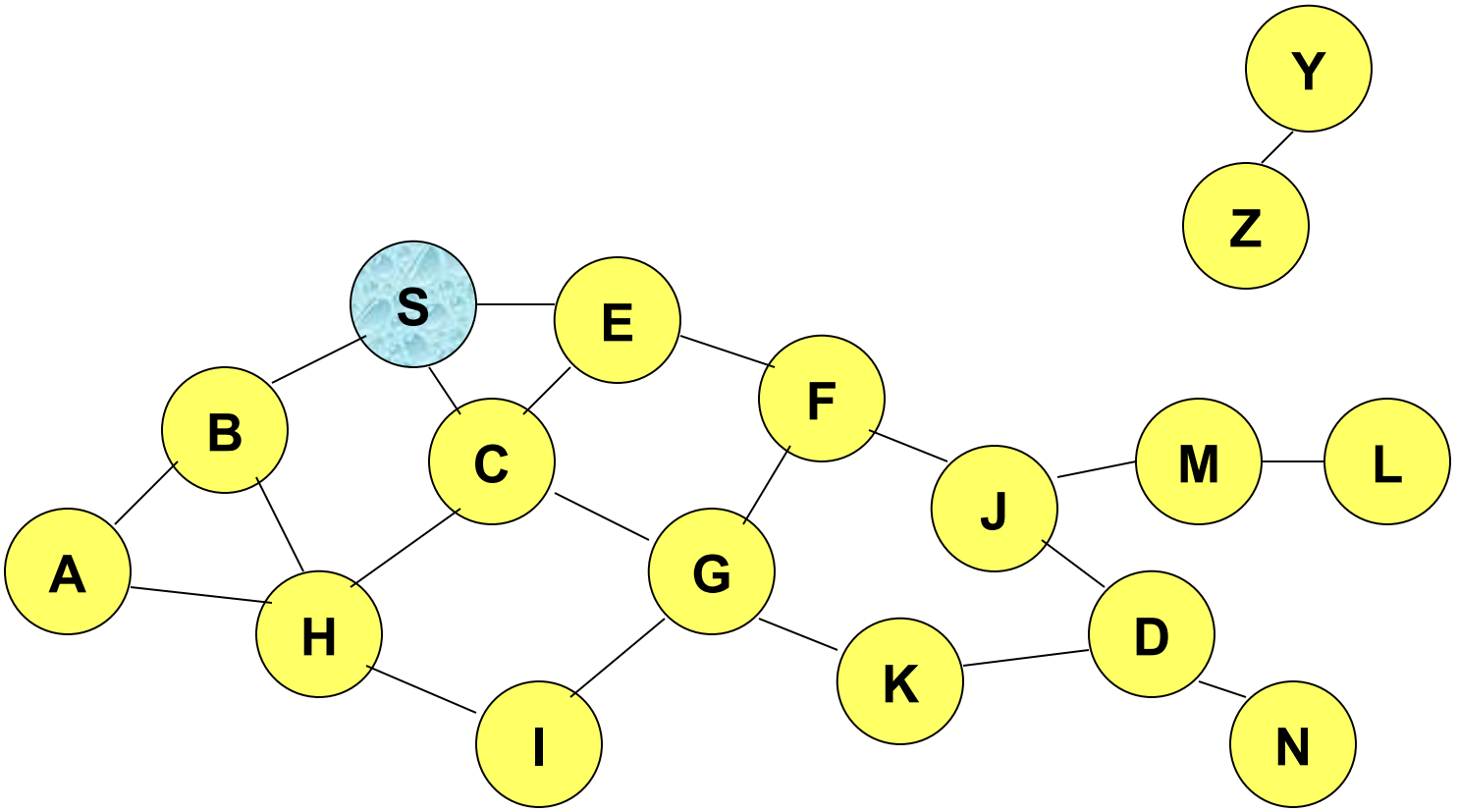
When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source

- AODV assumes symmetric (bi-directional) links

When the intended destination receives a Route Request, it replies by sending a Route Reply

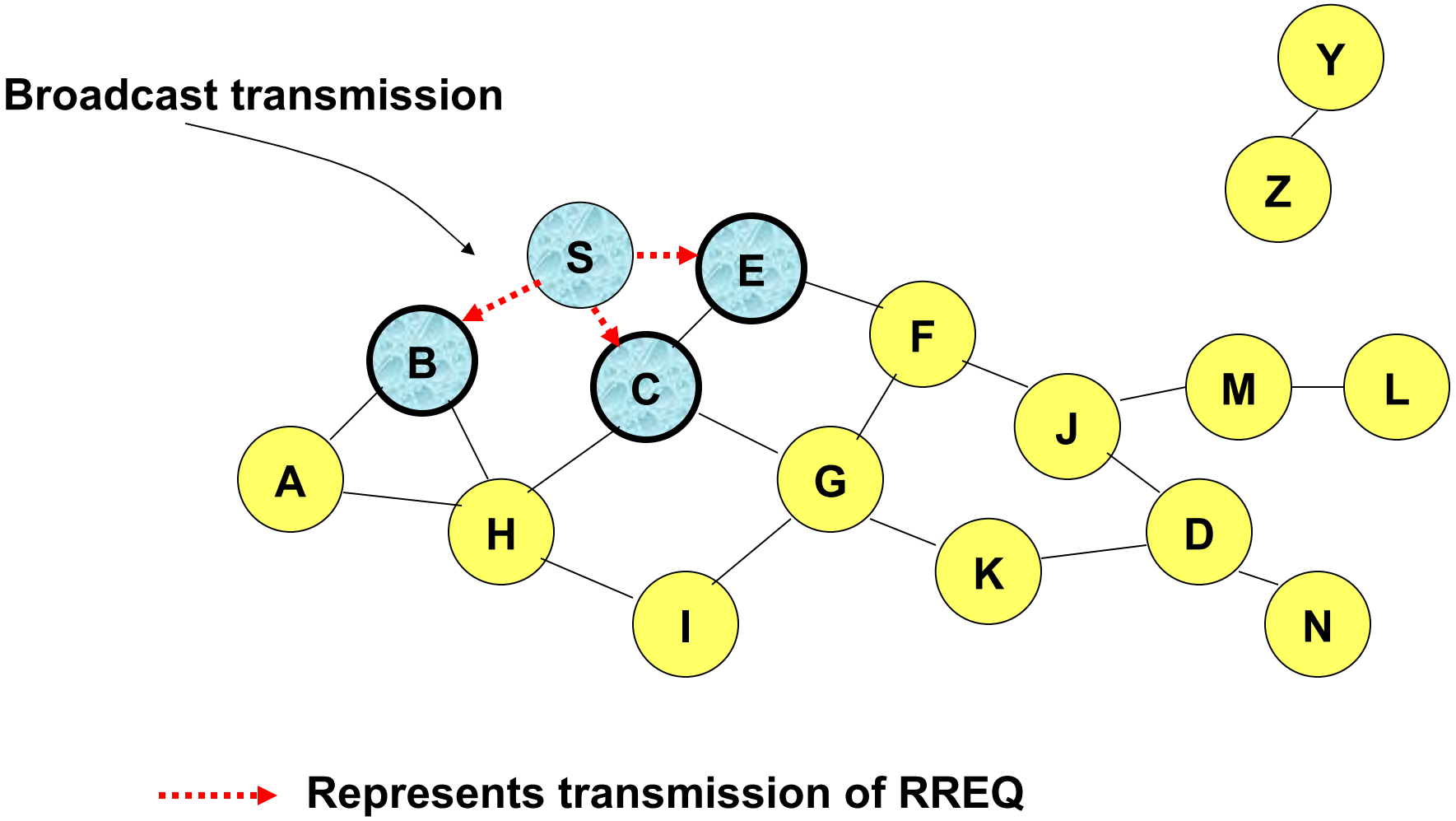
Route Reply travels along the reverse path set-up when Route Request is forwarded

Route Requests in AODV

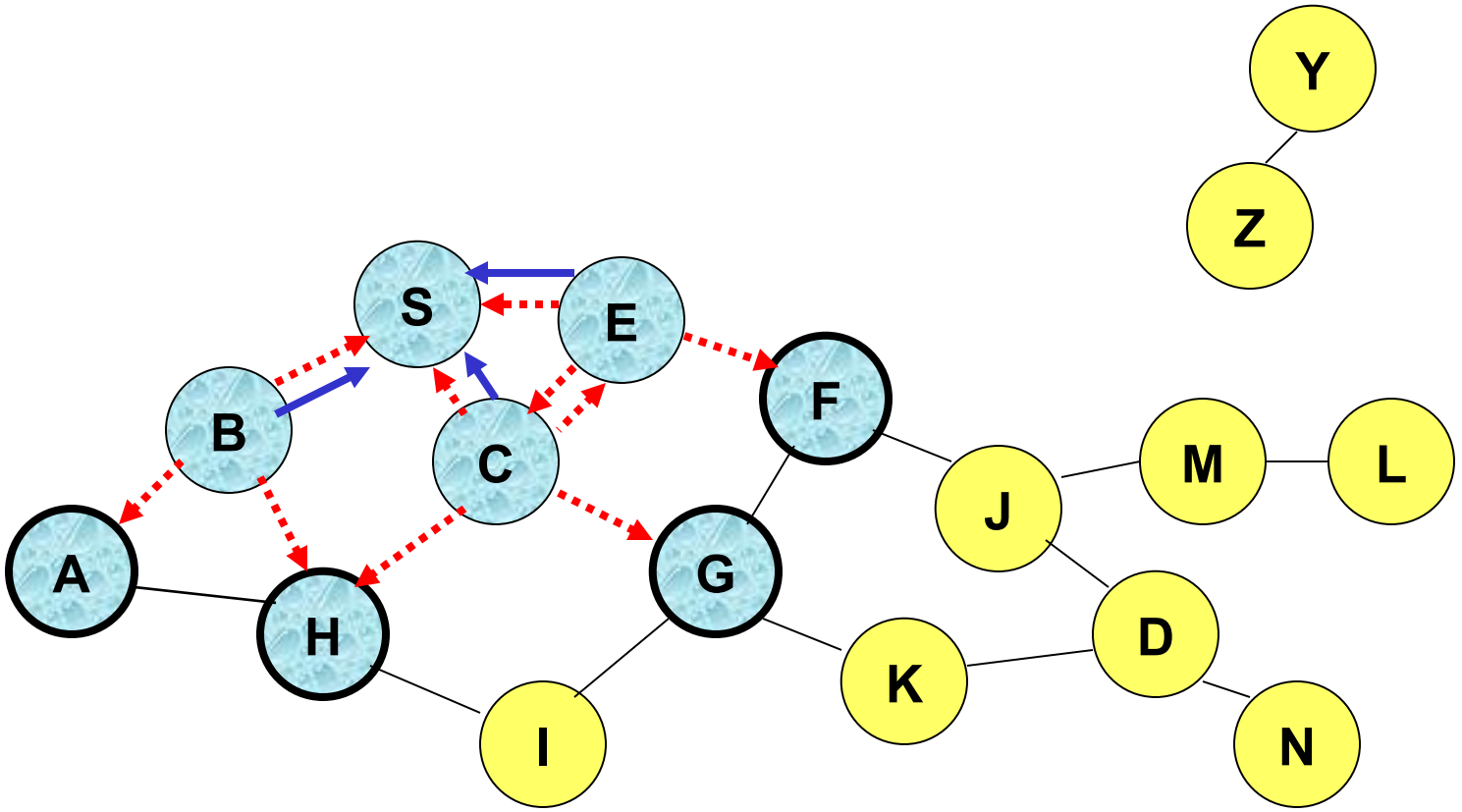


Represents a node that has received RREQ for D from S

Route Requests in AODV

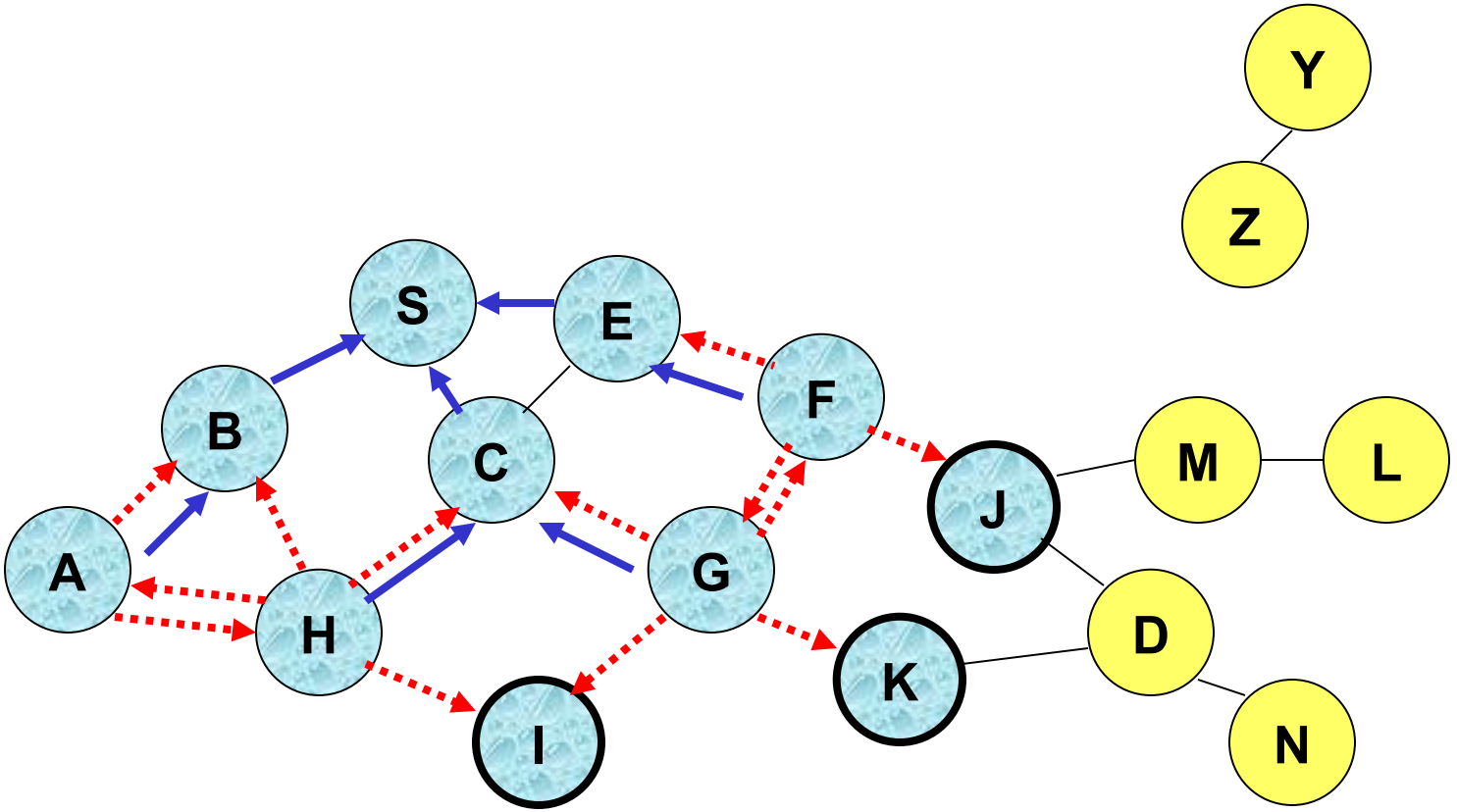


Route Requests in AODV



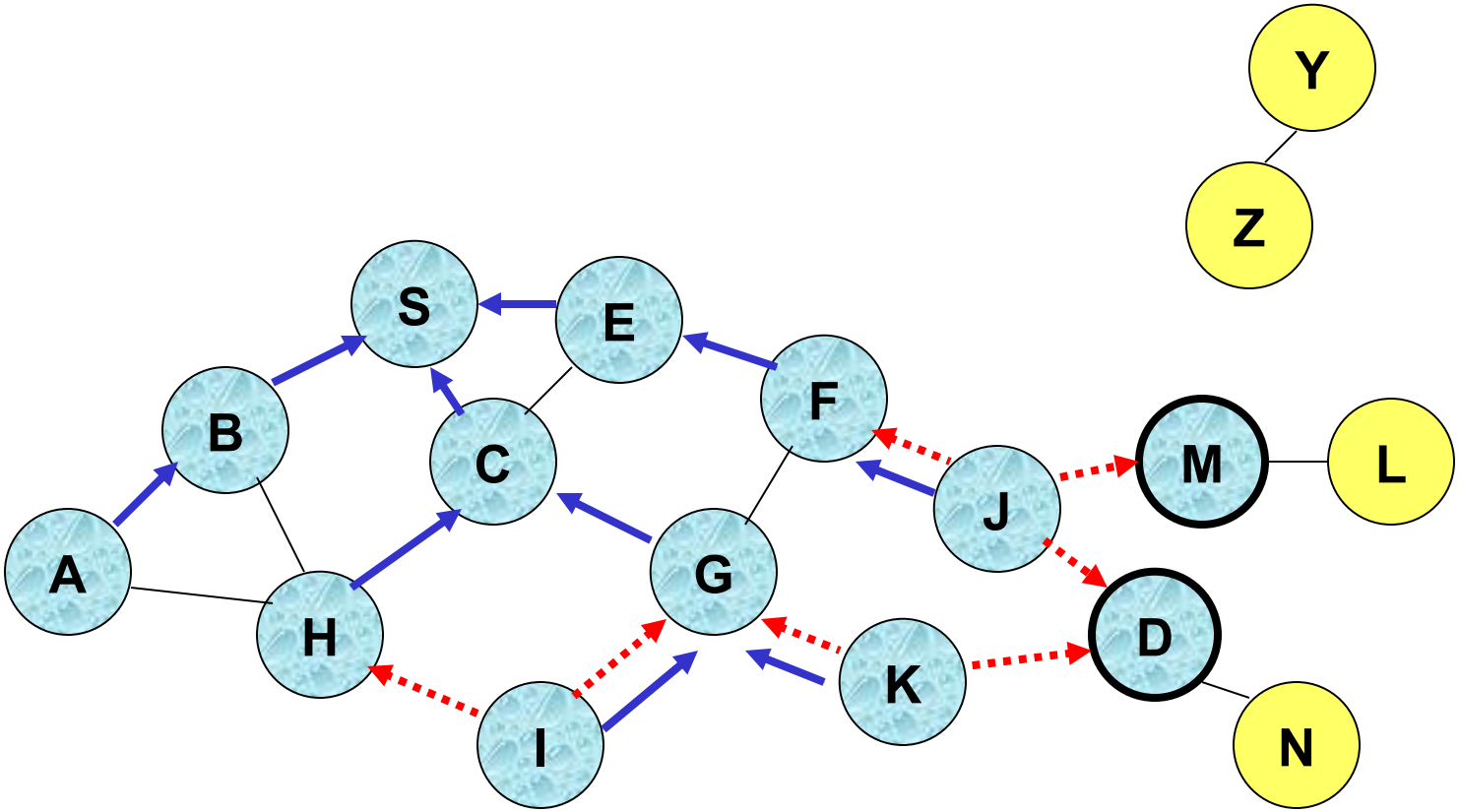
Represents links on Reverse Path

Reverse Path Setup in AODV

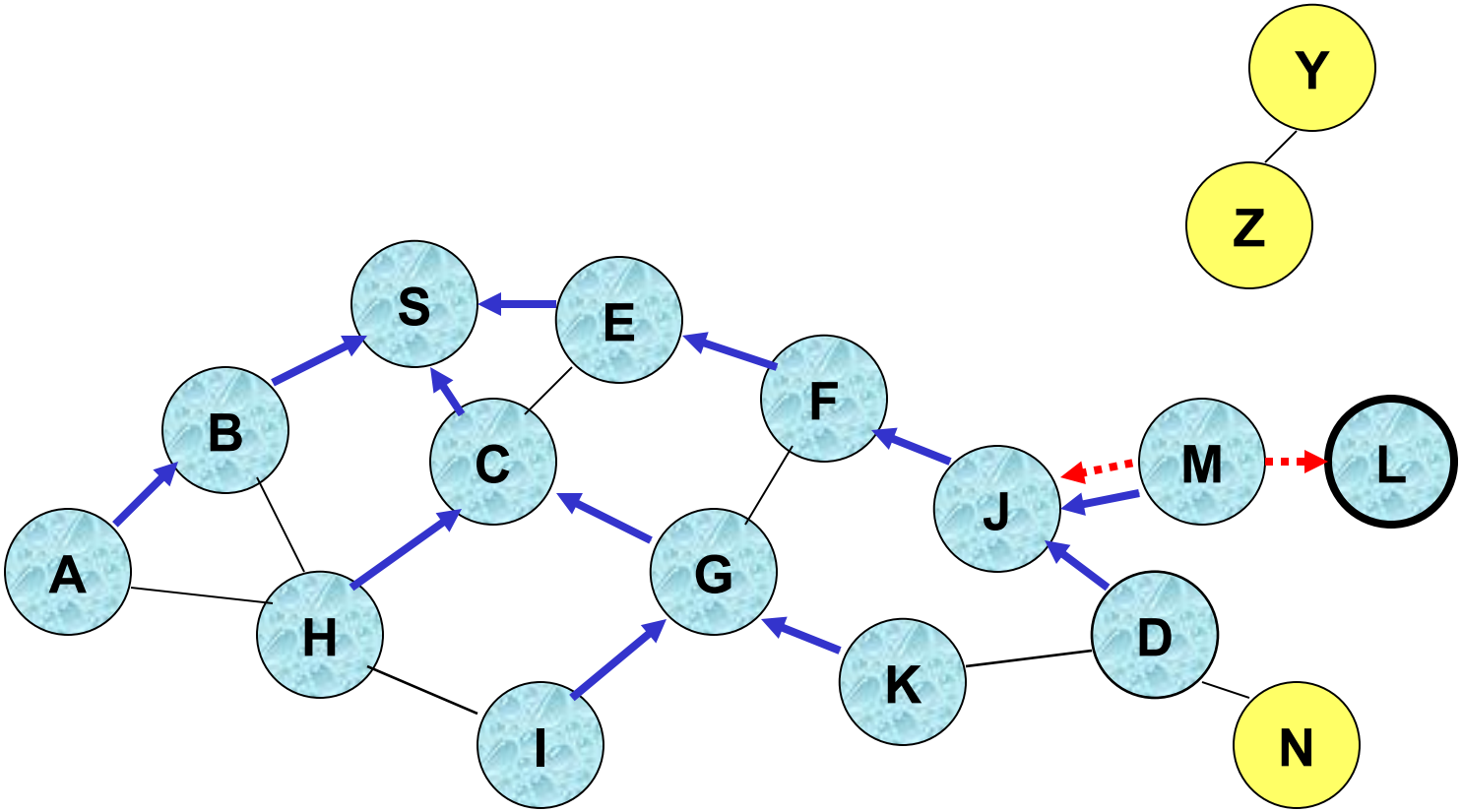


- **Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once**

Reverse Path Setup in AODV

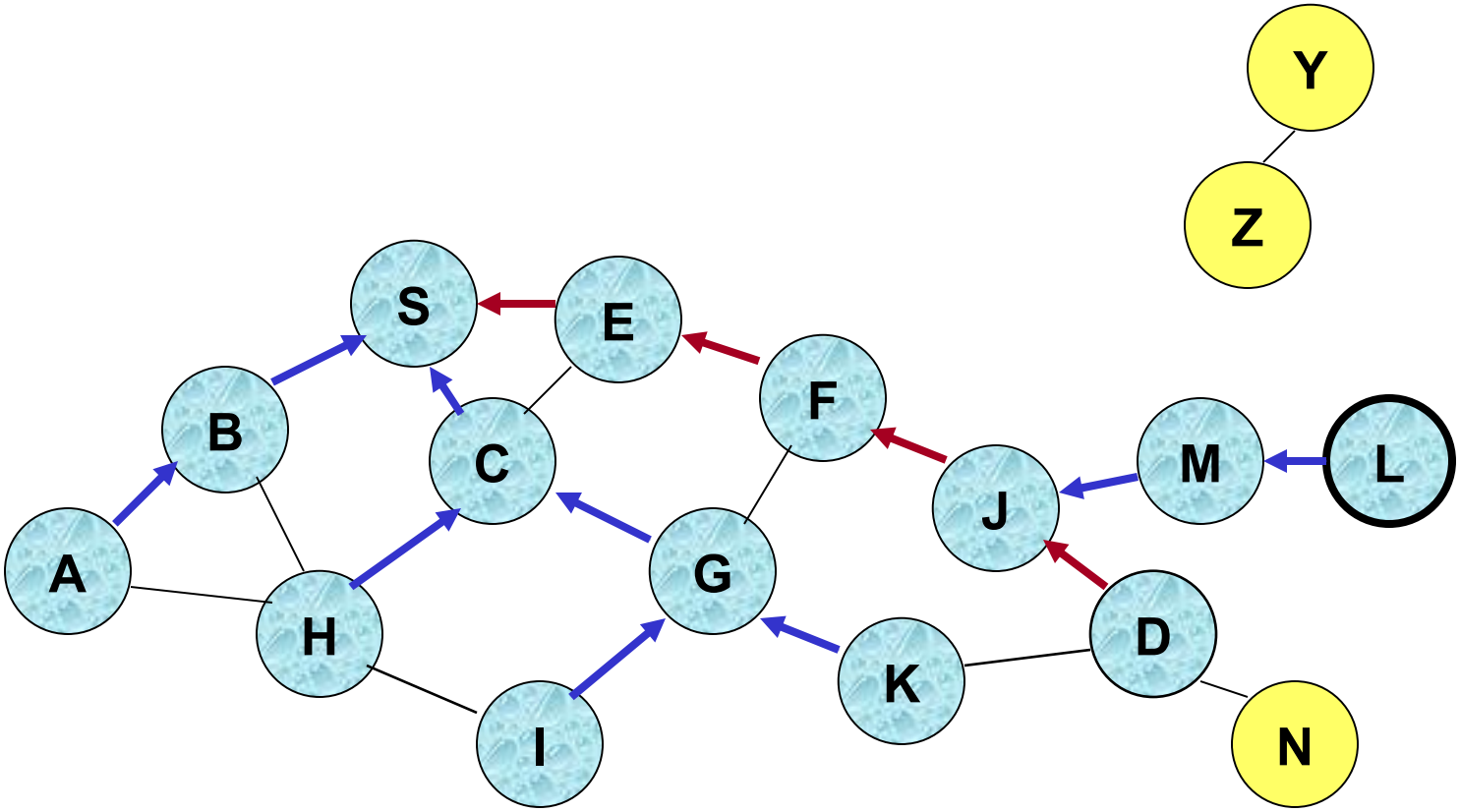


Reverse Path Setup in AODV



- Node D **does not forward** RREQ, because node D is the **intended target** of the RREQ

Route Reply in AODV



Represents links on path taken by RREP

Route Reply in AODV

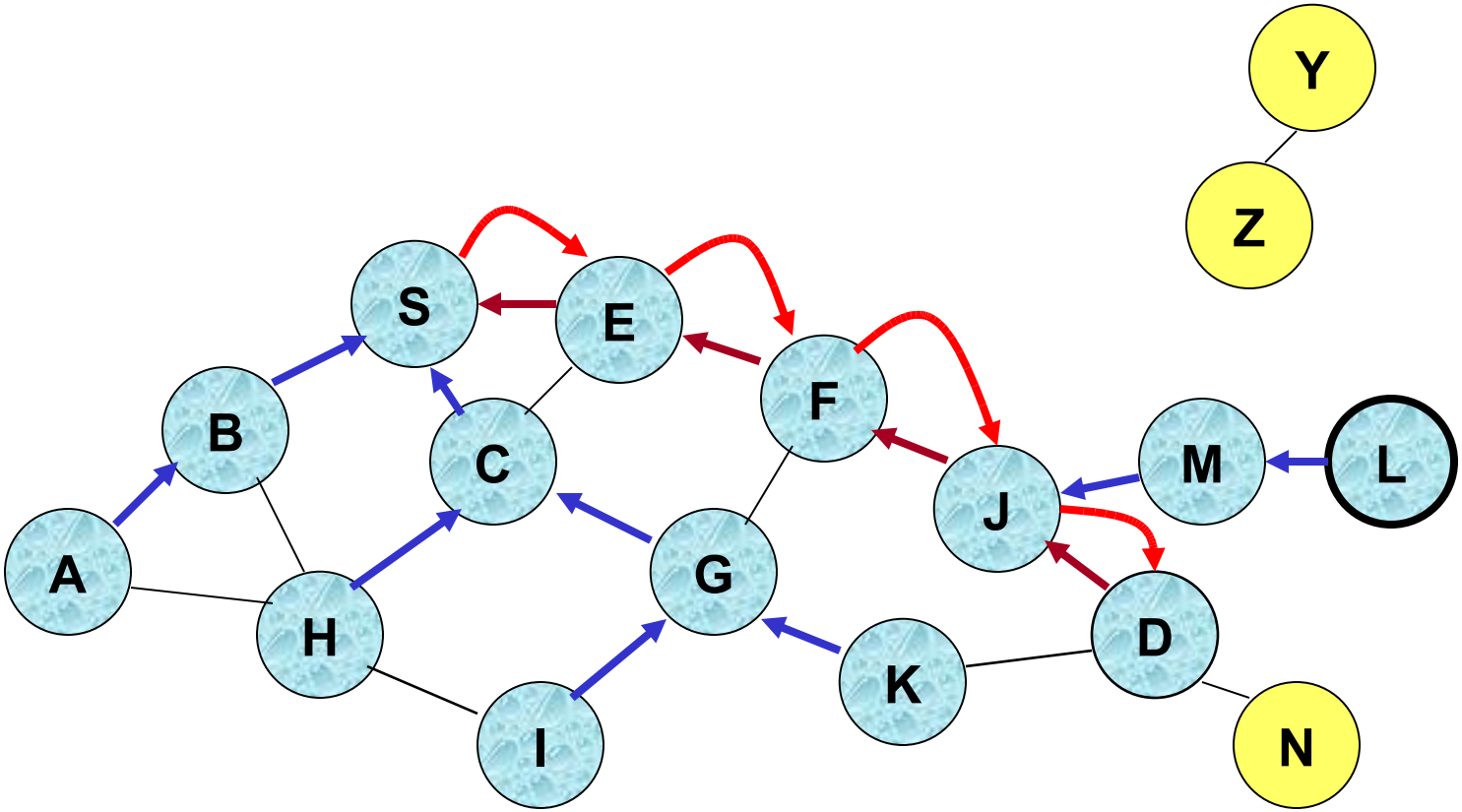
An intermediate node (not the destination) may also send a Route Reply (RREP) provided that it knows a more recent path than the one previously known to sender S

To determine whether the path known to an intermediate node is more recent, *destination sequence numbers* are used

The likelihood that an intermediate node will send a Route Reply when using AODV not as high as DSR

- A new Route Request by node S for a destination is assigned a higher destination sequence number. An intermediate node which knows a route, but with a smaller sequence number, cannot send Route Reply

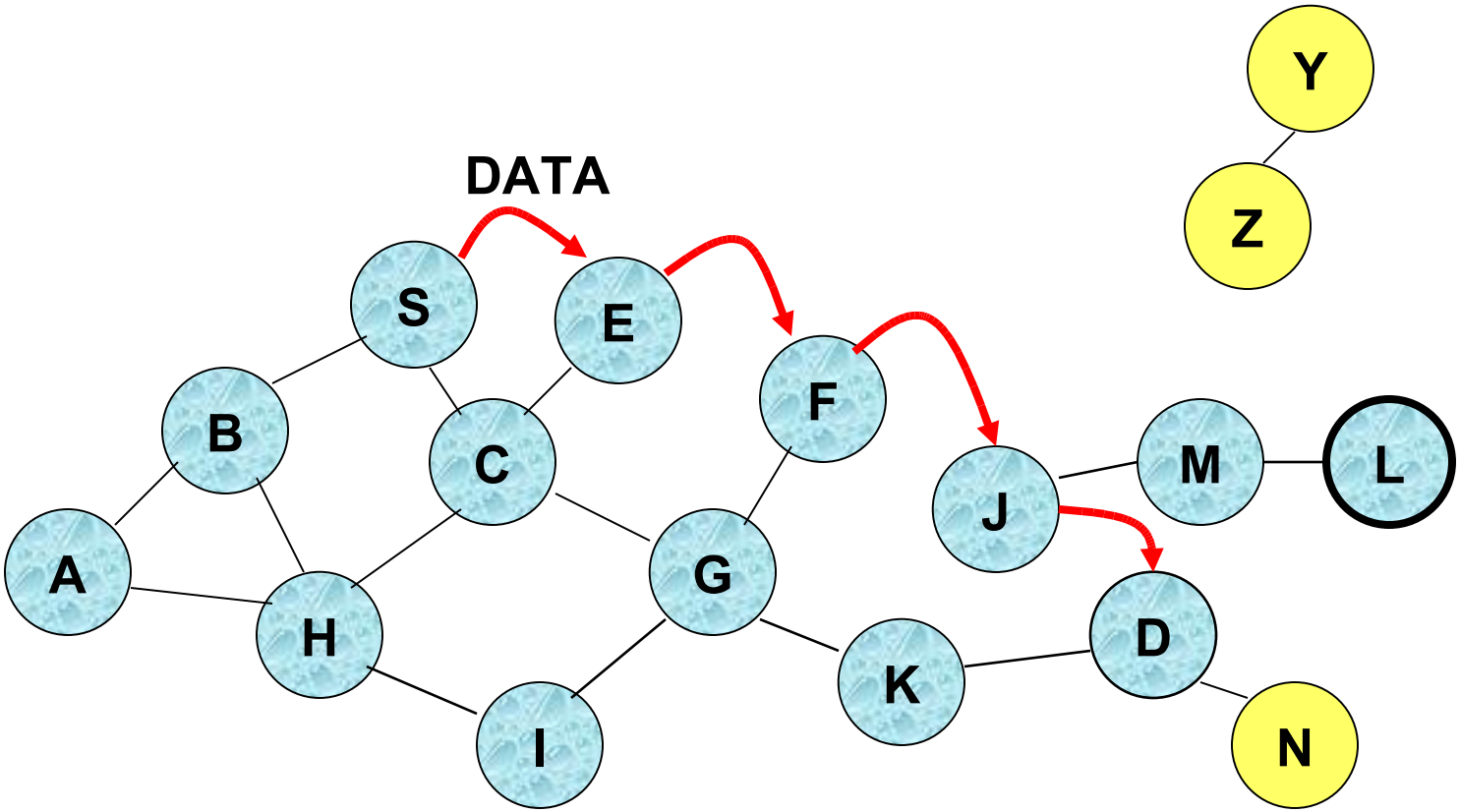
Forward Path Setup in AODV



Forward links are setup when RREP travels along the reverse path

Represents a link on the forward path

Data Delivery in AODV



Routing table entries used to forward data packet.

Route is *not* included in packet header.

Timeouts

A routing table entry maintaining a reverse path is purged after a timeout interval

- timeout should be long enough to allow RREP to come back

A routing table entry maintaining a forward path is purged if *not used* for a *active_route_timeout* interval

- if no data is being sent using a particular routing table entry, that entry will be deleted from the routing table (even if the route may actually still be valid)

Link Failure Reporting

A neighbor of node X is considered active for a routing table entry if the neighbor sent a packet within *active_route_timeout* interval which was forwarded using that entry

When the next hop link in a routing table entry breaks, all active neighbors are informed

Link failures are propagated by means of Route Error messages, which also update destination sequence numbers

Route Error

When node X is unable to forward packet P (from node S to node D) on link (X,Y), it generates a RERR message

Node X increments the destination sequence number for D cached at node X

The incremented sequence number *N* is included in the RERR

When node S receives the RERR, it initiates a new route discovery for D using destination sequence number at least as large as *N*

Destination Sequence Number

Continuing from the previous slide …

When node D receives the route request with destination sequence number N, node D will set its sequence number to N, unless it is already larger than N

Link Failure Detection

Hello messages: Neighboring nodes periodically exchange hello message

Absence of hello message is used as an indication of link failure

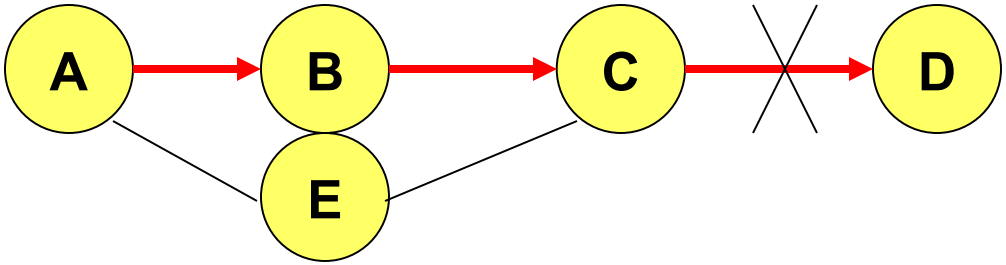
Alternatively, failure to receive several MAC-level acknowledgement may be used as an indication of link failure

Why Sequence Numbers in AODV

To avoid using old/broken routes

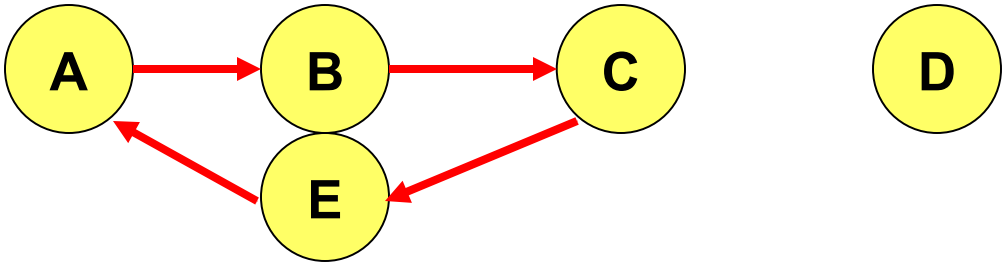
- To determine which route is newer

To prevent formation of loops



- Assume that A does not know about failure of link C-D because RERR sent by C is lost
- Now C performs a route discovery for D. Node A receives the RREQ (say, via path C-E-A)
- Node A will reply since A knows a route to D via node B
- Results in a loop (for instance, C-E-A-B-C)

Why Sequence Numbers in AODV



Loop C-E-A-B-C

Optimization: Expanding Ring Search

Route Requests are initially sent with small Time-to-Live (TTL) field, to limit their propagation

- DSR also includes a similar optimization

If no Route Reply is received, then larger TTL tried

Summary: AODV

Routes need not be included in packet headers

Nodes maintain routing tables containing entries only for routes that are in active use

At most one next-hop per destination maintained at each node

- Multi-path extensions can be designed
- DSR may maintain several routes for a single destination

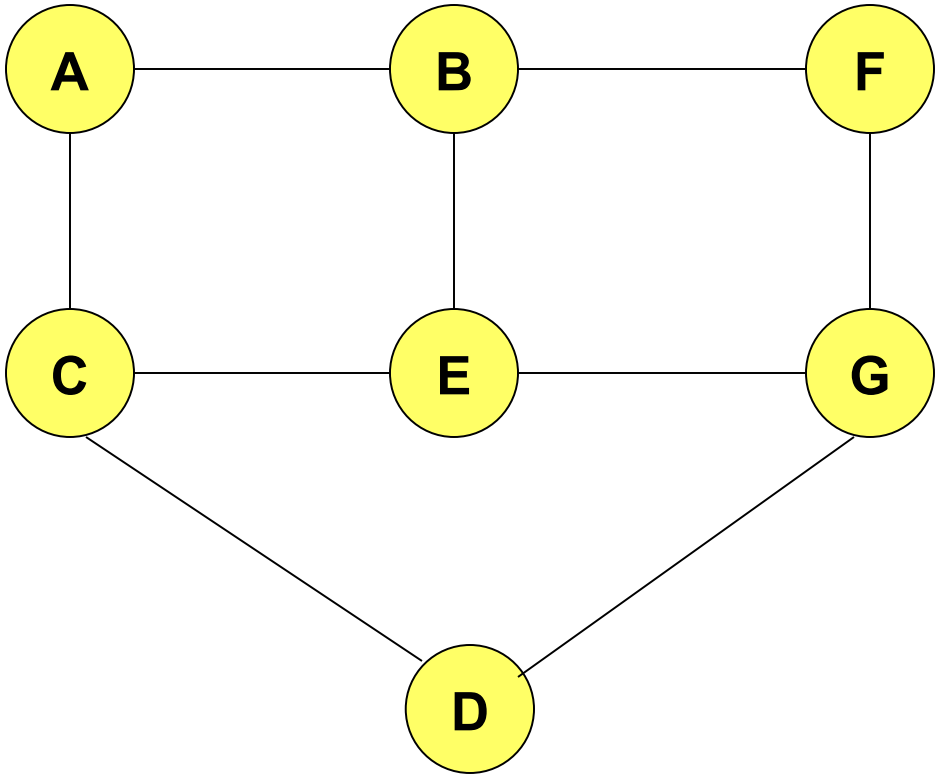
Unused routes expire even if topology does not change

So far ...

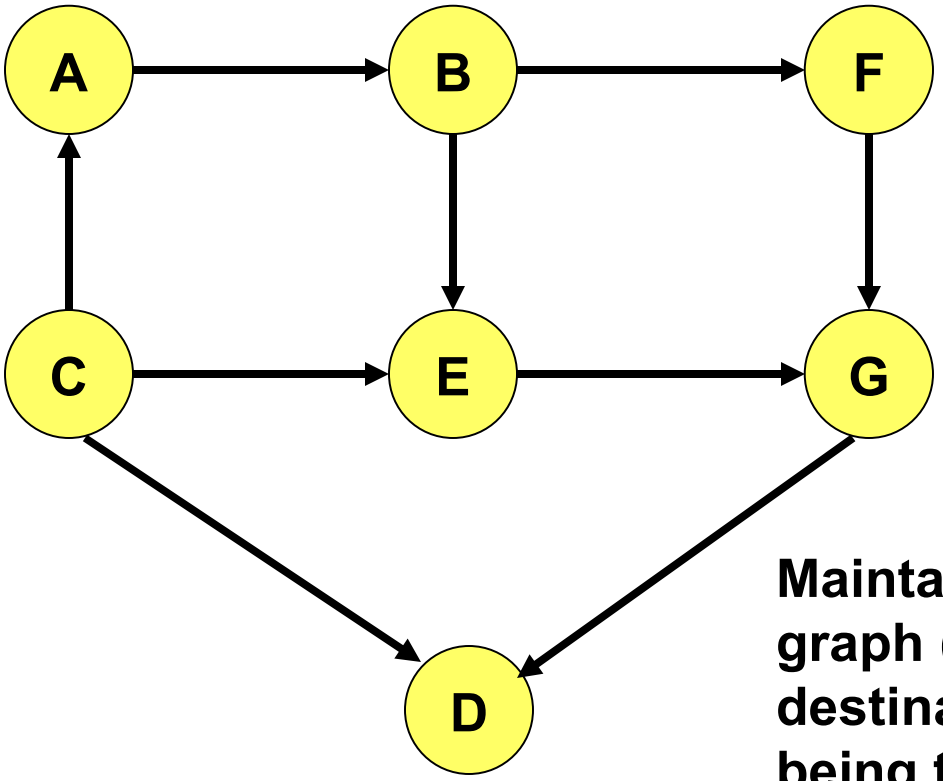
All protocols discussed so far perform some form of flooding

Now we will consider protocols which try to reduce/avoid such behavior

Link Reversal Algorithm [Gafni81]



Link Reversal Algorithm



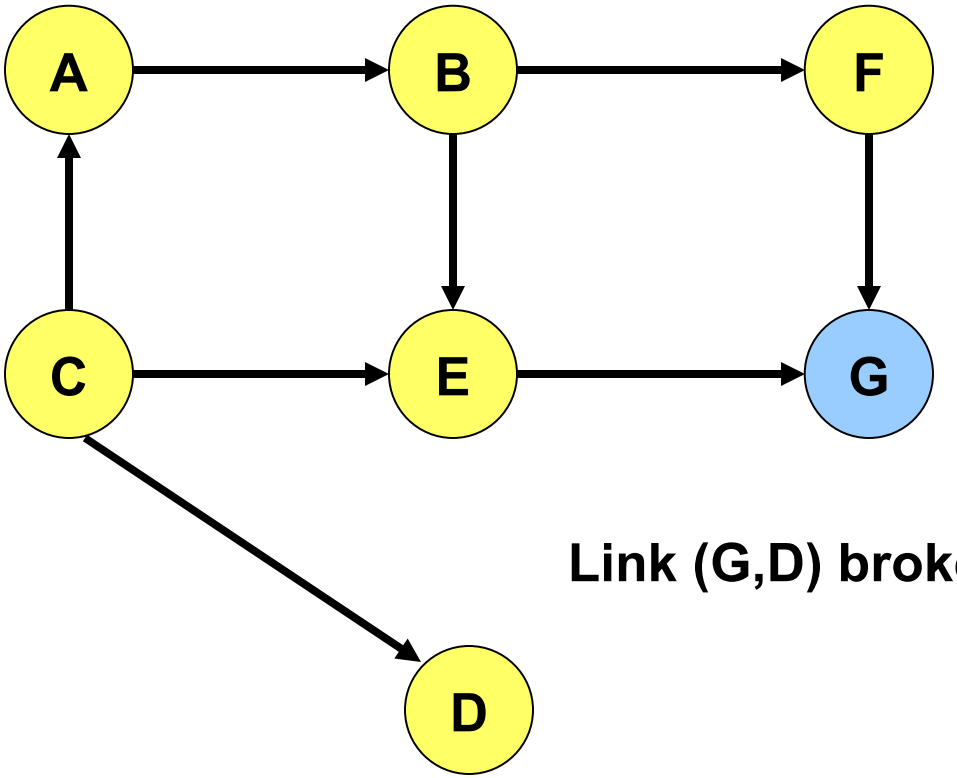
Links are bi-directional

But algorithm imposes logical directions on them

Maintain a directed acyclic graph (DAG) for each destination, with the destination being the *only sink*

This DAG is for *destination node D*

Link Reversal Algorithm

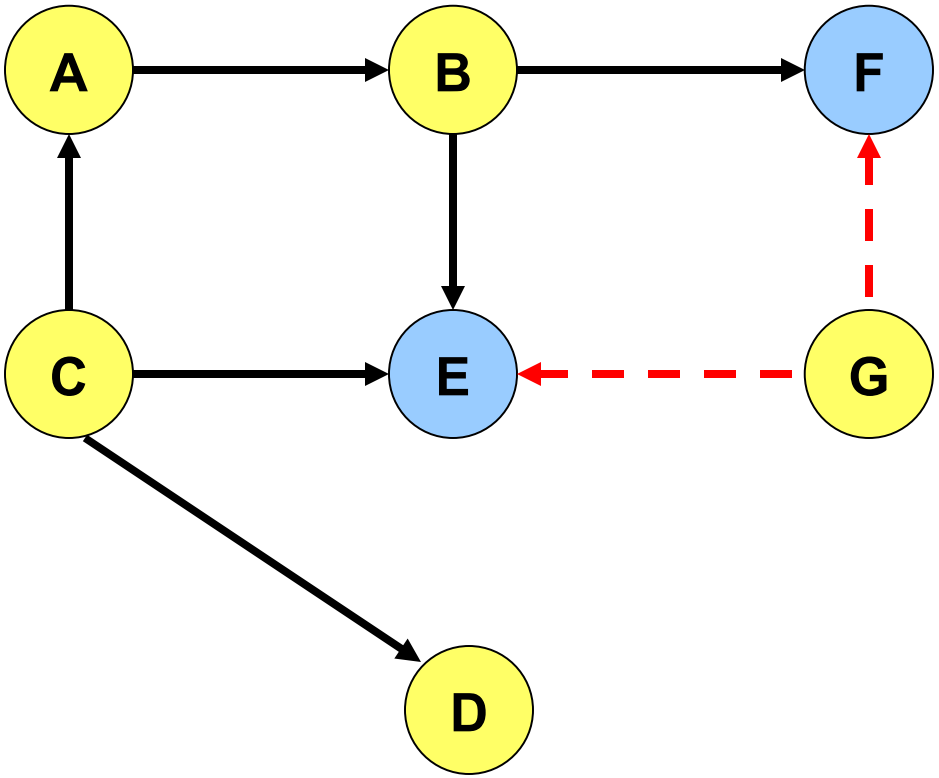


Link (G,D) broke

Any node, other than the destination, that has no outgoing links reverses all its incoming links.

Node G has no outgoing links

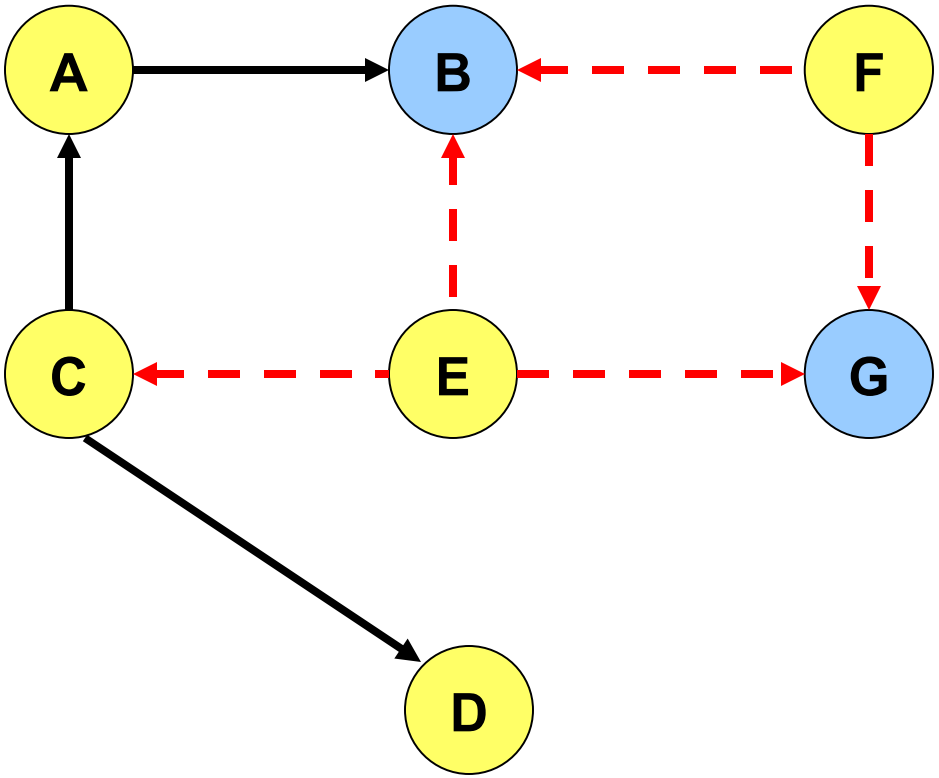
Link Reversal Algorithm



Represents a link that was reversed recently

Now nodes E and F have no outgoing links

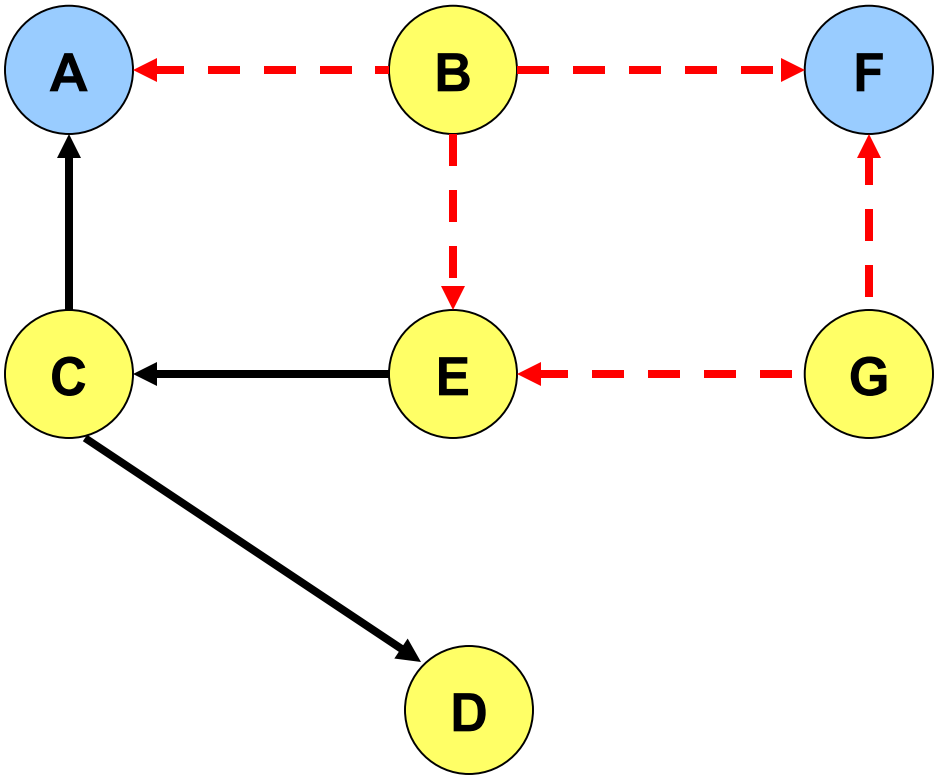
Link Reversal Algorithm



Represents a link that was reversed recently

Now nodes B and G have no outgoing links

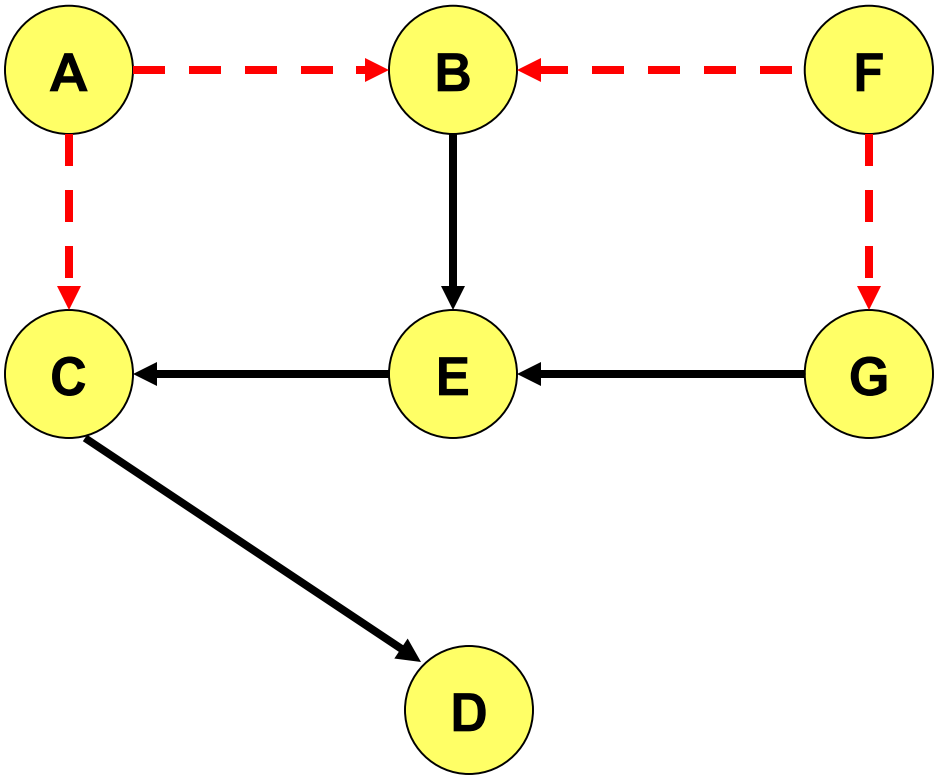
Link Reversal Algorithm



Represents a link that was reversed recently

Now nodes A and F have no outgoing links

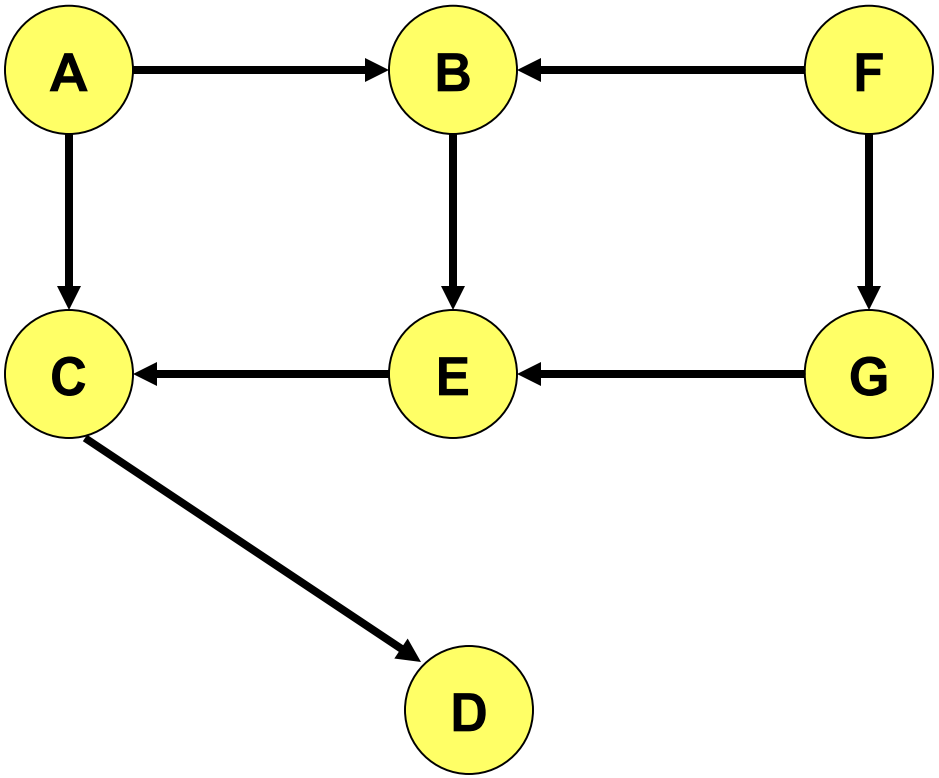
Link Reversal Algorithm



Represents a link that was reversed recently

Now all nodes (other than destination D) have an outgoing link

Link Reversal Algorithm



DAG has been restored with only the destination as a sink

Link Reversal Algorithm

Attempts to keep link reversals local to where the failure occurred

- But this is not guaranteed

When the first packet is sent to a destination, the destination oriented DAG is constructed

The initial construction does result in flooding of control packets

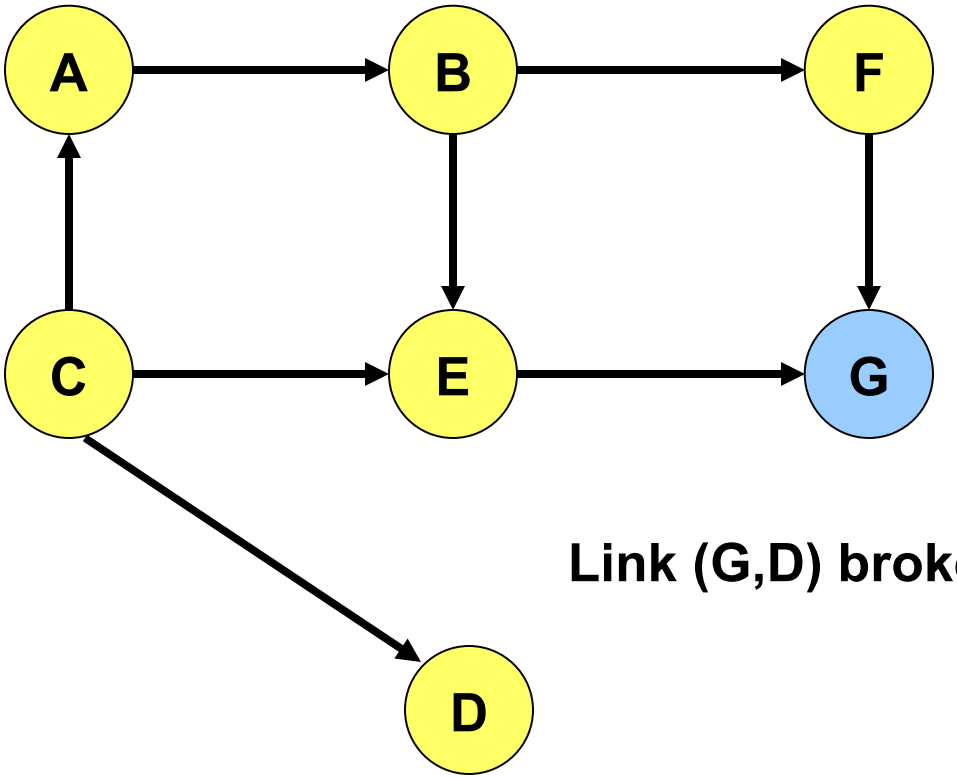
Link Reversal Algorithm

The previous algorithm is called a full reversal method since when a node reverses links, it reverses *all* its incoming links

Partial reversal method [Gafni81]: A node reverses incoming links from only those neighbors who have not themselves reversed links "previously"

- If all neighbors have reversed links, then the node reverses all its incoming links
- "Previously" at node X means *since the last link reversal done by node X*

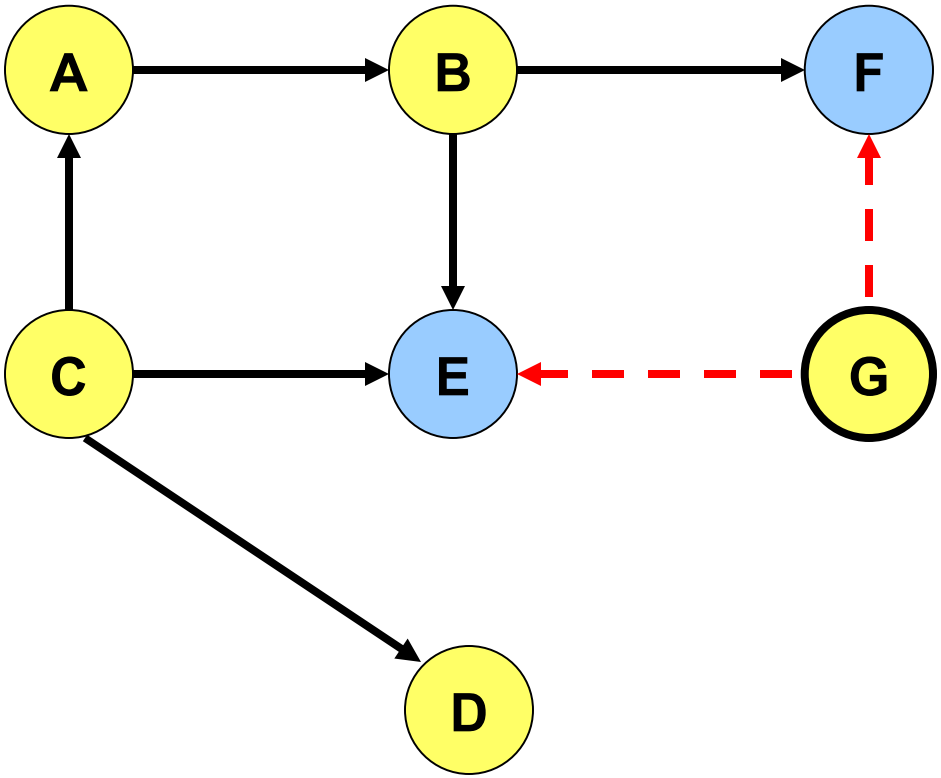
Partial Reversal Method



Link (G,D) broke

Node G has no outgoing links

Partial Reversal Method

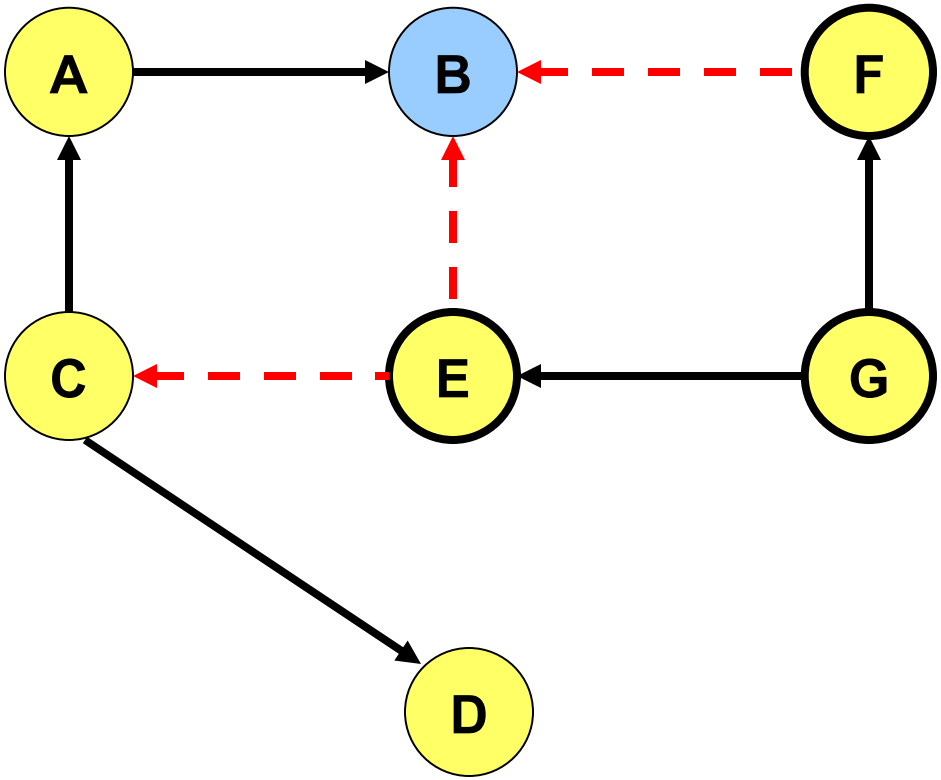


Represents a link that was reversed recently

Represents a node that has reversed links

Now nodes E and F have no outgoing links

Partial Reversal Method

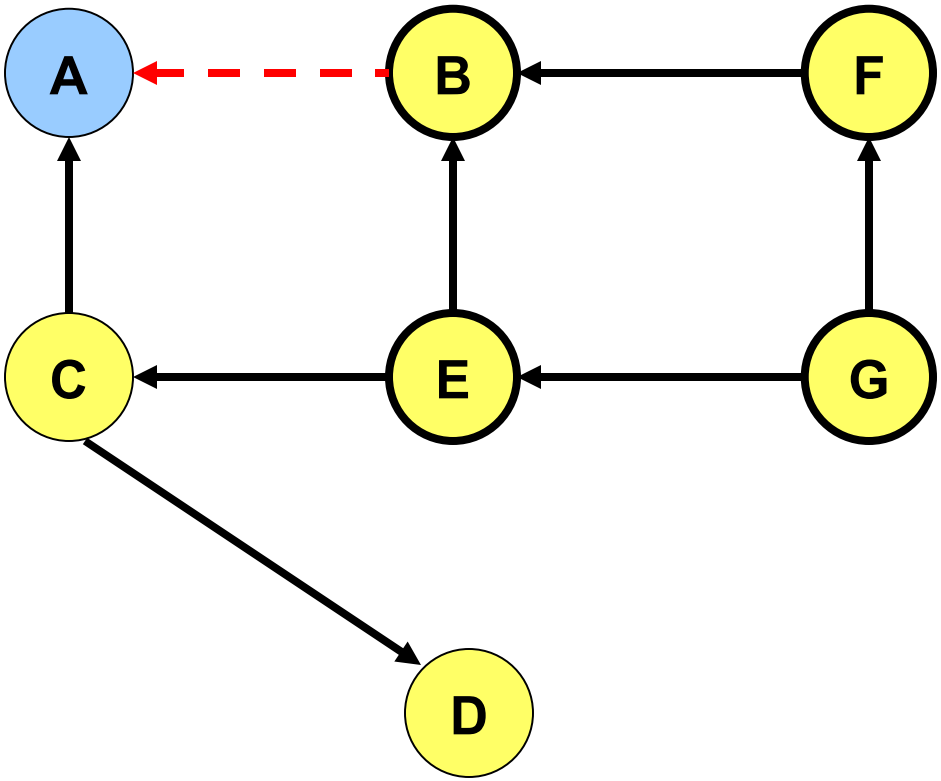


Represents a link that was reversed recently

Nodes E and F *do not* reverse links from node G

Now node B has no outgoing links

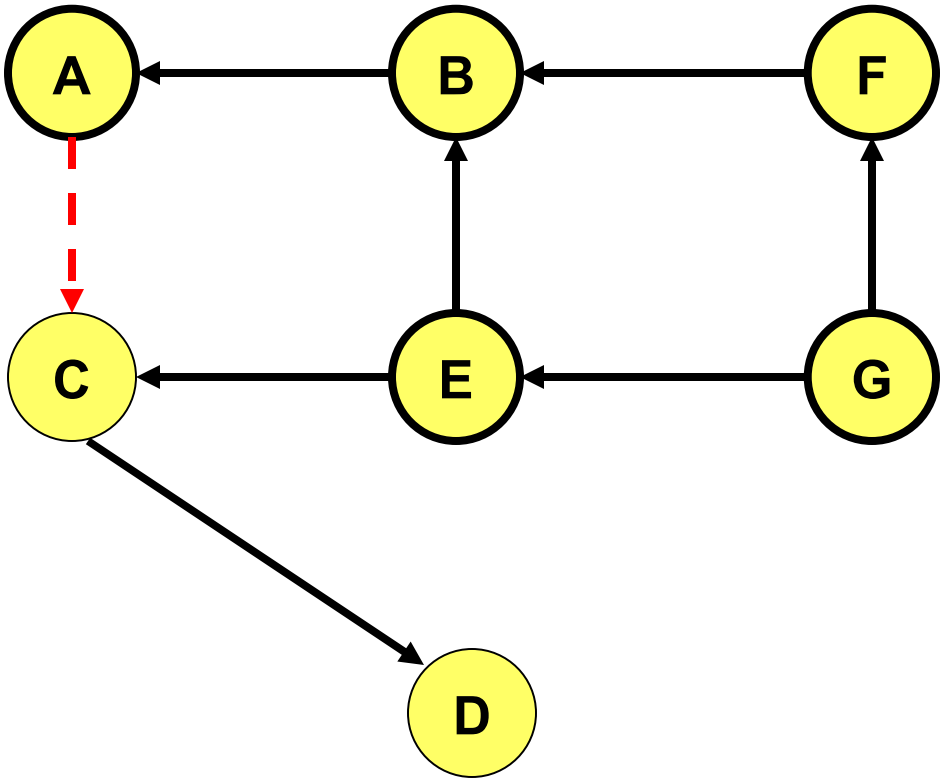
Partial Reversal Method



Represents a link that was reversed recently

Now node A has no outgoing links

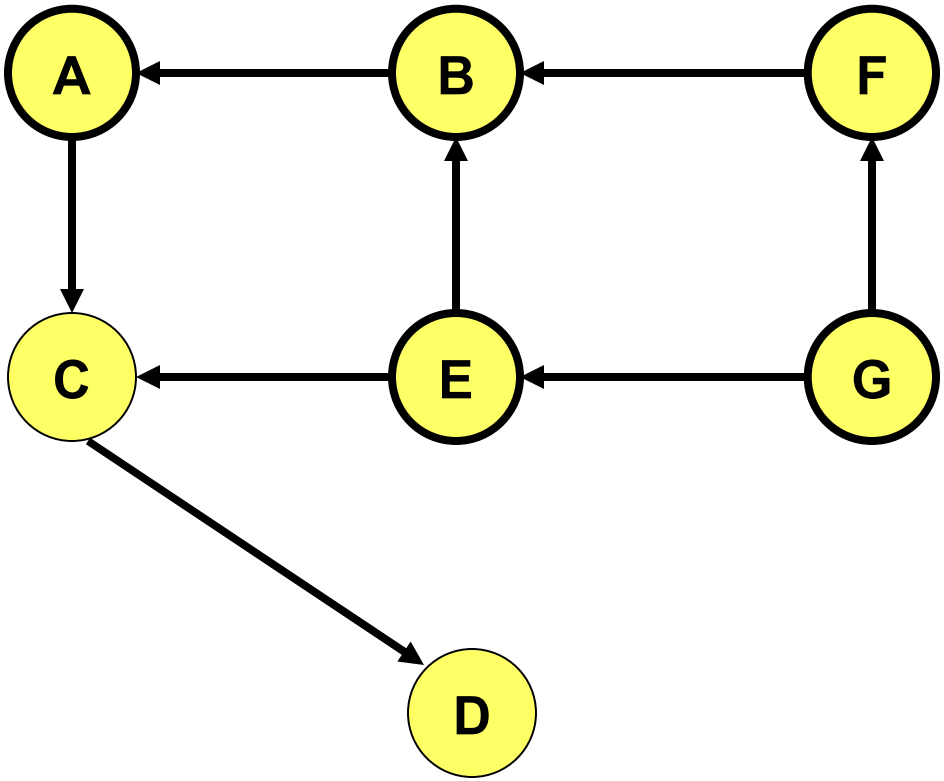
Partial Reversal Method



Represents a link that was reversed recently

Now all nodes (except destination D) have outgoing links

Partial Reversal Method



DAG has been restored with only the destination as a sink

Link Reversal Methods: Advantages

Link reversal methods attempt to limit updates to routing tables at nodes in the vicinity of a broken link

Each node may potentially have multiple routes to a destination

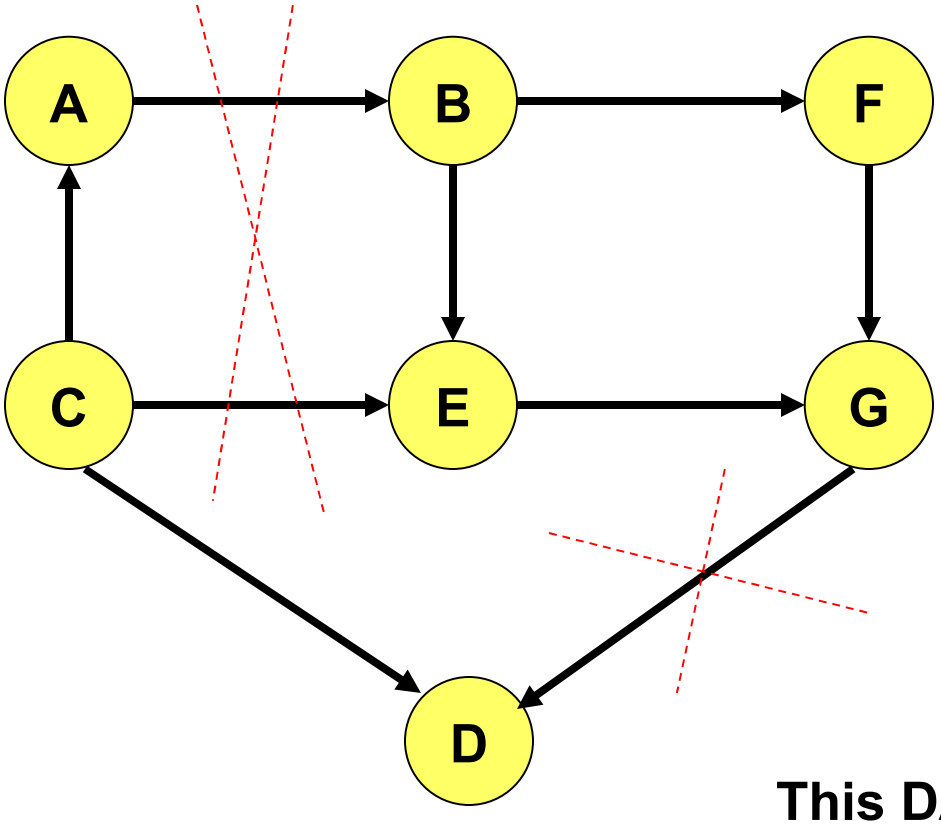
Link Reversal Methods: Disadvantage

Need a mechanism to detect link failure

- hello messages may be used
- but hello messages can add to contention

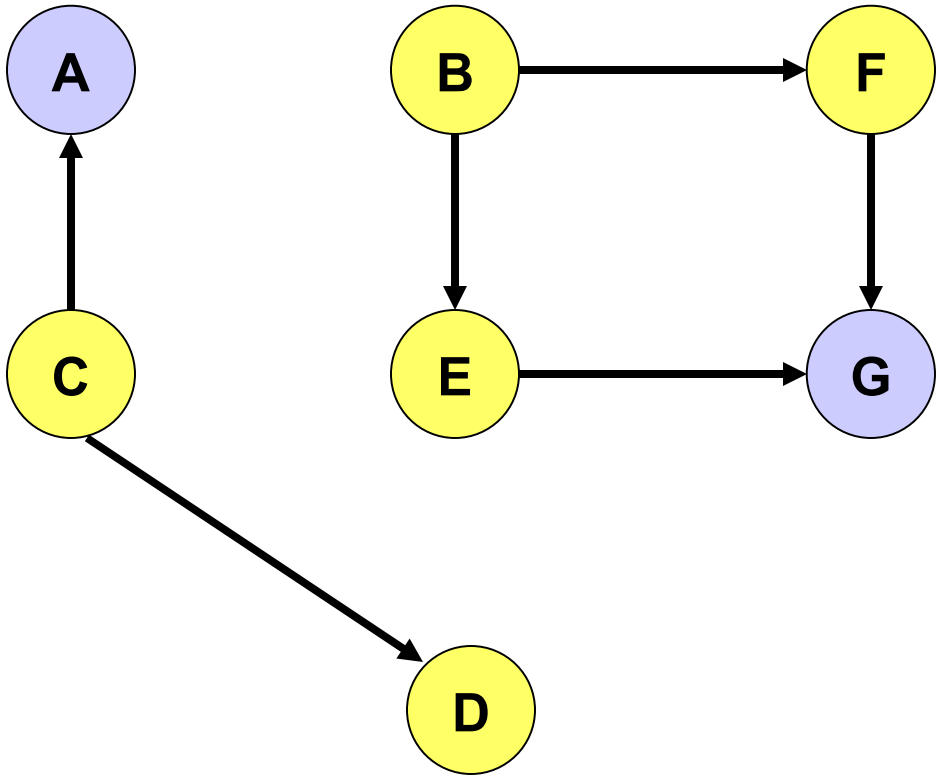
If network is partitioned, link reversals continue indefinitely

Link Reversal in a Partitioned Network



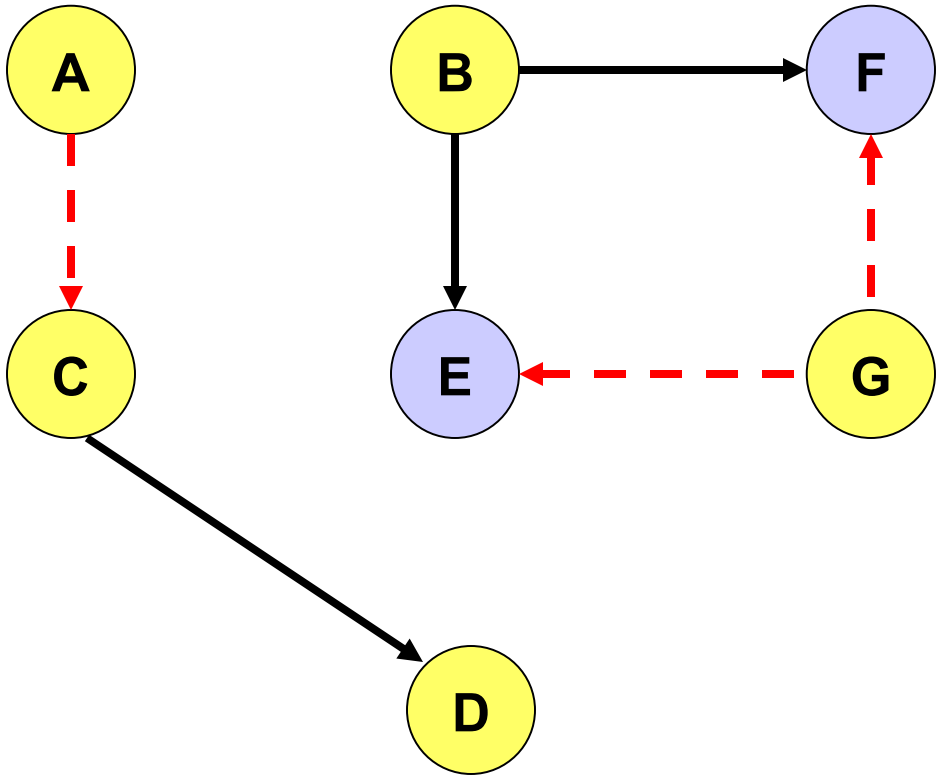
This DAG is for *destination node D*

Full Reversal in a Partitioned Network



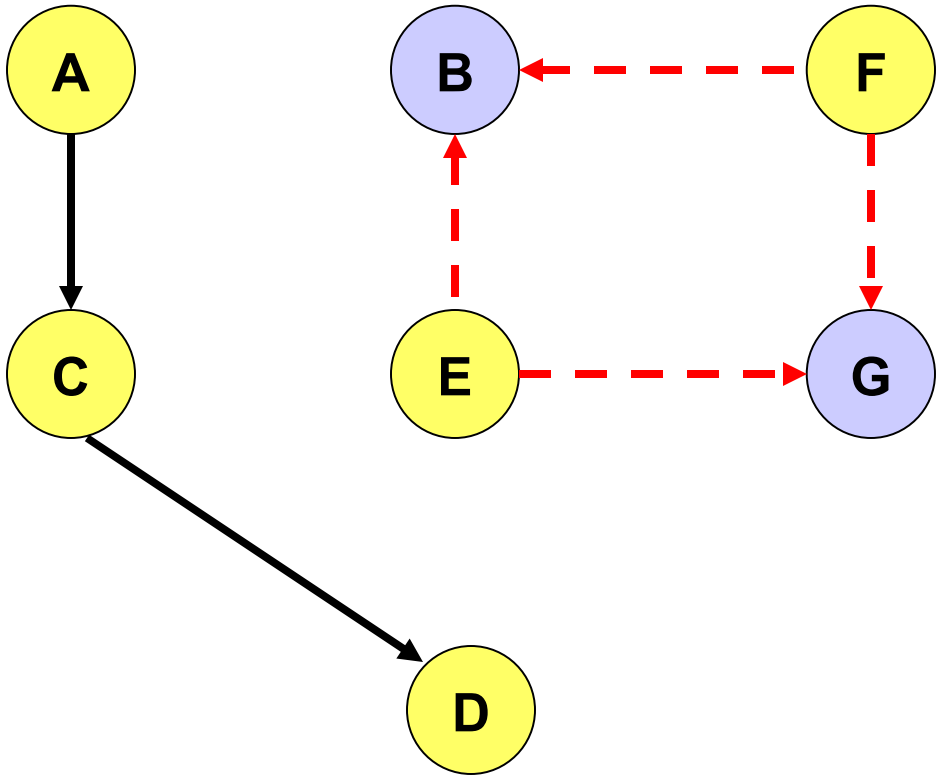
A and G do not have outgoing links

Full Reversal in a Partitioned Network



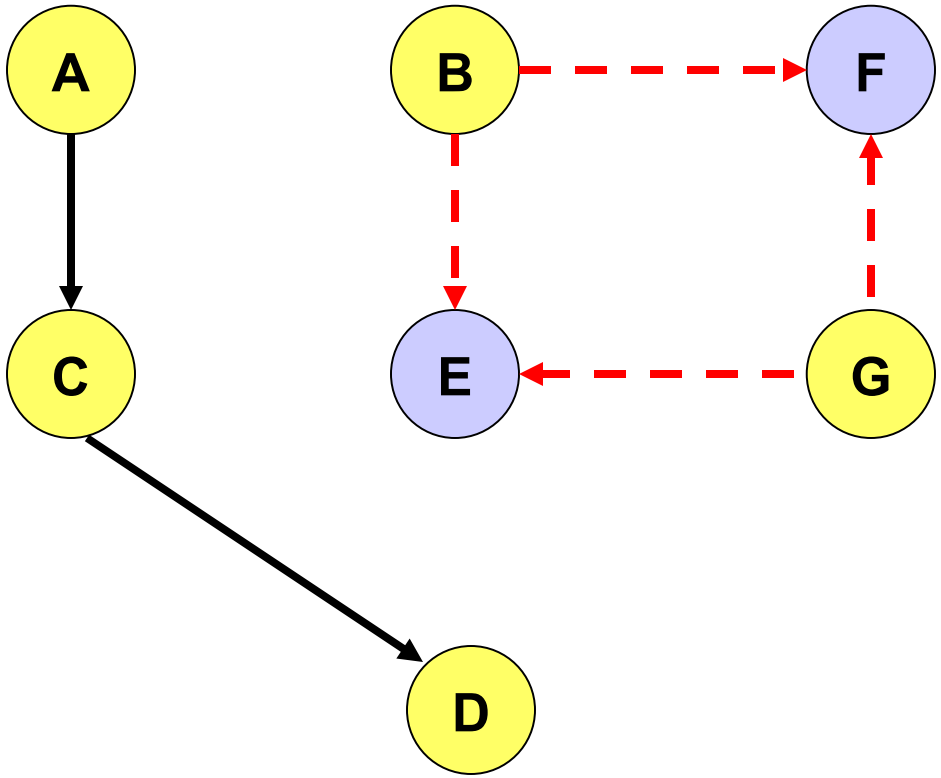
E and F do not have outgoing links

Full Reversal in a Partitioned Network



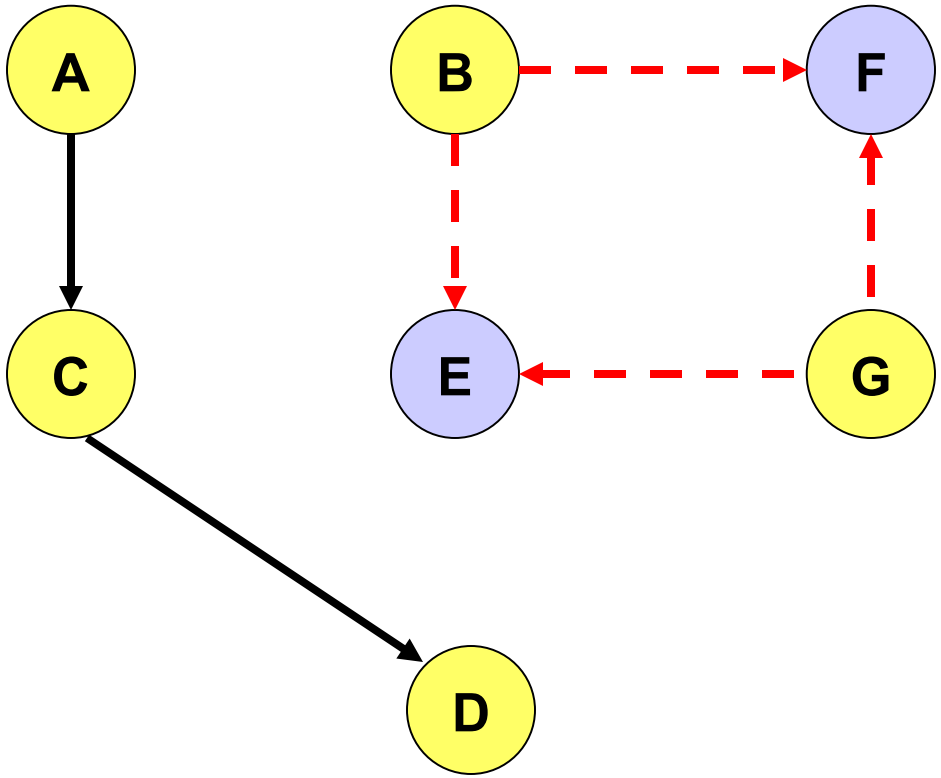
B and G do not have outgoing links

Full Reversal in a Partitioned Network



E and F do not have outgoing links

Full Reversal in a Partitioned Network



In the partition disconnected from destination D, link reversals continue, until the partitions merge

Need a mechanism to minimize this wasteful activity

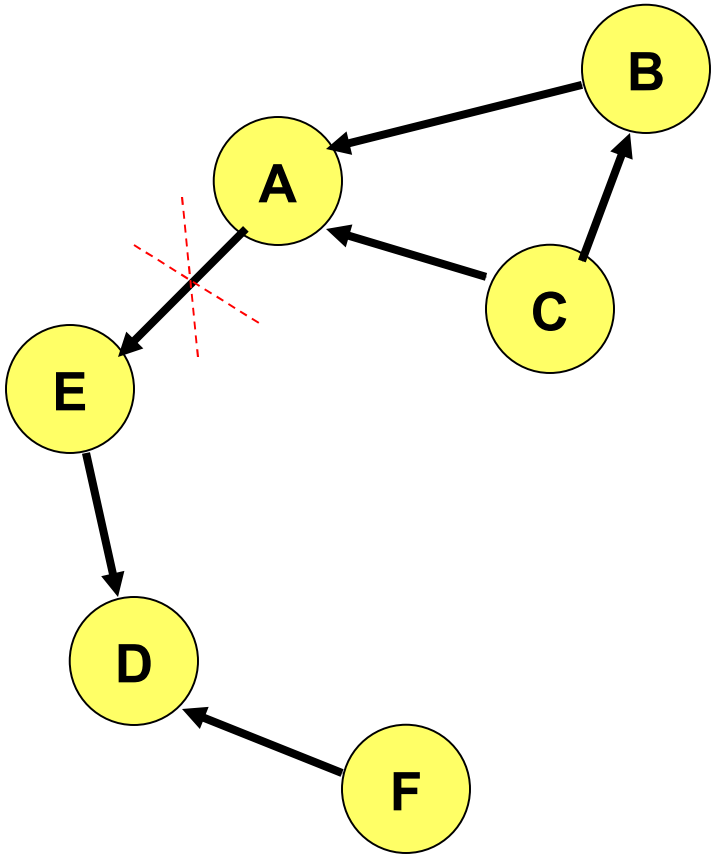
Similar scenario can occur with partial reversal method too

Temporally-Ordered Routing Algorithm (TORA) [Park97Infocom]

TORA modifies the partial link reversal method to be able to detect partitions

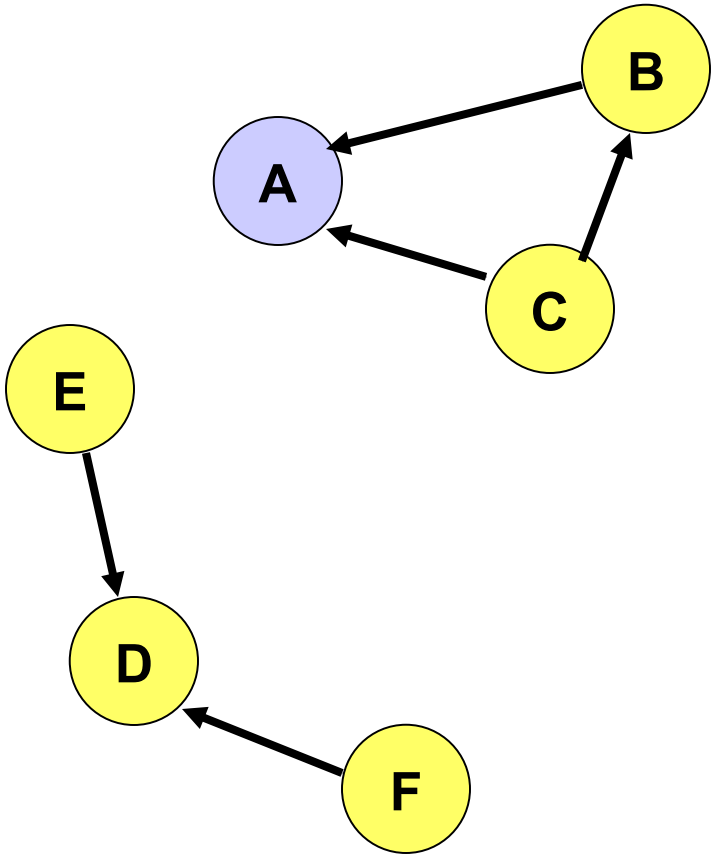
When a partition is detected, all nodes in the partition are informed, and link reversals in that partition cease

Partition Detection in TORA



DAG for destination D

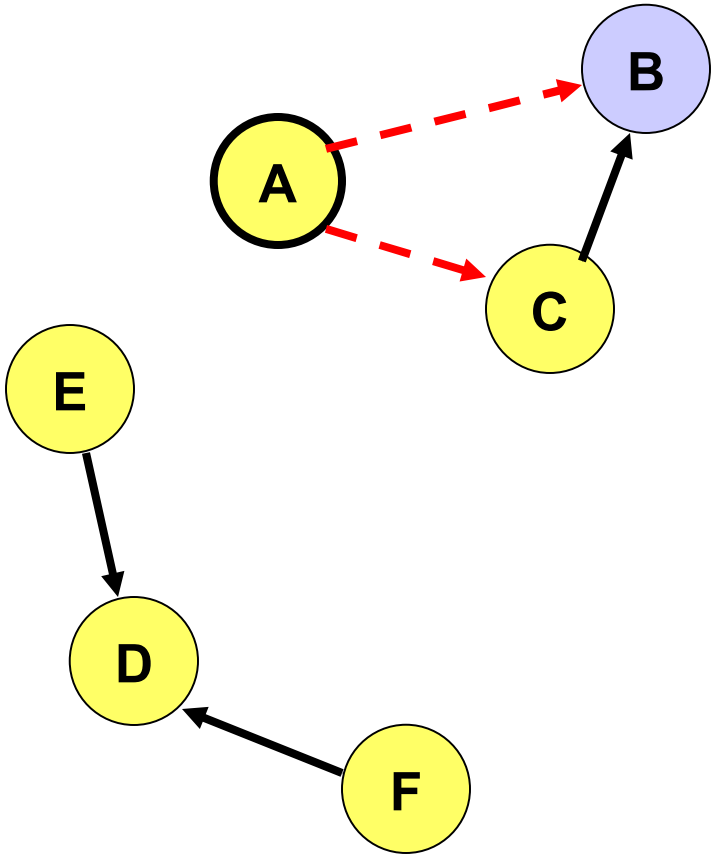
Partition Detection in TORA



TORA uses a modified partial reversal method

Node A has no outgoing links

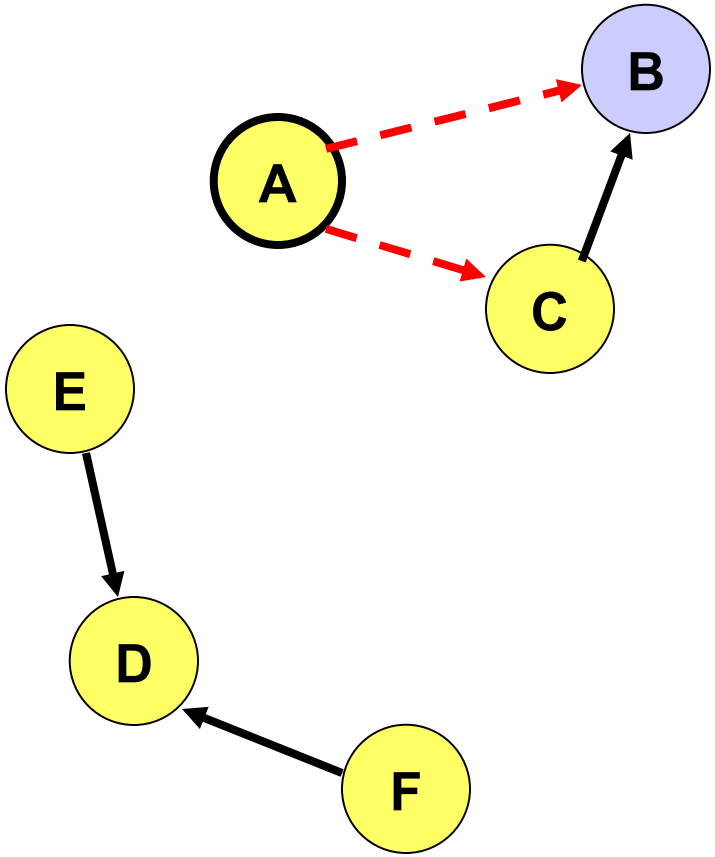
Partition Detection in TORA



TORA uses a modified partial reversal method

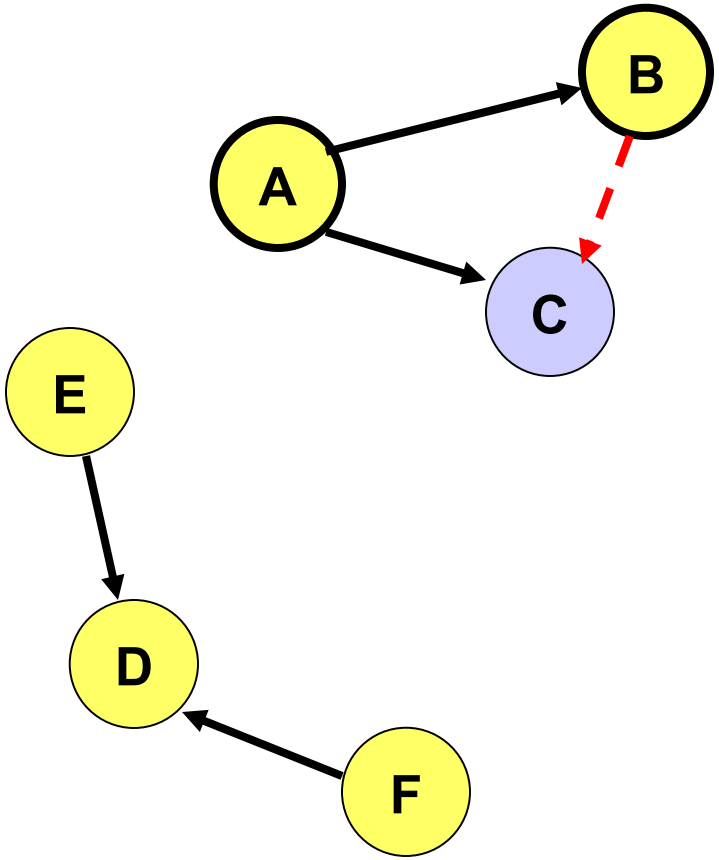
Node B has no outgoing links

Partition Detection in TORA



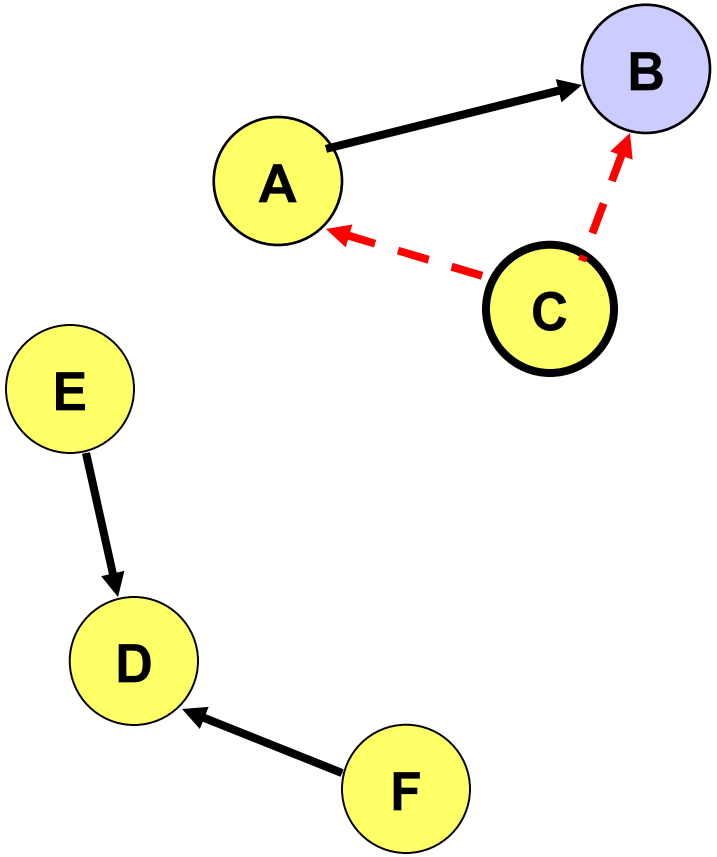
Node B has no outgoing links

Partition Detection in TORA



Node C has no outgoing links -- all its neighbor have reversed links previously.

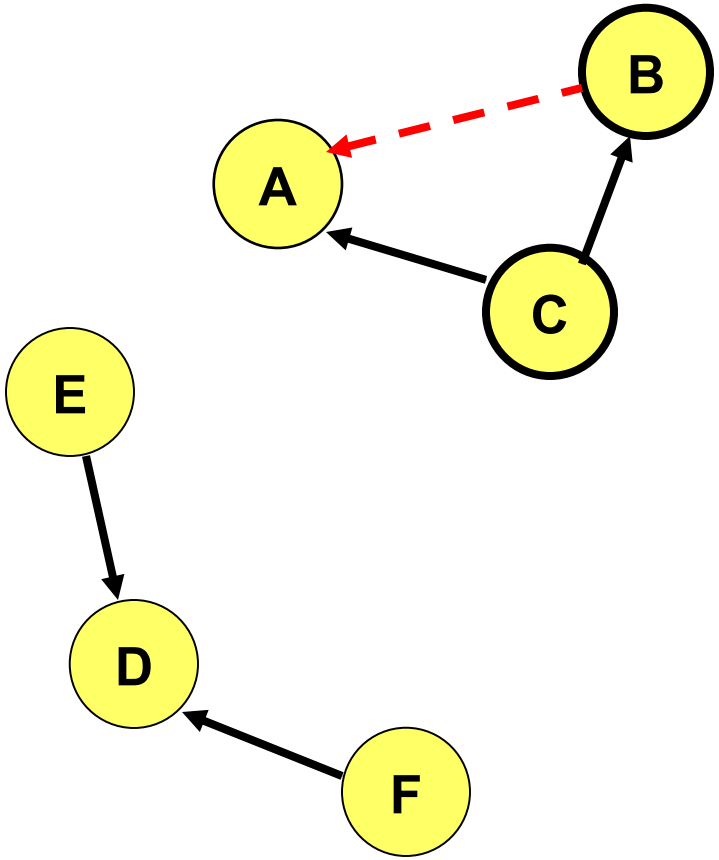
Partition Detection in TORA



Nodes A and B receive the reflection from node C

Node B now has no outgoing link

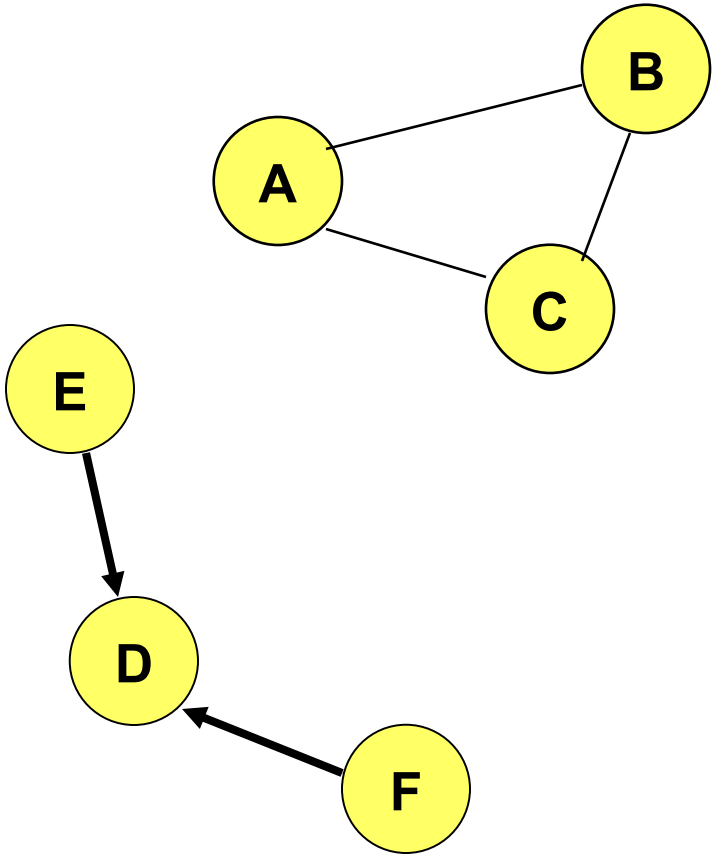
Partition Detection in TORA



Node B propagates the reflection to node A

Node A has received the reflection from all its neighbors.
Node A determines that it is partitioned from destination D.

Partition Detection in TORA



On detecting a partition, node A sends a clear (CLR) message that purges all directed links in that partition

TORA

Improves on the partial link reversal method in [Gafni81] by detecting partitions and stopping non-productive link reversals

Paths may not be shortest

The DAG provides many hosts the ability to send packets to a given destination

- Beneficial when many hosts want to communicate with a single destination

TORA Design Decision

TORA performs link reversals as dictated by [Gafni81]

However, when a link breaks, it looses its direction

When a link is repaired, it may not be assigned a direction, unless some node has performed a route discovery after the link broke

- if no one wants to send packets to D anymore, eventually, the DAG for destination D may disappear

TORA makes effort to maintain the DAG for D only if someone needs route to D

- Reactive behavior

TORA Design Decision

One proposal for modifying TORA optionally allowed a more proactive behavior, such that a DAG would be maintained even if no node is attempting to transmit to the destination

Moral of the story: The link reversal algorithm in [Gafni81] does not dictate a proactive or reactive response to link failure/repair

Decision on reactive/proactive behavior should be made based on environment under consideration

So far ...

All nodes had identical responsibilities

Some schemes propose giving special responsibilities to a subset of nodes

- “Core” based schemes assign additional tasks to nodes belonging to the “core
- Clustering schemes assign additional tasks to cluster “leaders”

Not discussed further in this tutorial

Proactive Protocols

Proactive Protocols

Most of the schemes discussed so far are reactive

Proactive schemes based on distance-vector and link-state mechanisms have also been proposed

Link State Routing [Huitema95]

Each node periodically floods status of its links

Each node re-broadcasts link state information received from its neighbor

Each node keeps track of link state information received from other nodes

Each node uses above information to determine next hop to each destination

Optimized Link State Routing (OLSR)
[Jacquet00ietf,Jacquet99Inria]

The overhead of flooding link state information is reduced by requiring fewer nodes to forward the information

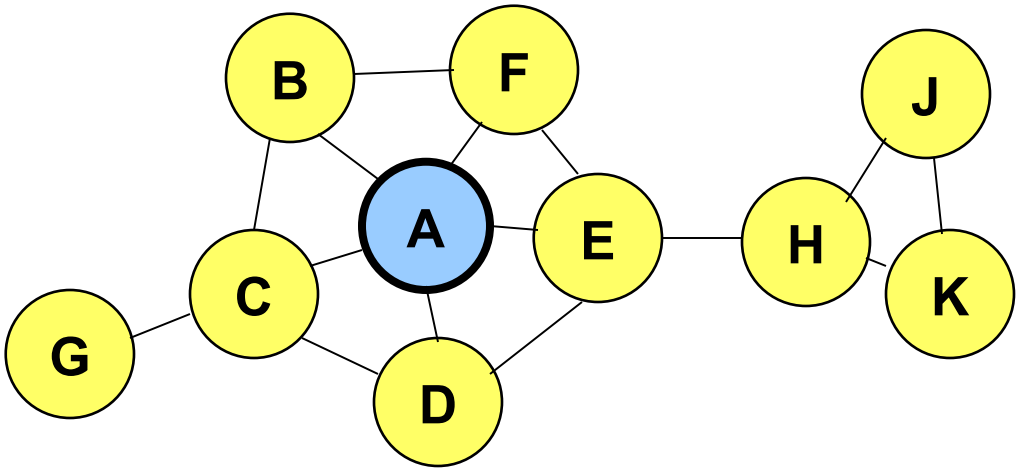
A broadcast from node X is only forwarded by its *multipoint relays*

Multipoint relays of node X are its neighbors such that each two-hop neighbor of X is a one-hop neighbor of at least one multipoint relay of X

- Each node transmits its neighbor list in periodic beacons, so that all nodes can know their 2-hop neighbors, in order to choose the multipoint relays

Optimized Link State Routing (OLSR)

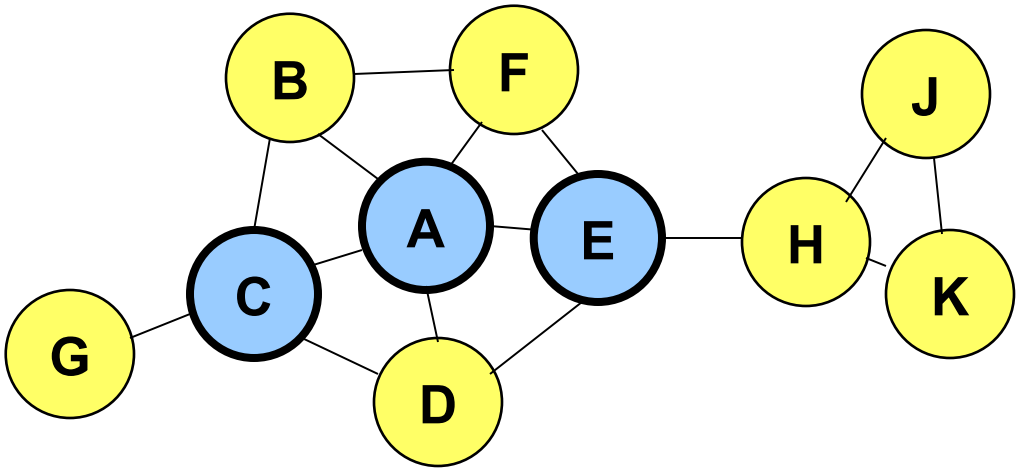
Nodes C and E are multipoint relays of node A



Node that has broadcast state information from A

Optimized Link State Routing (OLSR)

Nodes C and E forward information received from A



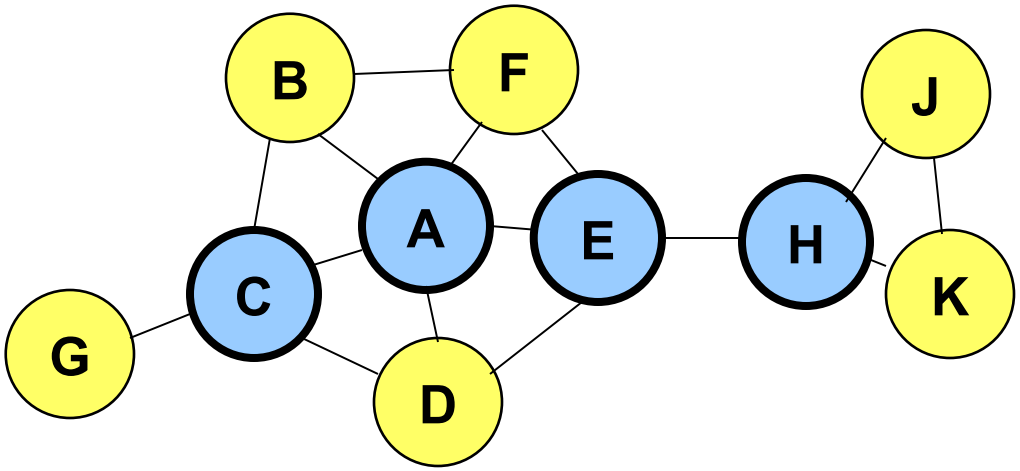
Node that has broadcast state information from A

Optimized Link State Routing (OLSR)

Nodes E and K are multipoint relays for node H

Node K forwards information received from H

E has already forwarded the same information once



Node that has broadcast state information from A

OLSR

OLSR floods information through the multipoint relays

The flooded information itself is for links connecting nodes to respective multipoint relays

Routes used by OLSR only include multipoint relays as intermediate nodes

Destination-Sequenced Distance-Vector (DSDV) [Perkins94Sigcomm]

Each node maintains a routing table which stores

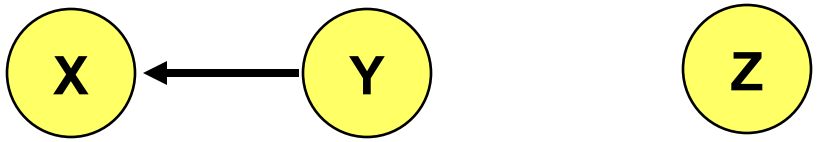
- next hop towards each destination
- a cost metric for the path to each destination
- a destination sequence number that is created by the destination itself
- Sequence numbers used to avoid formation of loops

Each node periodically forwards the routing table to its neighbors

- Each node increments and appends its sequence number when sending its local routing table
- This sequence number will be attached to route entries created for this node

Destination-Sequenced Distance-Vector (DSDV)

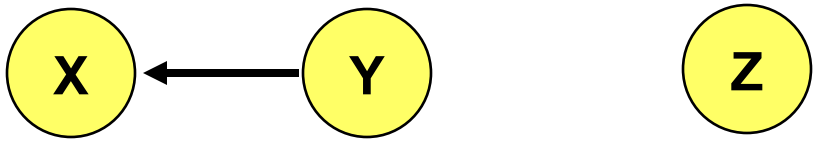
Assume that node X receives routing information from Y about a route to node Z



Let S(X) and S(Y) denote the destination sequence number for node Z as stored at node X, and as sent by node Y with its routing table to node X, respectively

Destination-Sequenced Distance-Vector (DSDV)

Node X takes the following steps:



If S(X) > S(Y), then X ignores the routing information received from Y

If S(X) = S(Y), and cost of going through Y is smaller than the route known to X, then X sets Y as the next hop to Z

If S(X) < S(Y), then X sets Y as the next hop to Z, and S(X) is updated to equal S(Y)

Hybrid Protocols

Zone Routing Protocol (ZRP) [Haas98]

Zone routing protocol combines

Proactive protocol: which pro-actively updates network state and maintains route regardless of whether any data traffic exists or not

Reactive protocol: which only determines route to a destination if there is some data to be sent to the destination

ZRP

All nodes within hop distance at most *d* from a node X are said to be in the routing zone of node X

All nodes at hop distance exactly *d* are said to be peripheral nodes of node X's routing zone

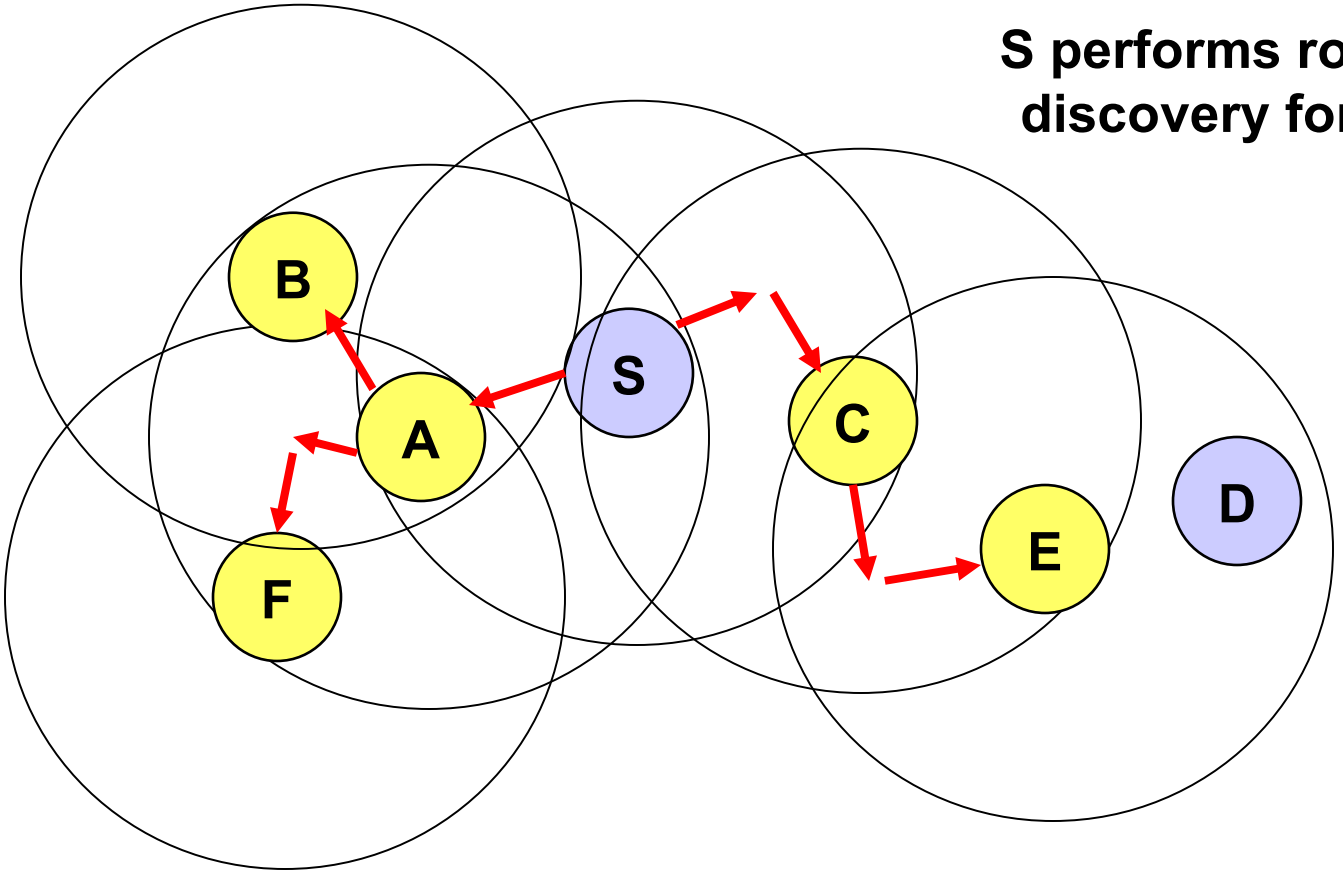
ZRP

Intra-zone routing: Pro-actively maintain state information for links within a short distance from any given node

- Routes to nodes within short distance are thus maintained proactively (using, say, link state or distance vector protocol)

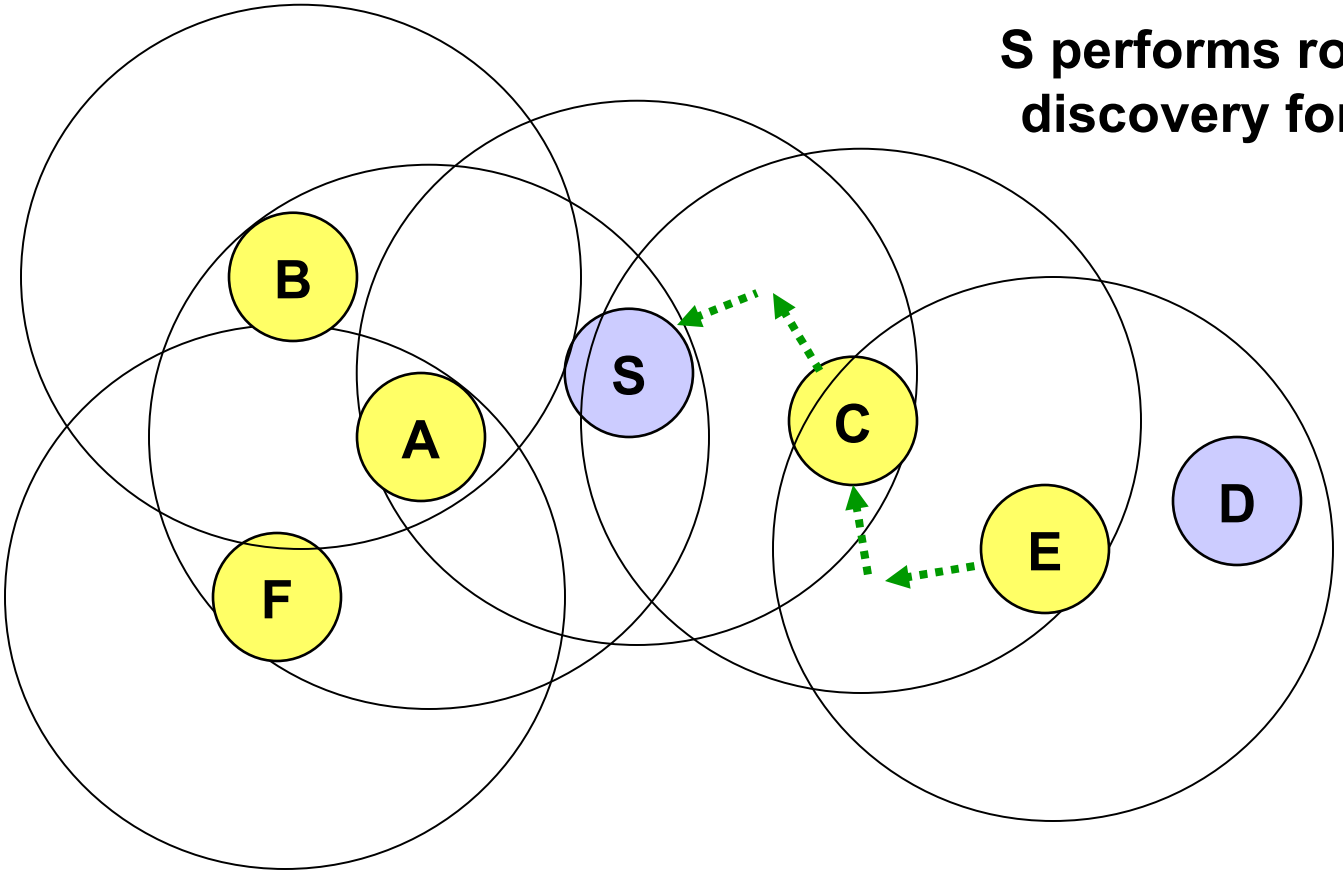
Inter-zone routing: Use a route discovery protocol for determining routes to far away nodes. Route discovery is similar to DSR with the exception that route requests are propagated via peripheral nodes.

ZRP: Example with Zone Radius = *d* = 2

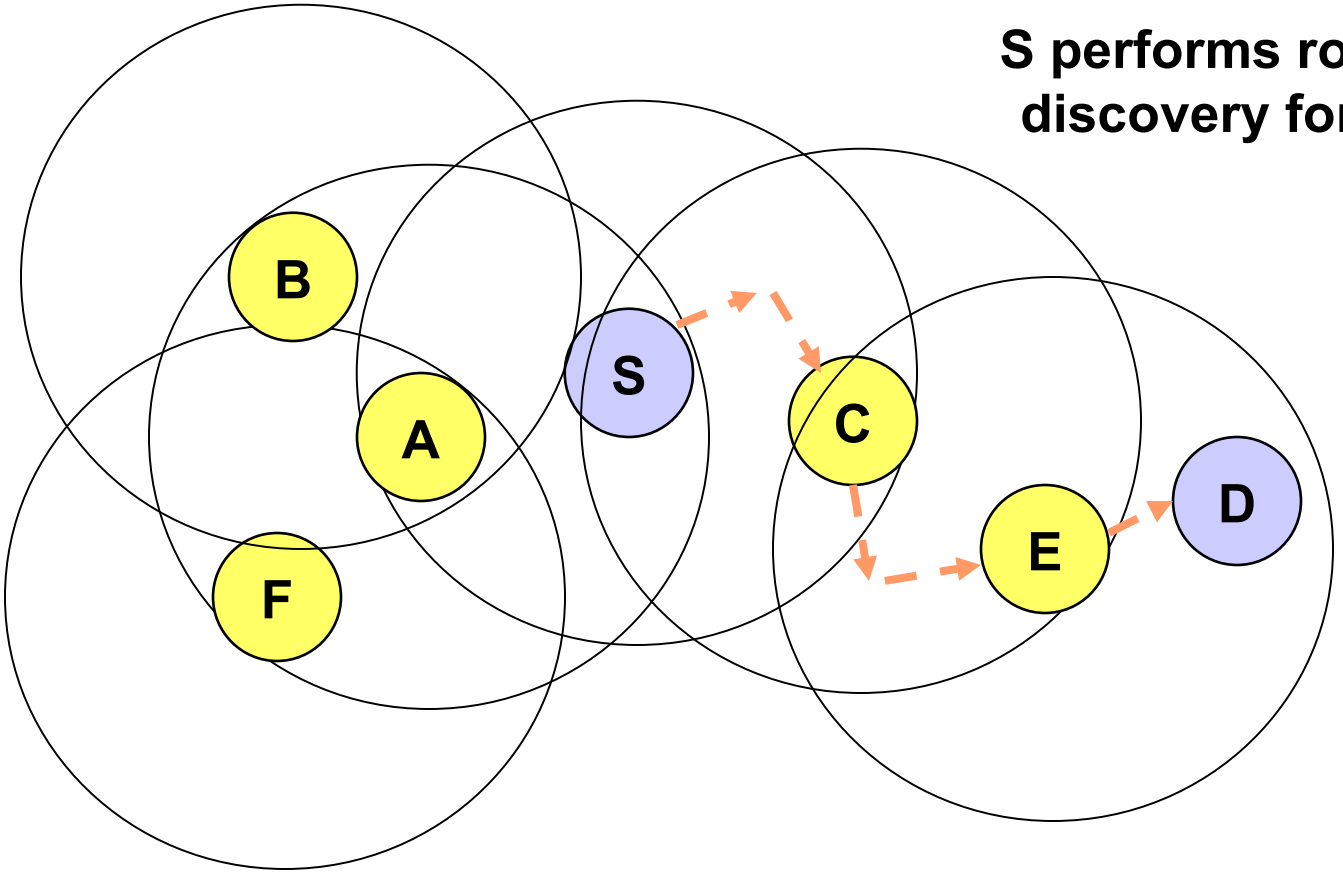


→ **Denotes route request**

ZRP: Example with *d* = 2



ZRP: Example with *d* = 2



Denotes route taken by Data

Landmark Routing (LANMAR) for MANET with Group Mobility [Pei00Mobihoc]

A *landmark* node is elected for a group of nodes that are likely to move together

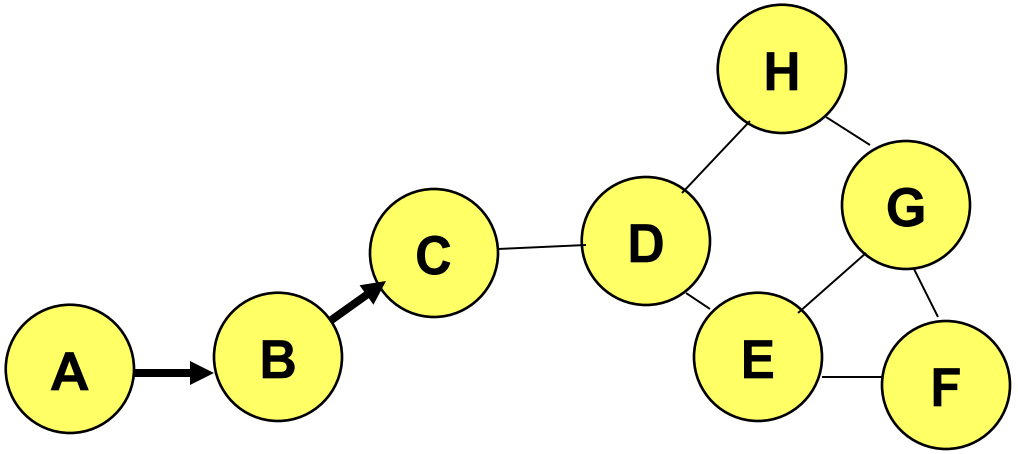
A *scope* is defined such that each node would typically be within the scope of its landmark node

Each node propagates *link state* information corresponding only to nodes within it *scope* and *distance-vector* information for all *landmark* nodes

- Combination of link-state and distance-vector
- Distance-vector used for landmark nodes outside the scope
- No state information for non-landmark nodes outside scope maintained

LANMAR Routing to Nodes Within Scope

Assume that node C is within scope of node A

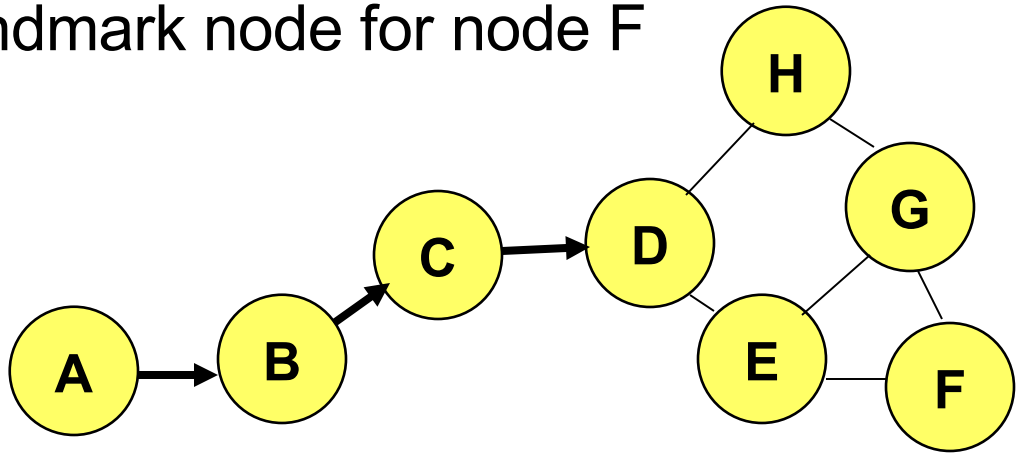


Routing from A to C: Node A can determine next hop to node C using the available link state information

LANMAR Routing to Nodes Outside Scope

Routing from node A to F, which is outside A's scope

Let H be the landmark node for node F

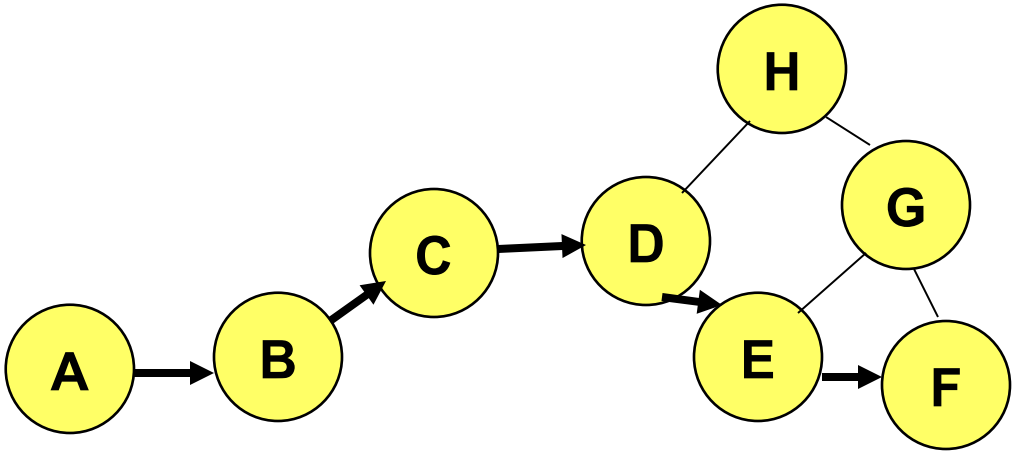


Node A somehow knows that H is the landmark for C

Node A can determine next hop to node H using the available distance vector information

LANMAR Routing to Nodes Outside Scope

Node D is within scope of node F



Node D can determine next hop to node F using link state information

The packet for F may never reach the landmark node H, even though initially node A sends it towards H

LANMAR scheme uses node identifiers as landmarks

Anchored Geodesic Scheme [LeBoudec00] uses geographical regions as landmarks

Routing

Protocols discussed so far find/maintain a route provided it exists

Some protocols attempt to ensure that a route exists by

- Power Control [Ramanathan00Infocom]
- Limiting movement of hosts or forcing them to take detours [Reuben98thesis]

Power Control

Protocols discussed so far find a route, on a *given* network topology

Some researchers propose *controlling* network topology by transmission power control to yield network properties which may be desirable [Ramanathan00Infocom]

- Such approaches can significantly impact performance at several layers of protocol stack

[Wattwnhofer00Infocom] provides a distributed mechanism for power control which allows for local decisions, but guarantees global connectivity

- Each node uses a power level that ensures that the node has at least one neighbor in each *cone* with angle $2\pi/3$

Some Variations

Power-Aware Routing
[Singh98Mobicom,Chang00Infocom]

Define optimization criteria as a function of energy consumption. Examples:

Minimize energy consumed per packet

Minimize time to network partition due to energy depletion

Maximize duration before a node fails due to energy depletion

Power-Aware Routing [Singh98Mobicom]

Assign a weight to each link

Weight of a link may be a function of energy consumed when transmitting a packet on that link, as well as the residual energy level

- low residual energy level may correspond to a high cost

Prefer a route with the smallest aggregate weight

Power-Aware Routing

Possible modification to DSR to make it power aware (for simplicity, assume no route caching):

Route Requests aggregate the weights of all traversed links

Destination responds with a Route Reply to a Route Request if

- it is the first RREQ with a given (“current”) sequence number, or
- its weight is smaller than all other RREQs received with the current sequence number

Preemptive Routing [Goff01MobiCom]

Add some proactivity to reactive routing protocols such as DSR and AODV

Route discovery initiated when it appears that an active route will break in the near future

Initiating route discover *before* existing route breaks reduces discovery latency

Performance of Unicast Routing in MANET

Several performance comparisons [Broch98Mobicom,Johansson99Mobicom,Das00Infocom,Das98ic3n]

We will discuss performance issue later in the tutorial

Address Auto-Configuration

Address Auto-configuration

Auto-configuration important for autonomous operation of an ad hoc network

IPv4 and *IPv6* auto-configuration mechanisms have been proposed

- Need to be adapted for ad hoc networks

Auto-Configuration in Ad Hoc Networks

Worst case network delays may be unknown, or highly variable

Partitions may occur, and merge

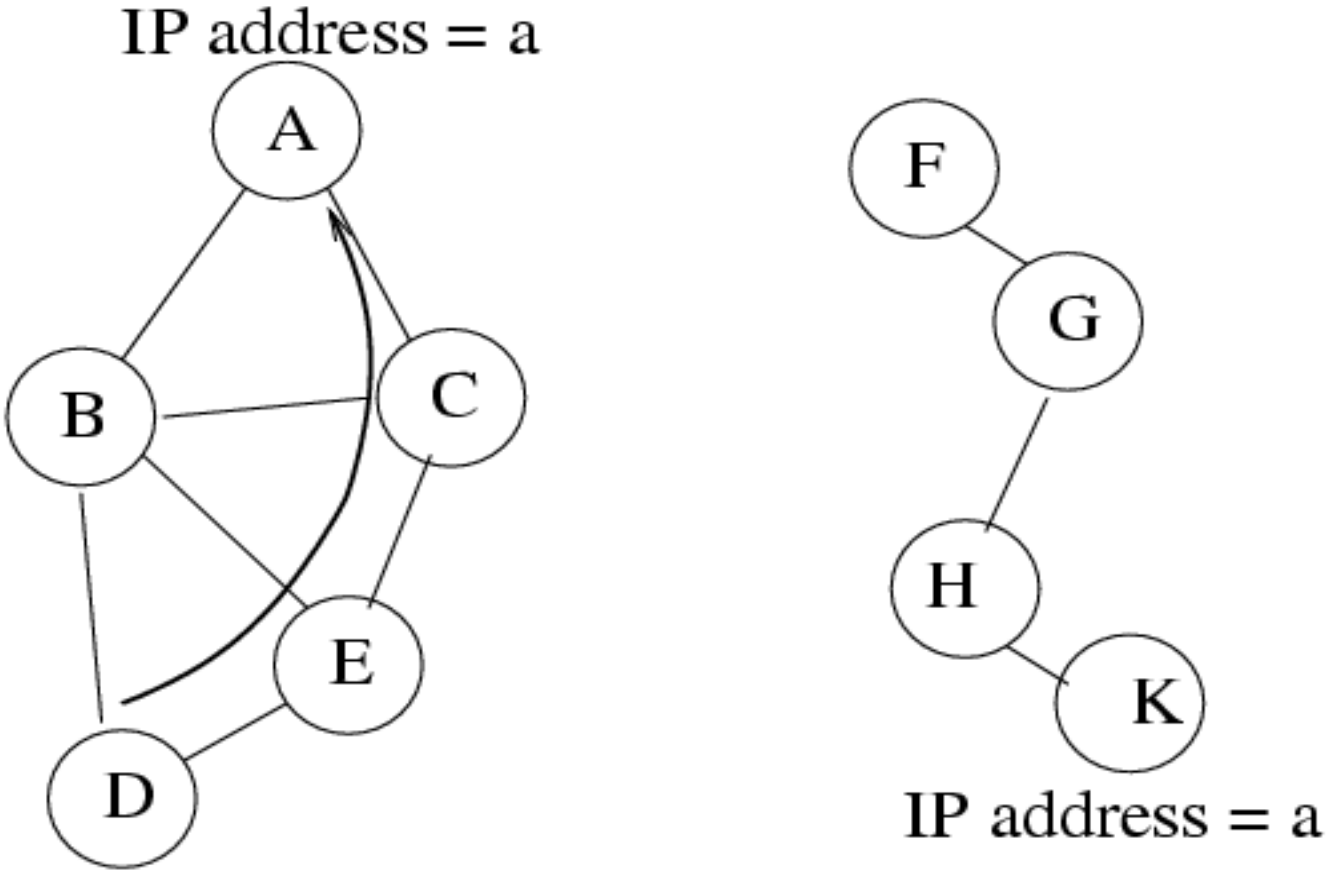
Duplicate Address Detection in Ad Hoc Networks

Several proposals

One example [Perkins]:

- Host picks an address randomly
- Host performs route discovery for the chosen address
- If a route reply is received, address duplication is detected

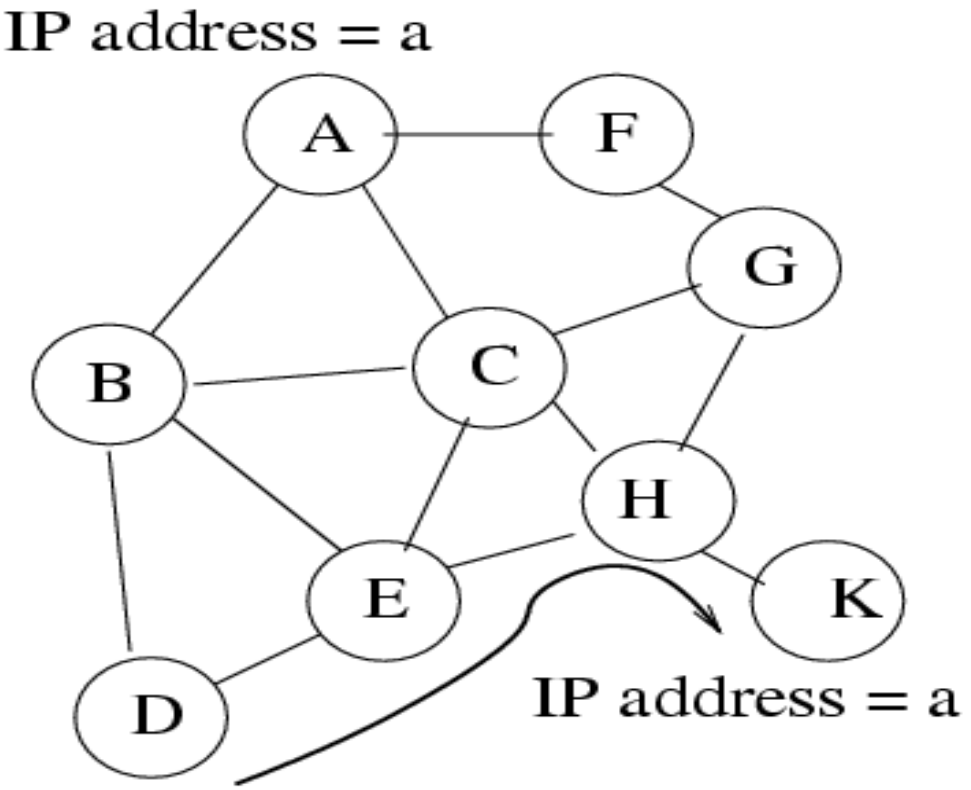
Example:
Initially Partitioned Network



D's packets for address *a* routed to A

Merged Network

Duplicate address detection (DAD) important To avoid misrouting



Strong DAD

Detect duplicate addresses within *t* seconds

Not possible to guarantee strong DAD in presence of unbounded delays

- May occur due to partitions
- Even when delays are bounded, bound may be difficult to calculate
  - Unknown network size

DAD

Strong DAD impossible with unbounded delay

How to achieve DAD ?

Design Principle

If you cannot solve a problem

Change the problem

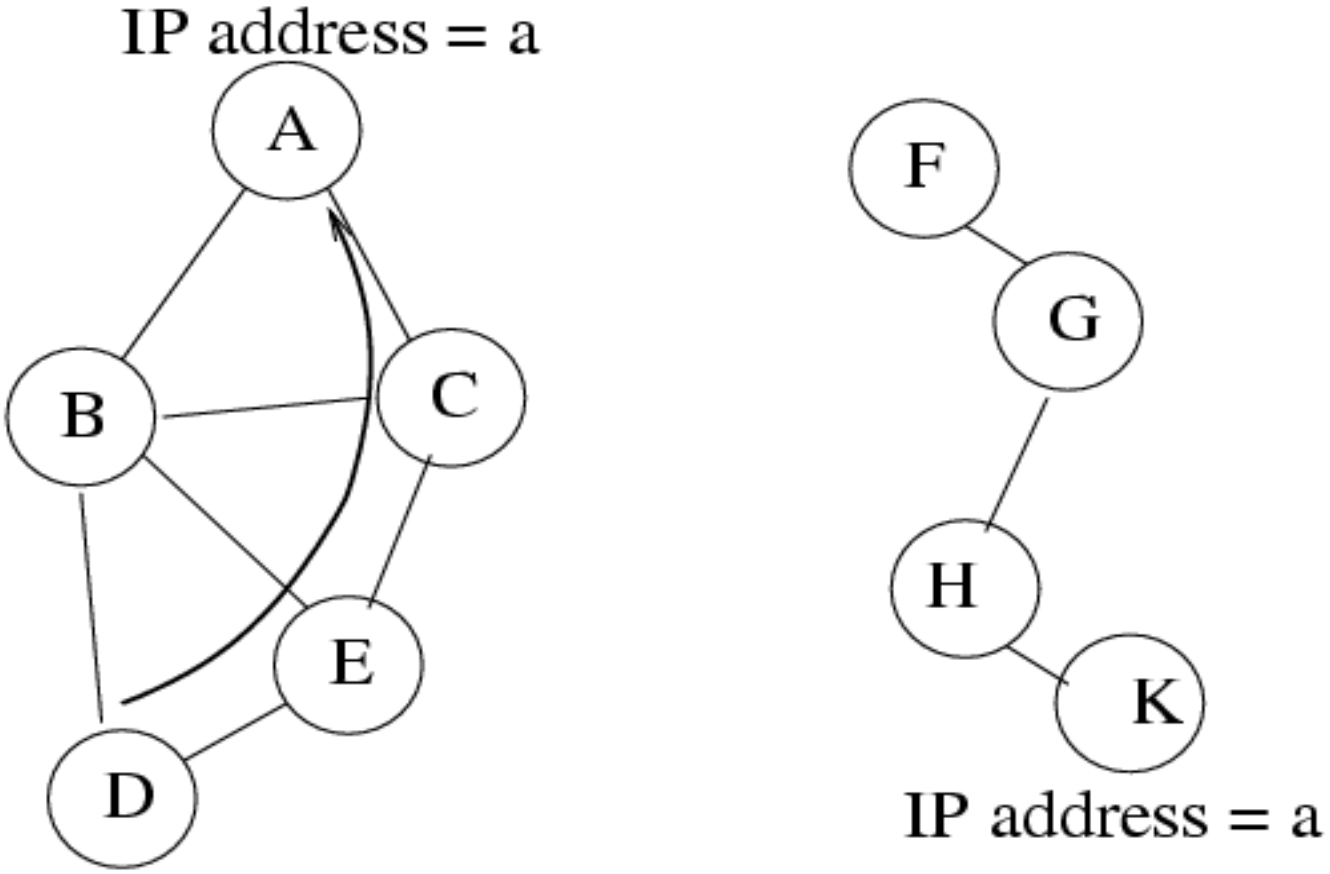
Weak DAD [Vaidya02MobiHoc]

Packets from a given host to a given *address*

should be routed to the same *destination*,

despite duplication of the *address*

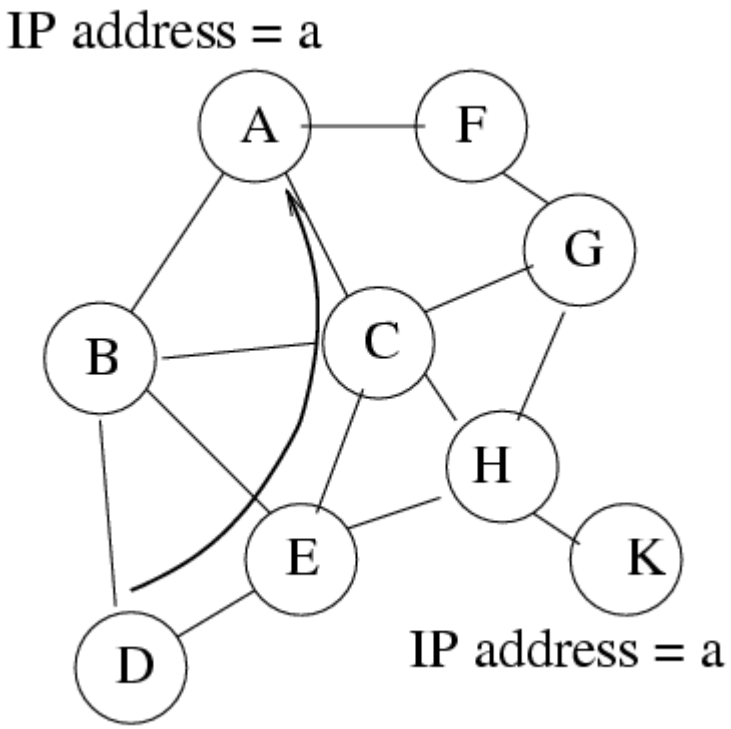
Example:
Initially Partitioned Network



D's packets for address *a* routed to A

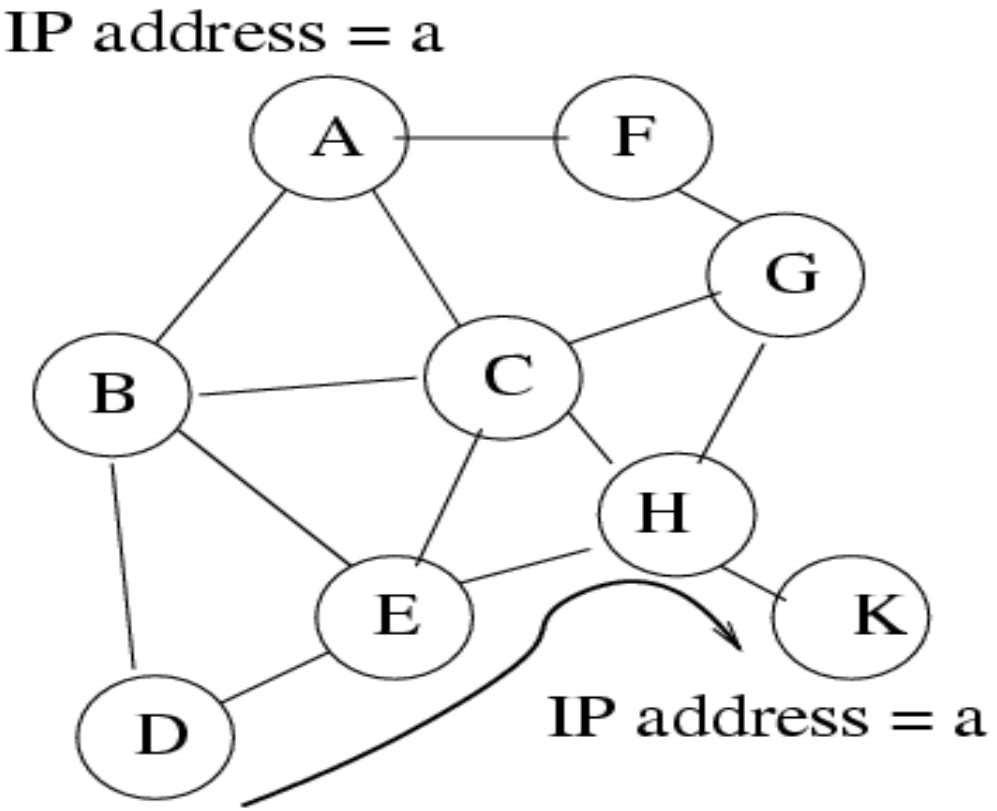
Merged Network: Acceptable Behavior with Weak DAD

Packets from D to address *a* still routed to host A



Merged Network: Unacceptable behavior

Packets from D to address *a* routed to host K instead of A



Weak DAD: Implementation

Integrate duplicate address detection with route maintenance

SKIP

Weak DAD with Link State Routing

Each host has a unique (with high probability) key

- May include MAC address, serial number, …
- May be large in size

In all routing-related packets (link state updates) IP addresses tagged by keys

- (IP, key) pair

Weak DAD with Link State Routing

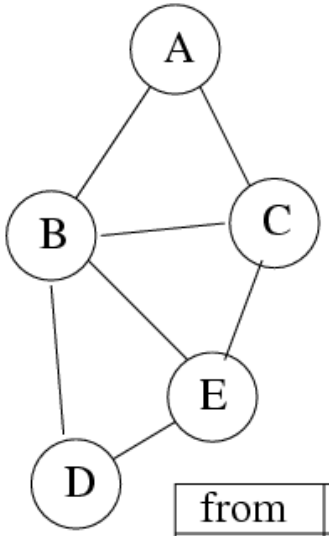
Address duplication not always detected

Duplication detected before misrouting can occur

Weak

➔ Reliable, but potentially delayed, DAD

Link State Routing (LSR): Example



Dest	Next Hop
IP_B	IP_B
IP_C	IP_E
IP_A	IP_B
IP_E	IP_E

Routing table at node D

from	to	cost
IP_D	IP_E	2
IP_D	IP_B	10

link state packet transmitted by D

Weak DAD with LSR

Dest	Key	Next Hop
IP_B	K_B	IP_B
IP_C	K_C	IP_E
IP_A	K_A	IP_B
IP_E	K_E	IP_E

Routing table at node D

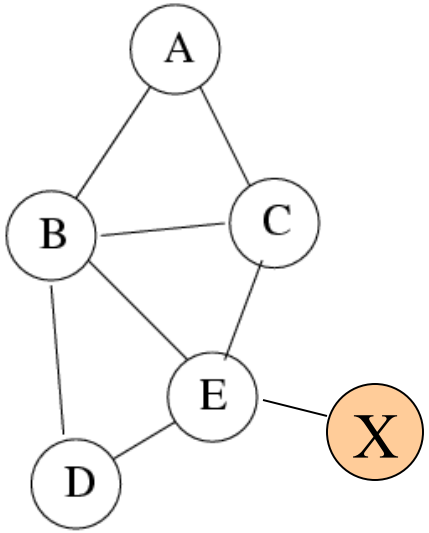
from	key	to	key	cost
IP_D	K_D	IP_E	K_E	2
IP_D	K_D	IP_B	K_B	10

link state packet transmitted by D

Weak DAD with LSR

Dest	Next Hop
IP_B	IP_B
IP_C	IP_E
IP_A	IP_B
IP_E	IP_E

Routing table at node D



Host X with key K_x joins and chooses IP_A

(address duplication)

Weak DAD with LSR

Dest	Key	Next Hop
IP_B	K_B	IP_B
IP_C	K_C	IP_E
IP_A	K_A	IP_B
IP_E	K_E	IP_E

Routing table at node D

If host D receives a link state update containing (IP_A, K_x), host D detects duplication of address IP_A

Two pairs with identical IP address but distinct keys imply duplication

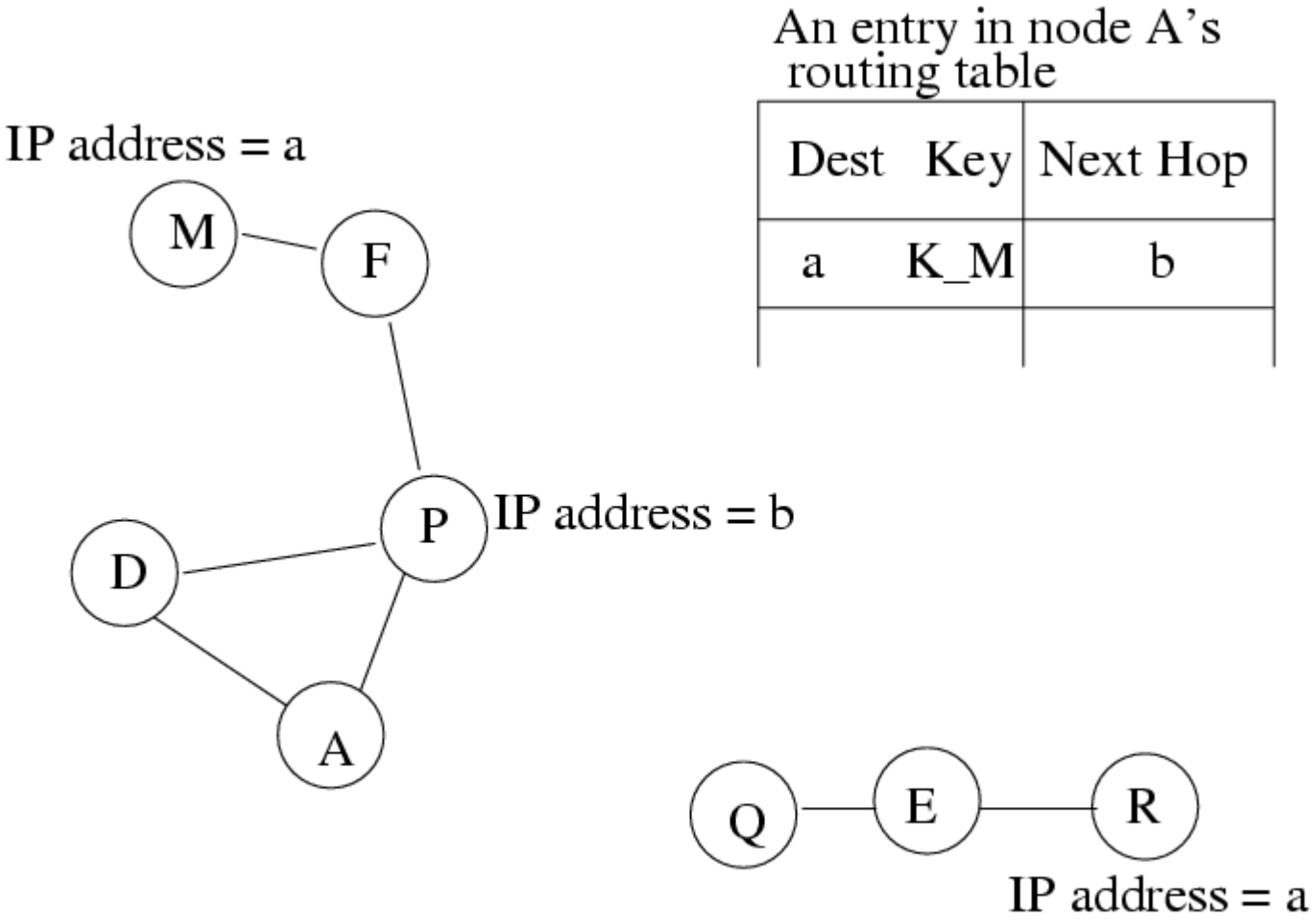
Just-in-Time DAD

Duplication detected before routing tables could be mis-configured

Higher Layer Interaction

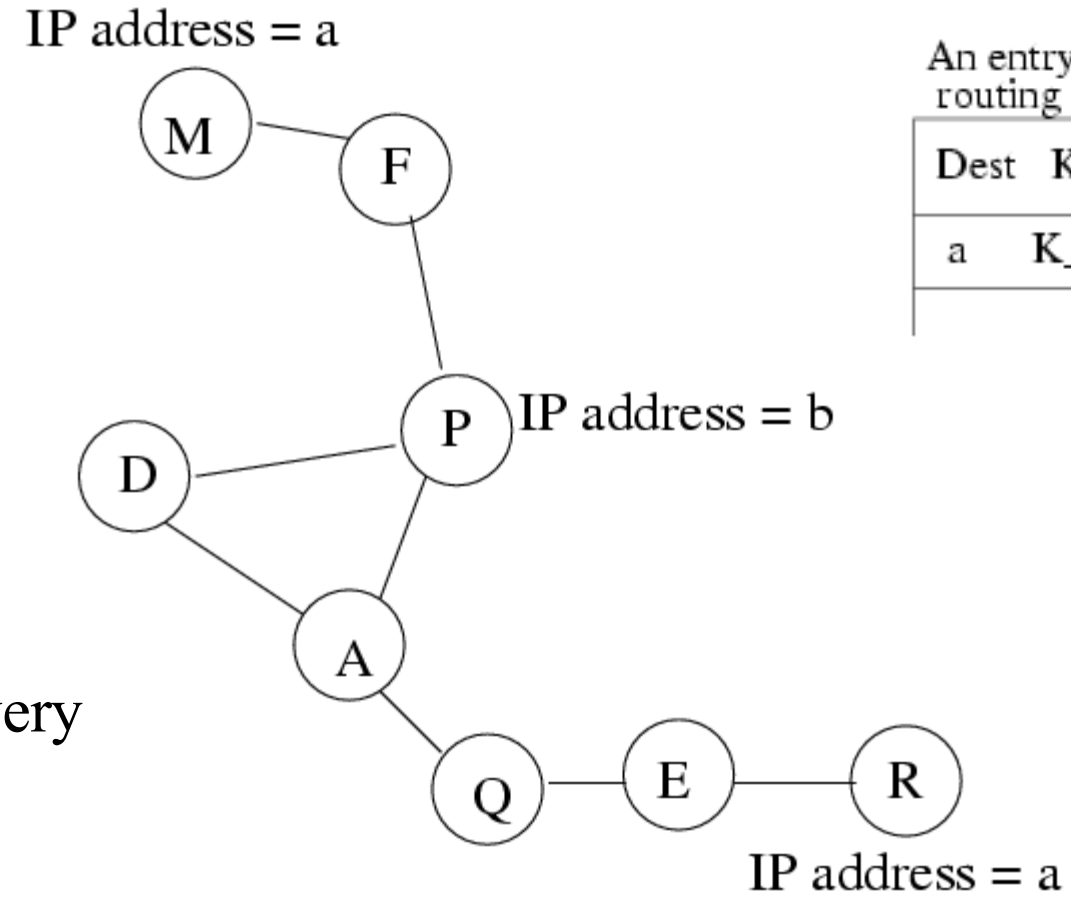
Higher layers interaction may result in undesirable behavior

Example



Q discovers service *Foo* at address *a*

Example: Networks merge

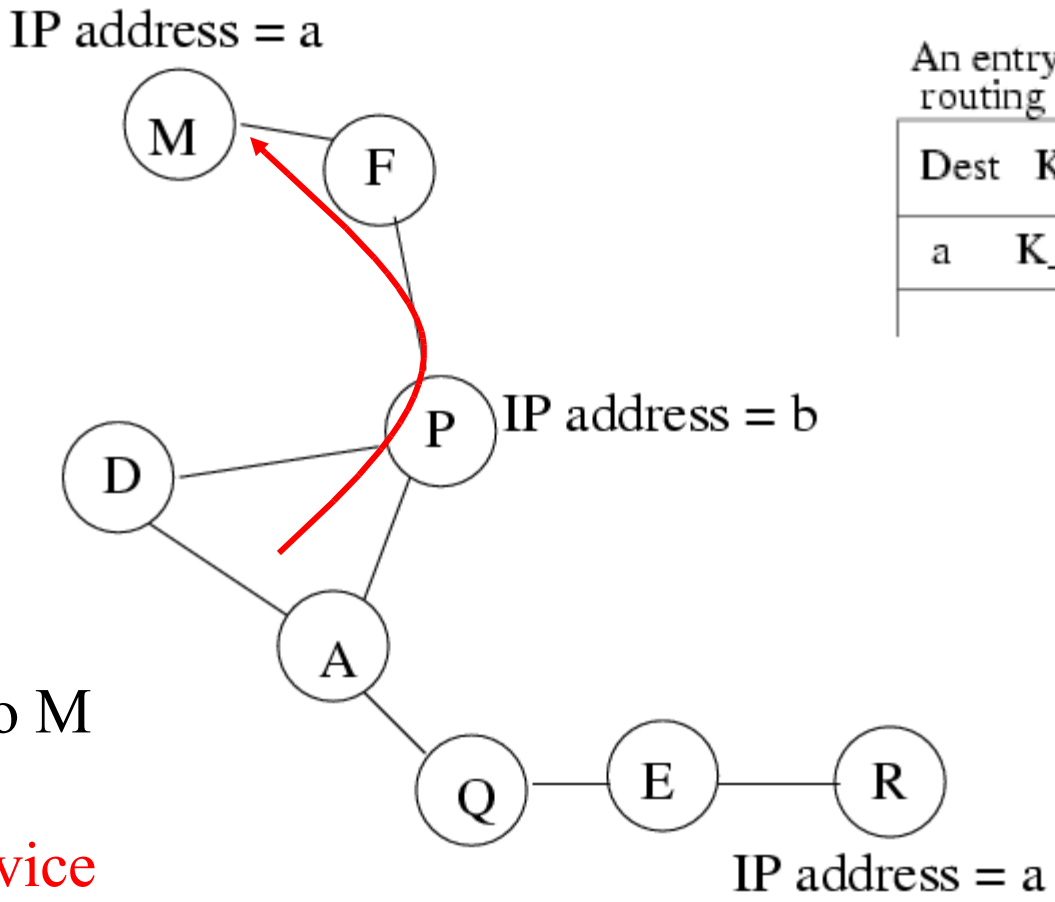


An entry in node A's routing table

Dest Key	Next Hop
a K_M	b

Node A performs service discovery for *Foo*, and learns from Q that *Foo* is available at address *a*

Example: Networks merge



An entry in node A's routing table

Dest Key	Next Hop
a K_M	b

Node A's packets to a are delivered to M

R provides service *Foo* not M

Enhanced Weak DAD

If the status of host A above the network layer depends on state of host B

(State A → state B)

→ then network layer of host A should be aware of (IP, key) pairs known to B

Enhanced Weak DAD

Works despite upper layer interaction

Weak DAD: Other Issues

Duplicate MAC addresses within two hops of each other bad

- Need a duplicate MAC address detection scheme

Network layers performing unicasts using multicast/flooding

Limited-time address leases

DAD with other routing protocols

Possible. [Vaidya02Mobihoc] also discusses DSR.

Summary

Strong DAD – Not always possible

Weak DAD feasible

- Combines DAD with route maintenance

Overhead of weak DAD

- Expected to be low

Capacity of Ad Hoc Networks

Capacity of Fixed Ad Hoc Networks

[Gupta00it]

n nodes in area A transmitting at W bits/sec using a fixed range (distance between a random pair of nodes is O(sqrt(n))

Bit-distance product that can be transported by the network per second is

Θ (W sqrt (A n))

Throughput per node

Θ (W / sqrt (n))

Capacity of Mobile Ad Hoc Networks
[Grossglauser01Infocom]

Assume random motion

Any two nodes become neighbors once in a while

Each node assumed sender for one *session,* and destination for another *session*

Relay packets through at most one other node

- Packet go from S to D directly, when S and D are neighbors, or from S to a relay and the the relay to D, when each pair becomes neighbor respectively

Throughput of each session is O(1)

- Independent of *n*

Continues from last slide …

Delay in packet delivery can be large if O(1) throughput is to be achieved

- Delay incurred waiting for the destination to arrive close to a relay or the sender

Trade-off between delay and throughput

Measured Capacity [Li01MobiCom]

Confirms intuition

In fixed networks, capacity is higher if average distance between source-destination pairs is small

Measured Scaling Law
[Gupta00]

Measured in static networks

Throughput declines worse with *n* than theoretically predicted

Existing MAC protocols unable to exploit “parallelism” in channel access

Capacity

How to design MAC and routing protocols to approach theoretical capacity ?

Open problem

Medium Access Control Protocols

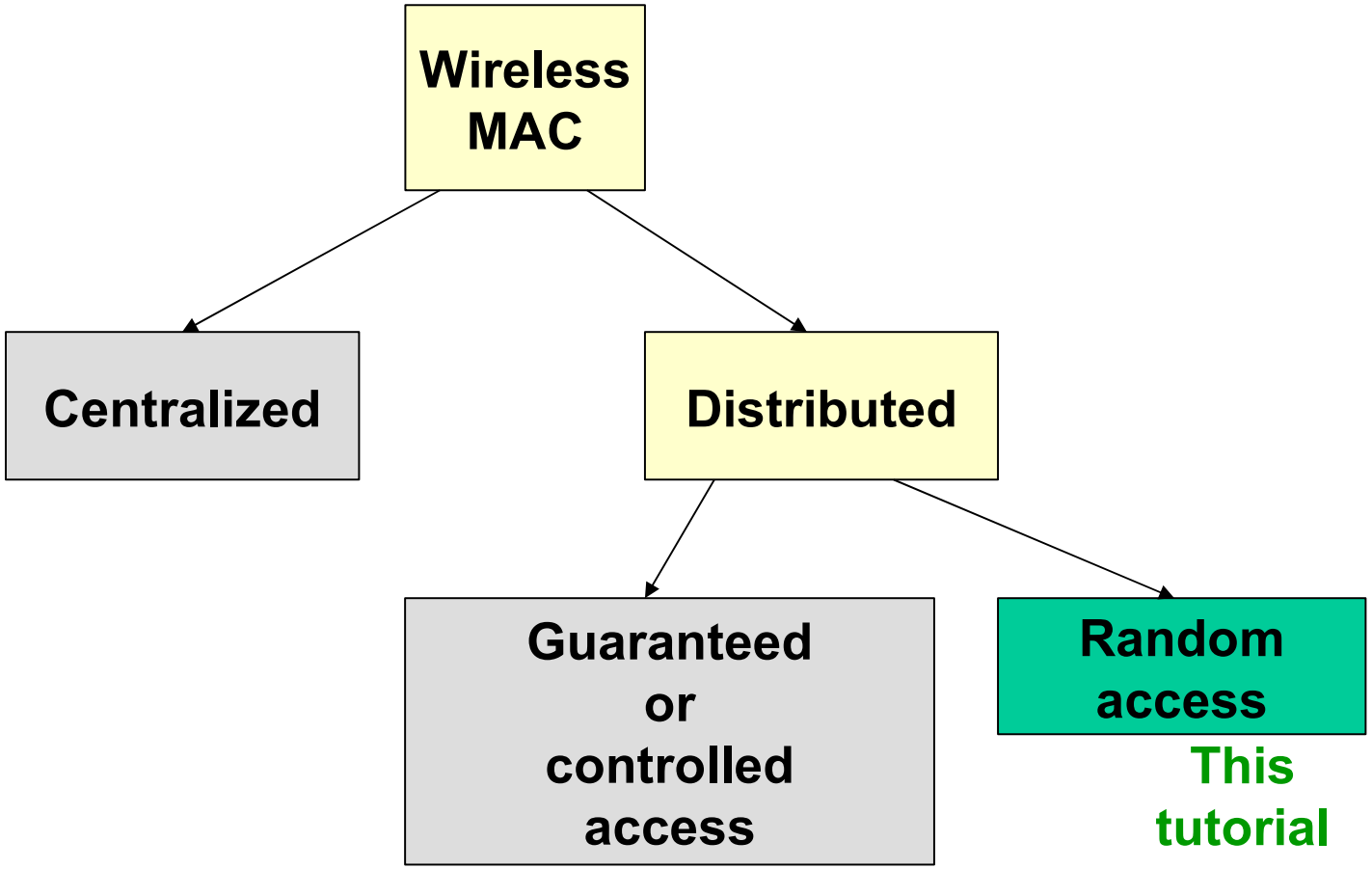
Medium Access Control

Wireless channel is a shared medium

Need access control mechanism to avoid interference

MAC protocol design has been an active area of research for many years [Chandra00survey]

MAC: A Simple Classification



This tutorial

Mostly focus on random access protocols

Not a comprehensive overview of MAC protocols

Provides discussion of some example protocols

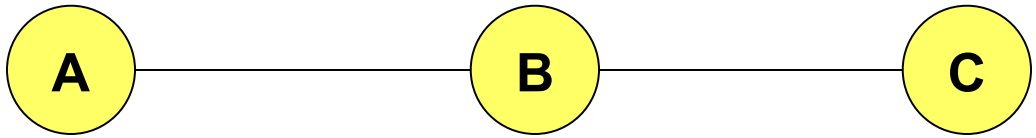
Hidden Terminal Problem [Tobagi75]

Node B can communicate with A and C both

A and C cannot hear each other

When A transmits to B, C cannot detect the transmission using the *carrier sense* mechanism

If C transmits, collision will occur at node B



Busy Tone [Tobagi75,Haas98]

A receiver transmits busy tone when receiving data

All nodes hearing busy tone keep silent

Avoids interference from hidden terminals

Requires a separate channel for busy tone

MACA Solution for Hidden Terminal Problem

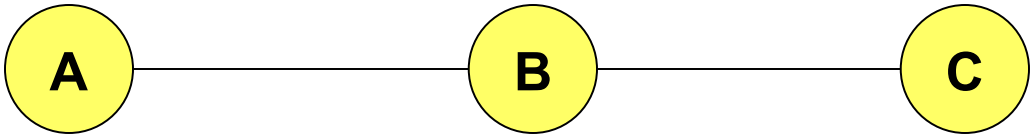
[Karn90]

When node A wants to send a packet to node B, node A first sends a *Request-to-Send (RTS)* to A

On receiving RTS, node A responds by sending *Clear-to-Send (CTS)*, provided node A is able to receive the packet

When a node (such as C) overhears a CTS, it keeps quiet for the duration of the transfer

Transfer duration is included in RTS and CTS both



Reliability

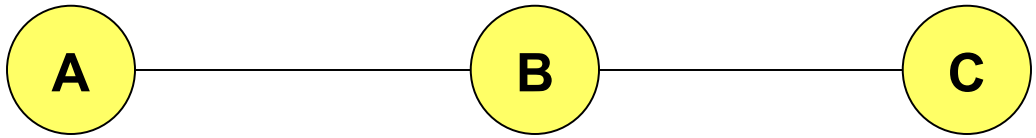
Wireless links are prone to errors. High packet loss rate detrimental to transport-layer performance.

Mechanisms needed to reduce packet loss rate experienced by upper layers

A Simple Solution to Improve Reliability

When node B receives a data packet from node A, node B sends an Acknowledgement (Ack). This approach adopted in many protocols [Bharghavan94,IEEE 802.11]

If node A fails to receive an Ack, it will retransmit the packet



IEEE 802.11 Wireless MAC

Distributed and centralized MAC components

- Distributed Coordination Function (DCF)
- Point Coordination Function (PCF)

DCF suitable for multi-hop ad hoc networking

DCF is a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol

IEEE 802.11 DCF

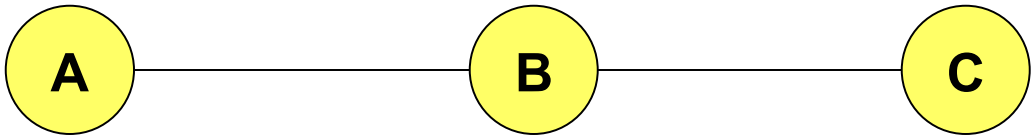
Uses RTS-CTS exchange to avoid hidden terminal problem

- Any node overhearing a CTS cannot transmit for the duration of the transfer

Uses ACK to achieve reliability

Any node receiving the RTS cannot transmit for the duration of the transfer

- To prevent collision with ACK when it arrives at the sender
- When B is sending data to C, node A will keep quite



Collision Avoidance

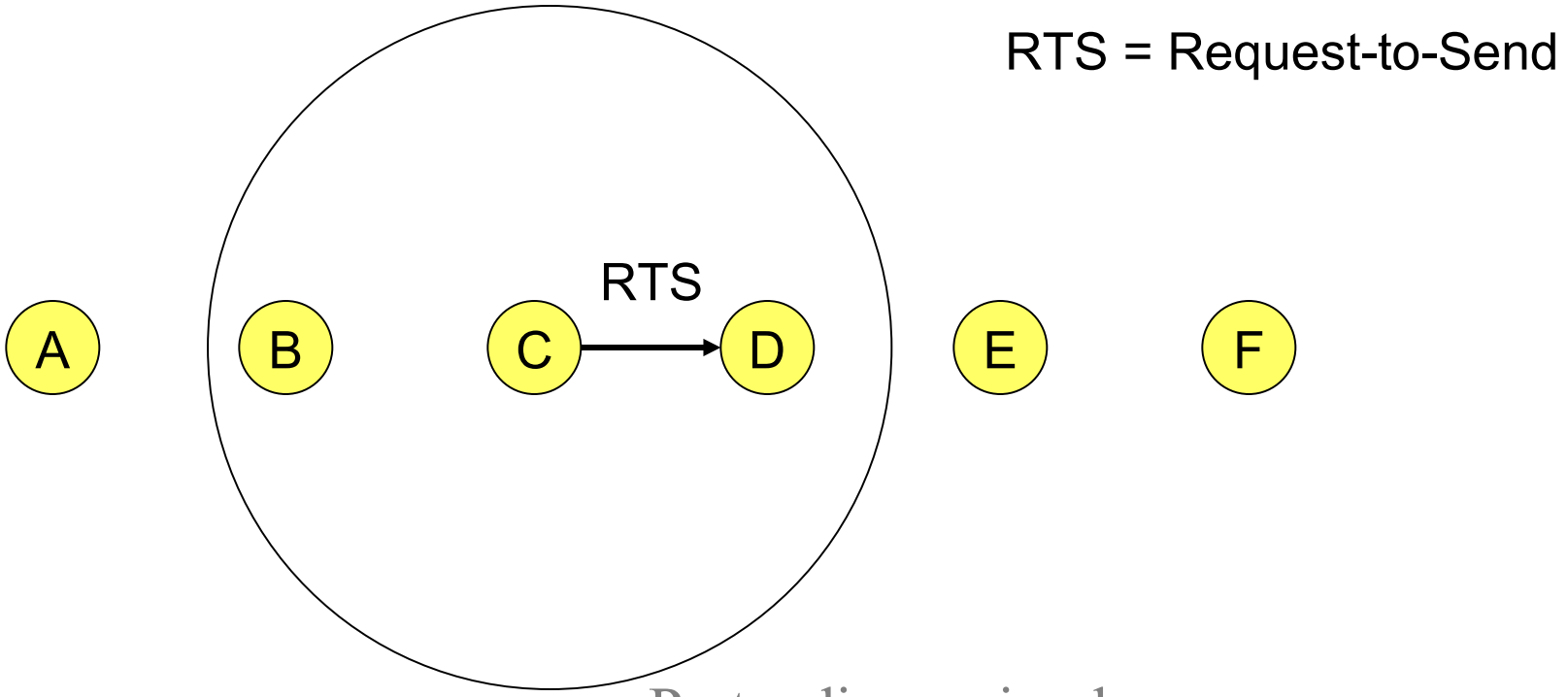
With half-duplex radios, collision detection is not possible

CSMA/CA: Wireless MAC protocols often use *collision avoidance* techniques, in conjunction with a (physical or virtual) *carrier sense* mechanism

Carrier sense: When a node wishes to transmit a packet, it first waits until the channel is idle.

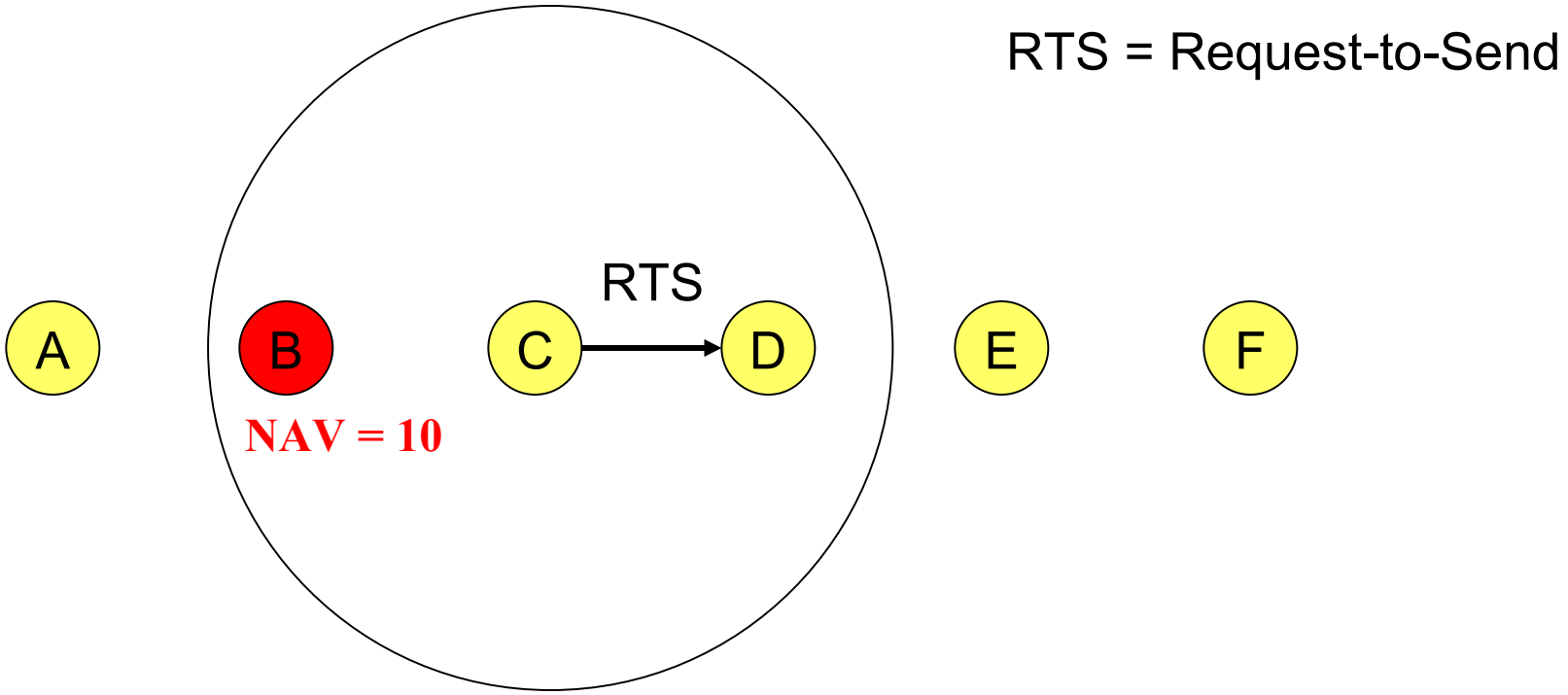
Collision avoidance: Nodes hearing RTS or CTS stay silent for the duration of the corresponding transmission. Once channel becomes idle, the node waits for a randomly chosen duration before attempting to transmit.

IEEE 802.11



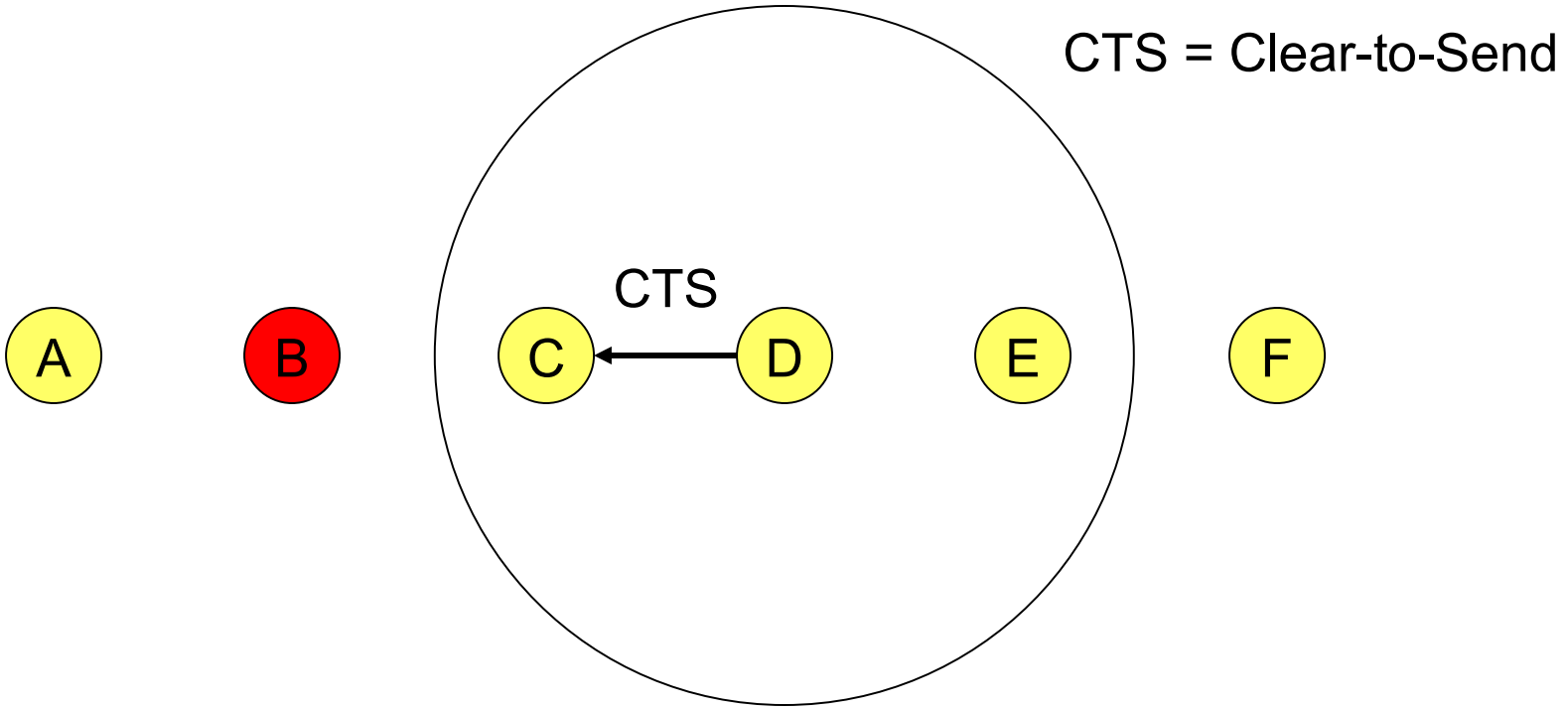
Pretending a circular range

IEEE 802.11

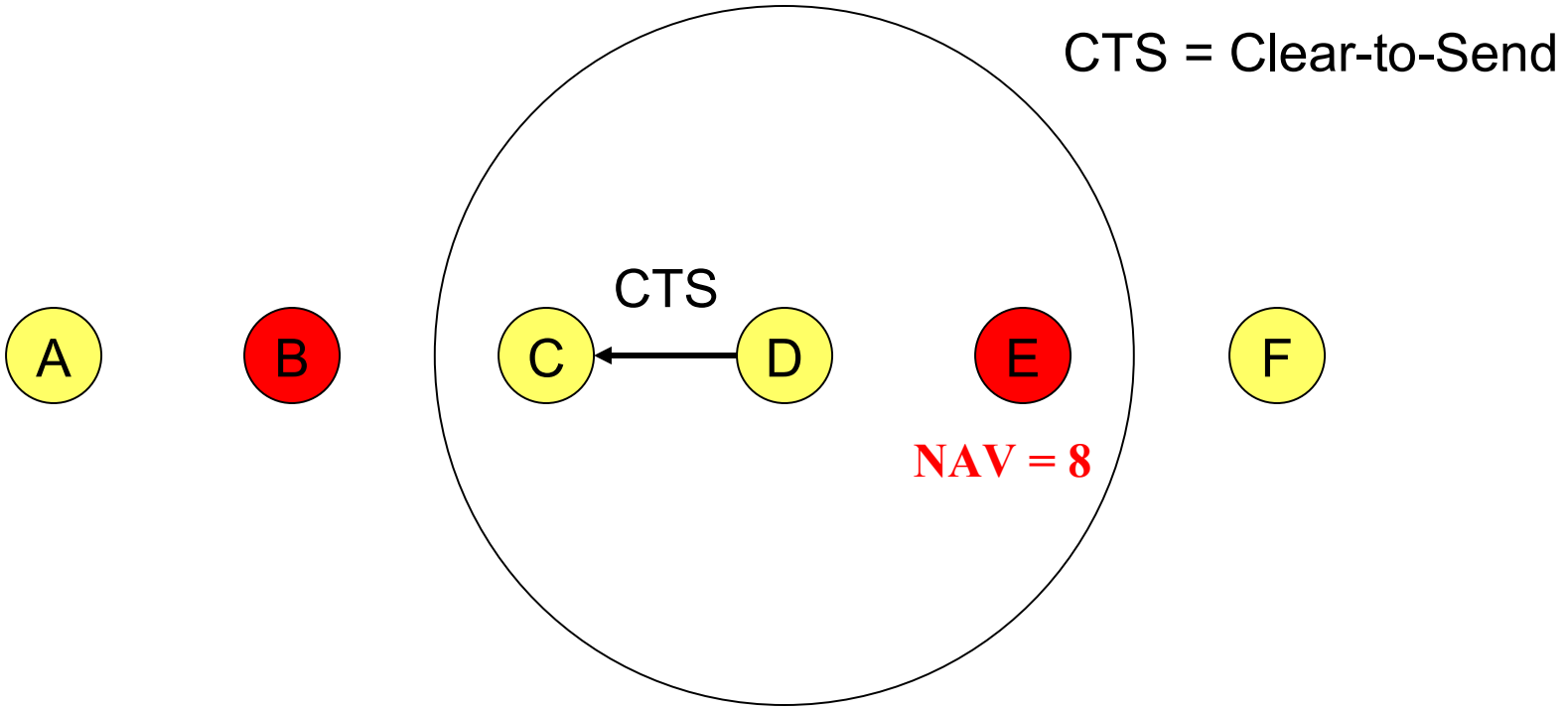


NAV = remaining duration to keep quiet

IEEE 802.11

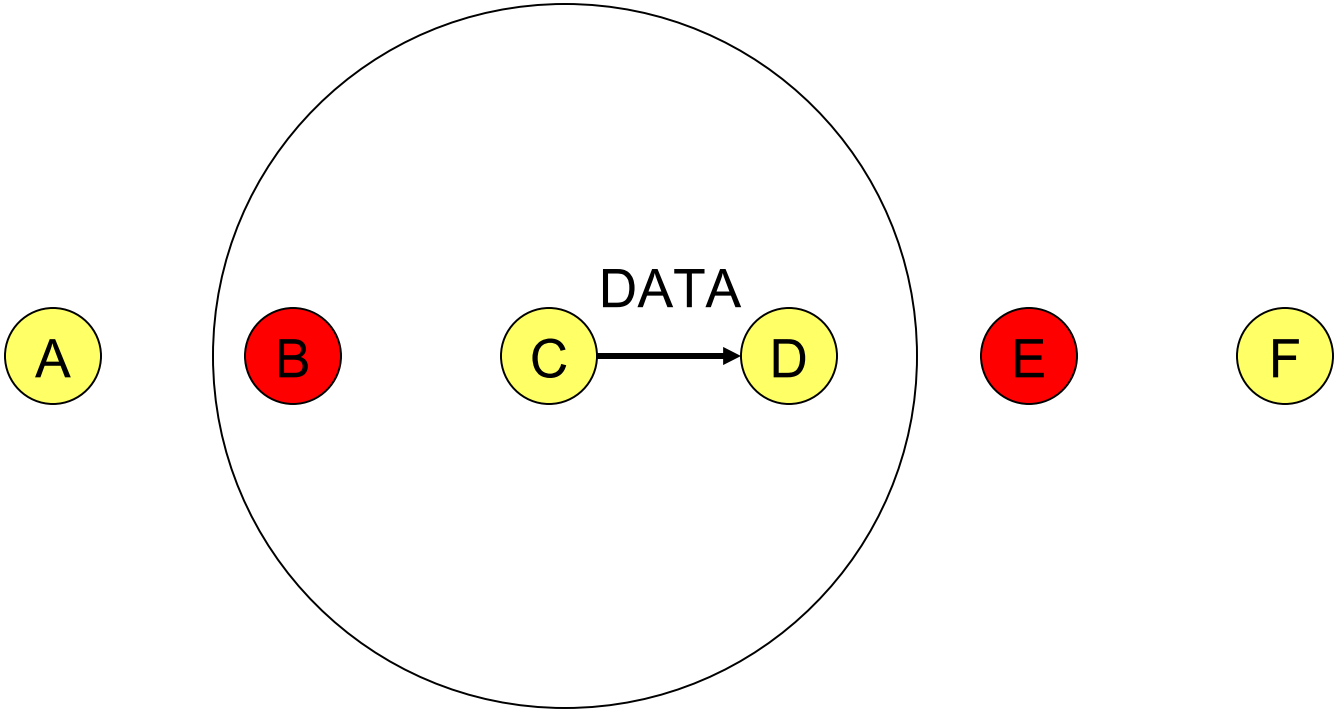


IEEE 802.11

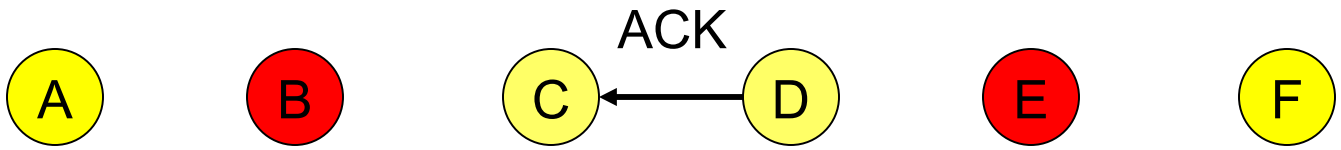


IEEE 802.11

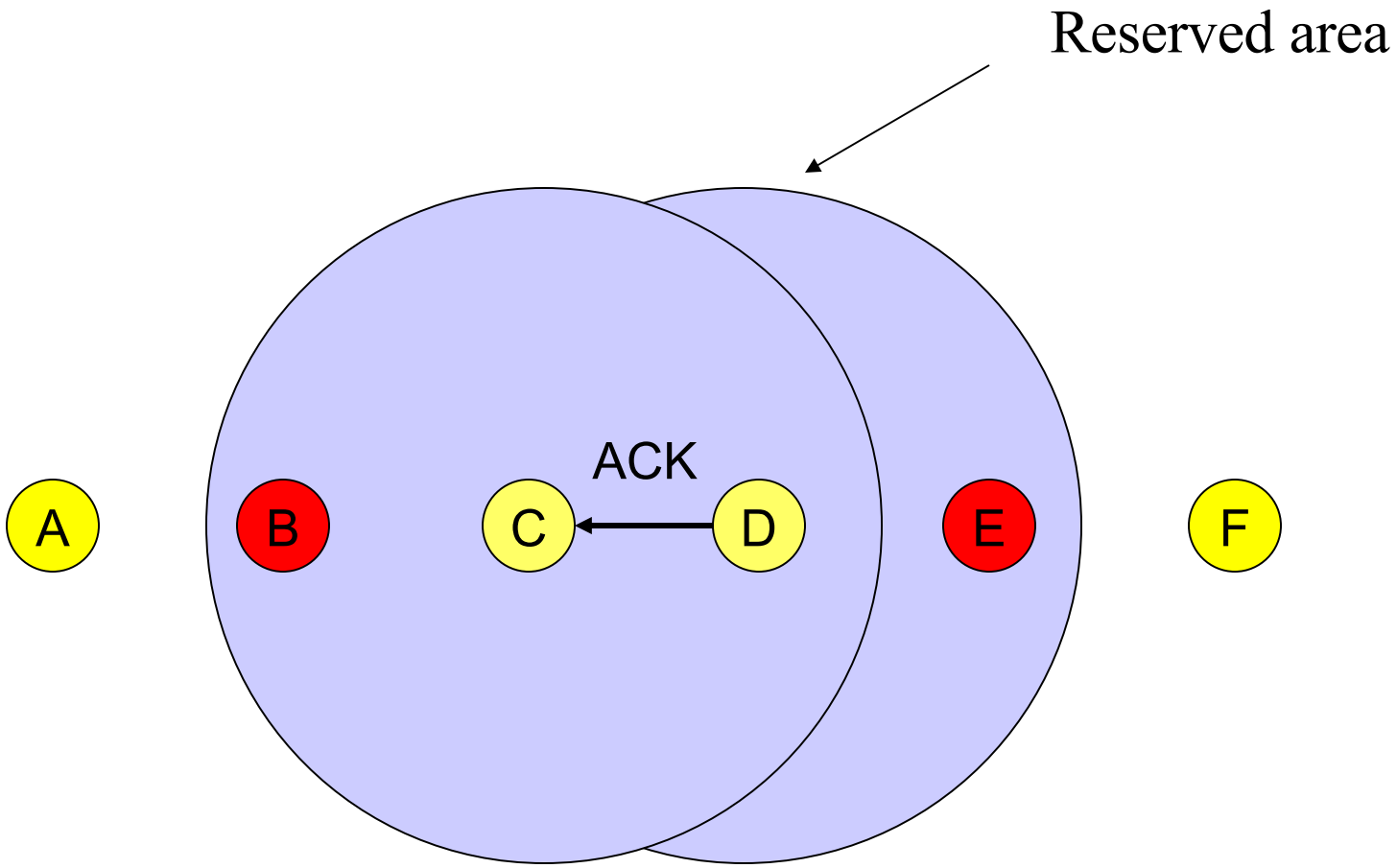
- DATA packet follows CTS. Successful data reception acknowledged using ACK.



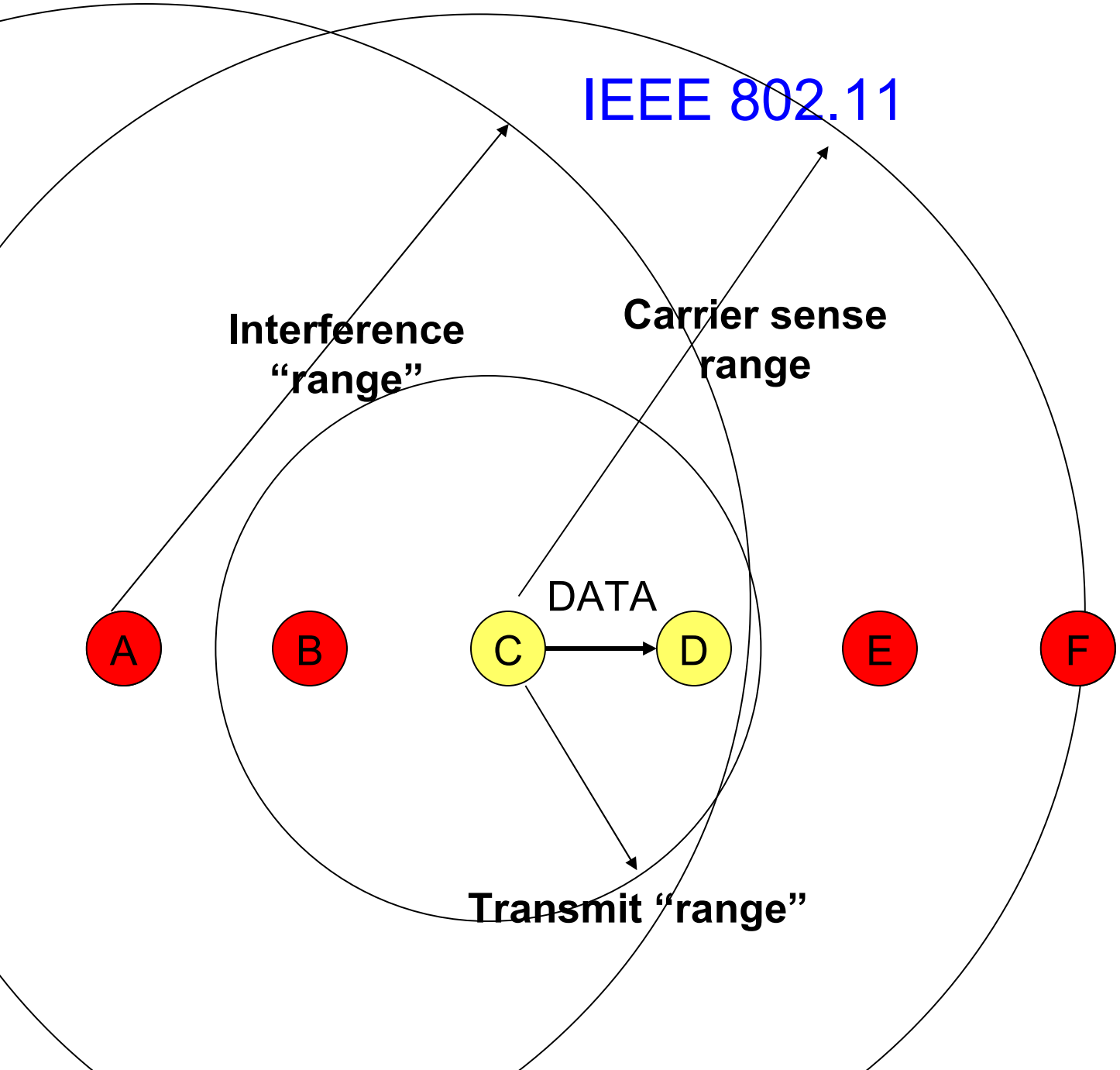
IEEE 802.11



IEEE 802.11



IEEE 802.11


Interference
“range”
Carrier sense
range
DATA
A
B
C
D
E
F
Transmit “range”

CSMA/CA

Physical carrier sense, and

Virtual carrier sense using Network Allocation Vector (NAV)

NAV is updated based on overheard RTS/CTS/DATA/ACK packets, each of which specified duration of a pending transmission

Nodes stay silent when carrier sensed (physical/virtual)

Backoff intervals used to reduce collision probability

Backoff Interval

When transmitting a packet, choose a backoff interval in the range [0,cw]

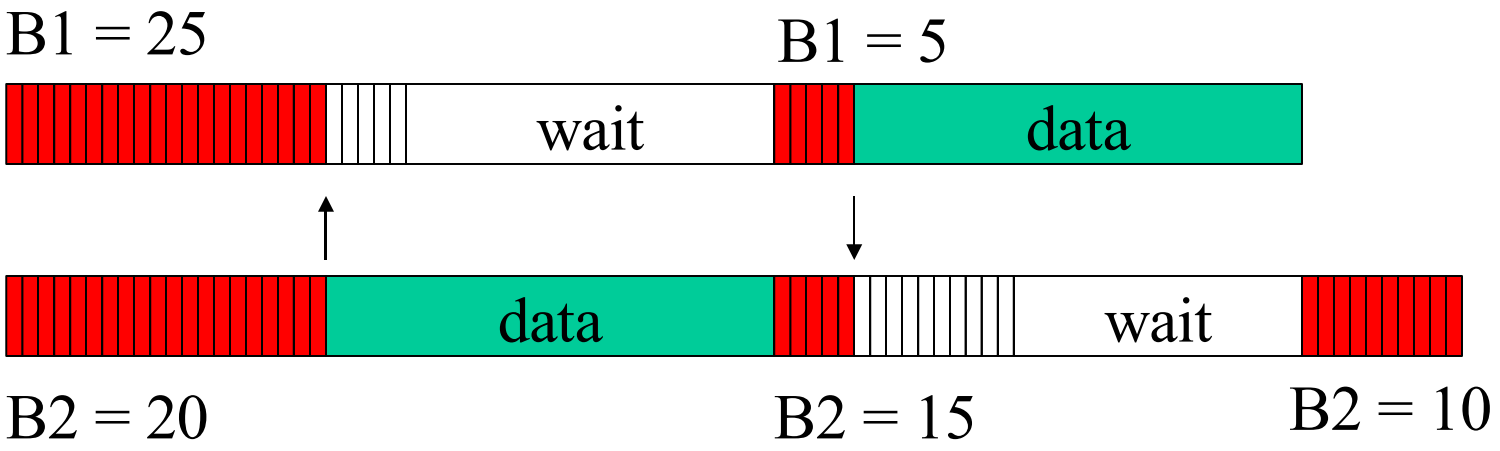
- cw is contention window

Count down the backoff interval when medium is idle

- Count-down is suspended if medium becomes busy

When backoff interval reaches 0, transmit RTS

DCF Example



cw = 31

B1 and B2 are backoff intervals at nodes 1 and 2

Backoff Interval

The time spent counting down backoff intervals is a part of MAC overhead

Choosing a *large cw* leads to large backoff intervals and can result in larger overhead

Choosing a *small cw* leads to a larger number of collisions (when two nodes count down to 0 simultaneously)

Since the number of nodes attempting to transmit simultaneously may change with time, some mechanism to manage contention is needed

IEEE 802.11 DCF: contention window *cw* is chosen dynamically depending on collision occurrence

Binary Exponential Backoff in DCF

When a node fails to receive CTS in response to its RTS, it increases the contention window

cw is doubled (up to an upper bound)

When a node successfully completes a data transfer, it restores *cw* to *CWmin*

cw follows a sawtooth curve

MILD Algorithm in MACAW [Bharghavan94]

When a node successfully completes a transfer, reduces *cw* by 1

- In 802.11 cw is restored to cw_{min}
- In 802.11, cw reduces much faster than it increases
- MACAW: cw reduces slower than it increases
  - Exponential Increase Linear Decrease

MACAW can avoid wild oscillations of cw when large number of nodes contend for the channel

Alternative Contention Resolution Mechanism [Hiperlan]

Elimination phase

- A node transmits a burst for a random number (geometrically distributed) of slots
- If medium idle at the end of the burst, go to yield phase, else give up until next round

Yield phase

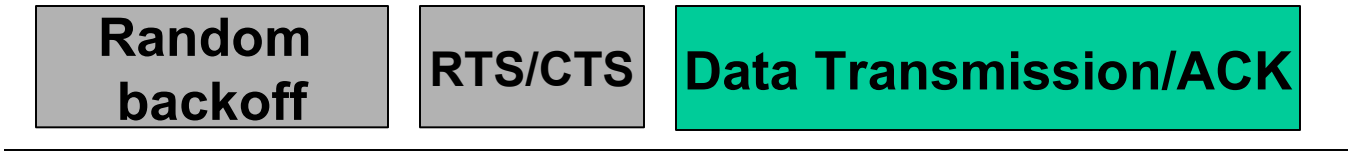
- Stay silent for a random number (geometrical distributed) of slots
- If medium still silent, transmit

Contention Resolution Overhead

Channel contention resolved using backoff

- Nodes choose random backoff interval from [0, CW]
- Count down for this interval before transmission

Backoff and (optional) RTS/CTS handshake before transmission of data packet



Inefficiency of IEEE 802.11

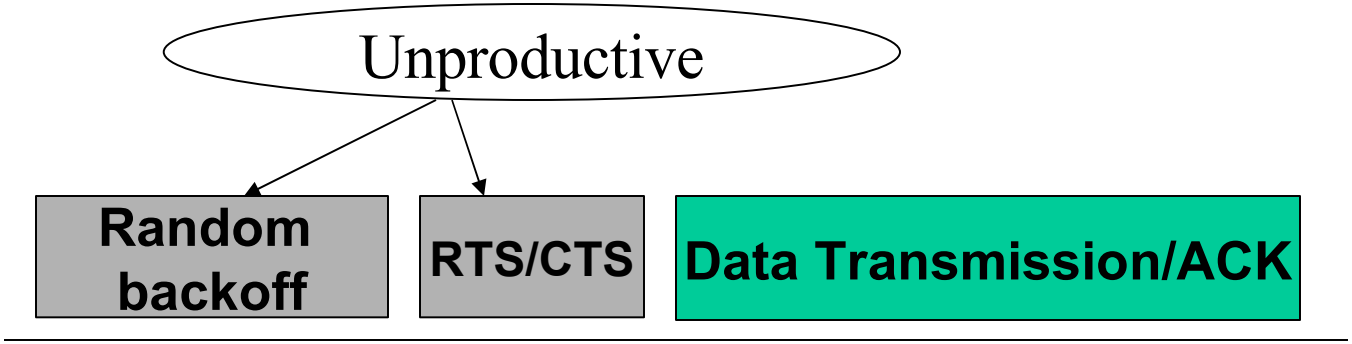
Backoff interval should be chosen appropriately for efficiency

Backoff interval with 802.11 far from optimum

Observation

Backoff and RTS/CTS handshake are unproductive:

- Do not contribute to throughput



Observation

Terry Todd observed that if a protocol has a "bandwidth-independent" overhead it is possible to improve performance by moving the bandwidth-independent overhead to a narrowband channel

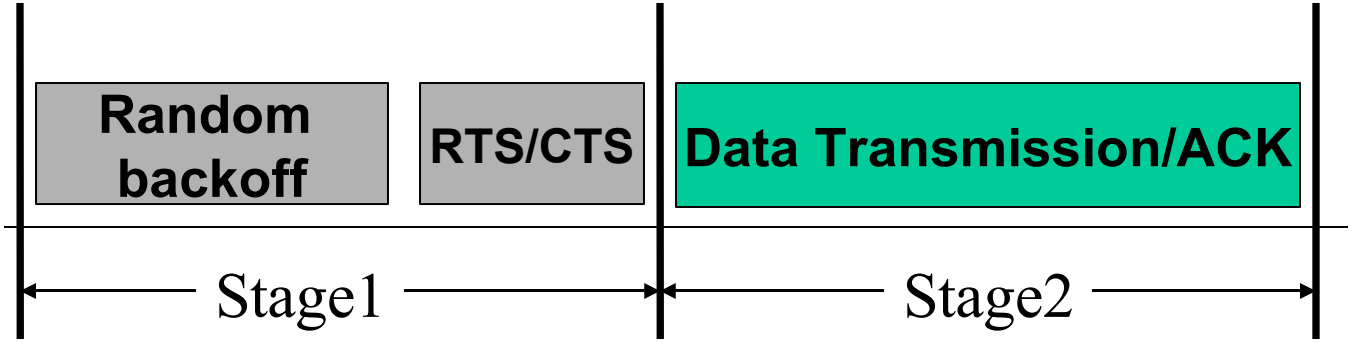
Pipelining motivated by these observations

Pipelining [Yang02techrep]

Two stage pipeline:

- Random backoff and RTS/CTS handshake
- Data transmission and ACK

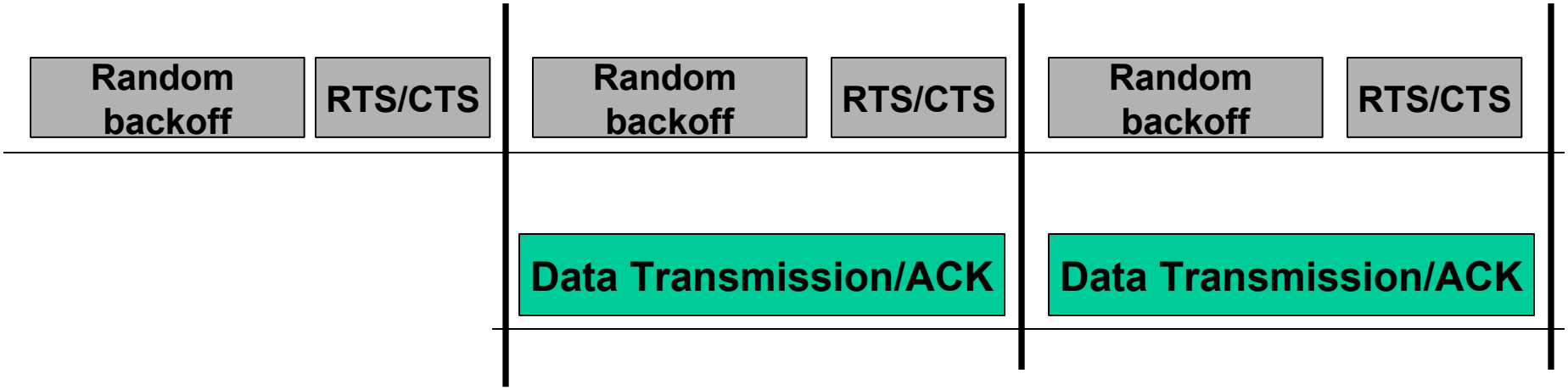
"Total" pipelining: Resolve contention completely in stage 1



How to Pipeline ?

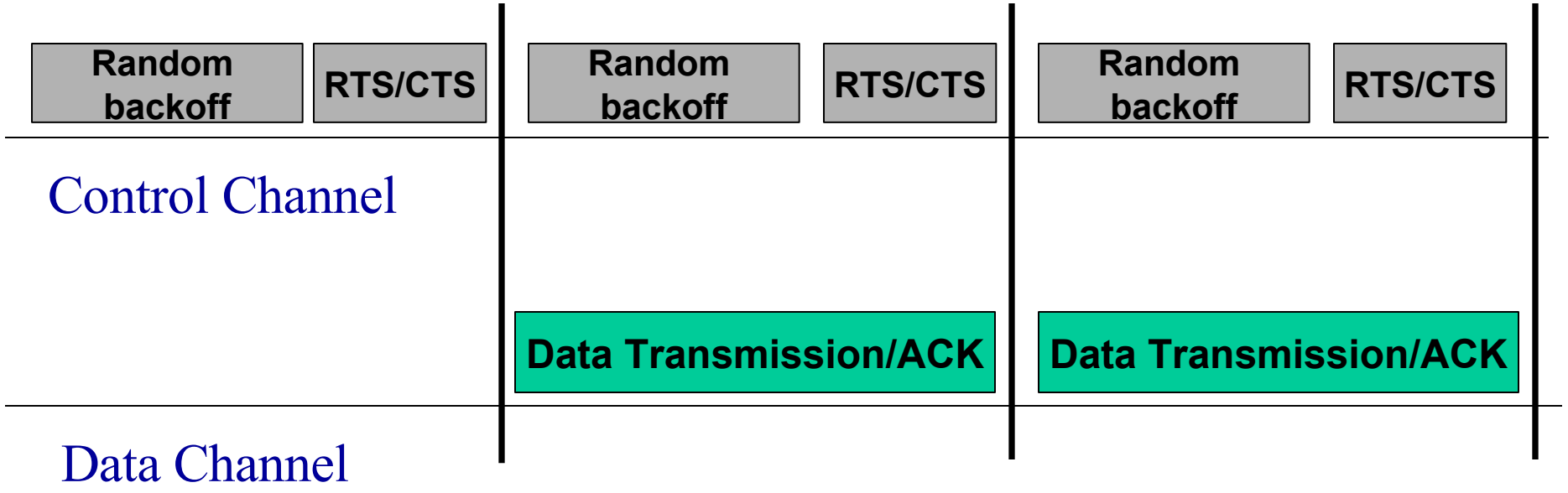
Use two channels

- Control Channel: Random backoff and RTS/CTS handshake
- Data Channel: Data transmission and ACK



Pipelining

Pipelining works well only if two stages are balanced!



Pipelining

Length of stage 1 depends on:

- Control channel bandwidth
- The random backoff duration
- The number of collisions occurred

Length of stage 2 depends on:

- Data channel bandwidth
- The data packet size

How much bandwidth does control channel require?

If small, then

- RTS/CTS takes very long time.
- Collision detection is slow

If large, then

- The portion of channel bandwidth used for productive data packet transmission is reduced

Total bandwidth is fixed!

Difficulty with Total Pipelining

The optimum division of channel bandwidth varies with contention level and data packet size

Performance with inappropriate bandwidth division could be even worse than 802.11 DCF

How to get around the issue of bandwidth division ?

Partial Pipelining

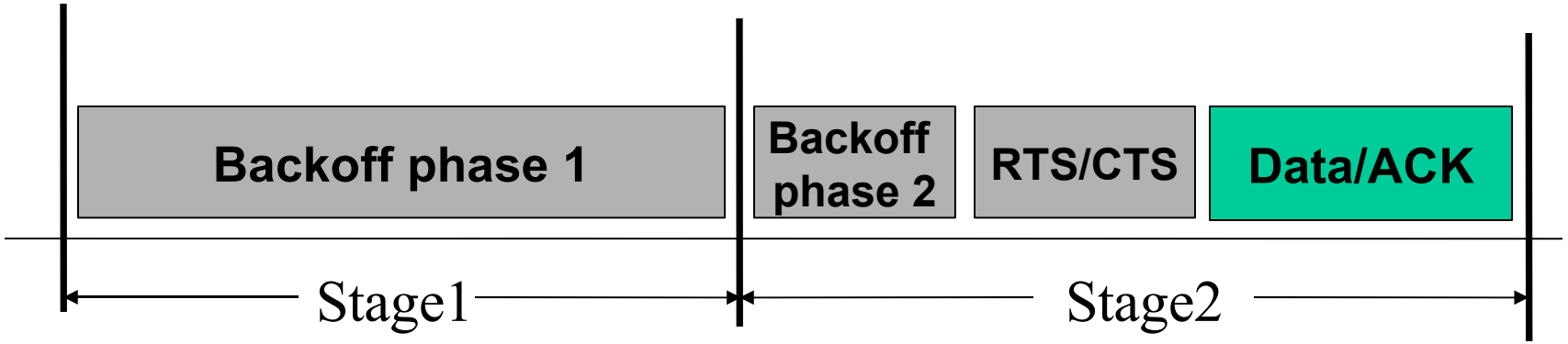
Only *partially* resolve channel contention in stage 1

Since no need to completely resolve contention, the length of stage 1 can be elastic to match the length of stage 2

Modified Two Stage Pipeline

Stage 1: Random backoff phase 1

Stage 2: Random backoff phase 2, RTS/CTS handshake and Data/ACK transmission

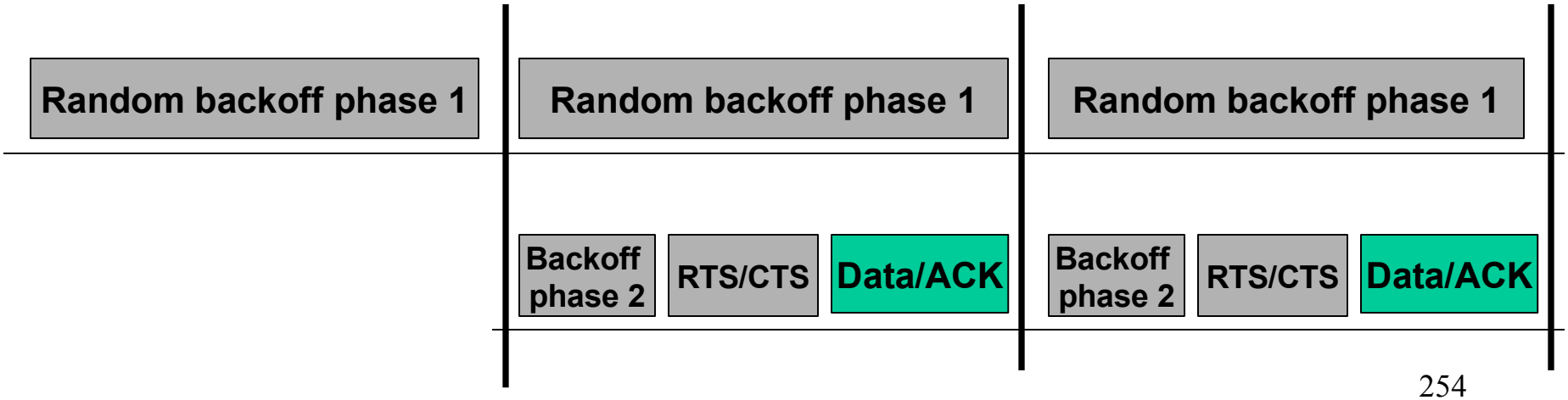


Still use two channels

Narrow Band Busy Tone Channel:

- Random backoff phase 1

Data Channel: Random backoff phase 2, RTS/CTS handshake and Data/ACK



Random Backoff Phase 1

Each Station maintains a counter for random backoff phase 1

The stations, which count to zero first, send a busy tone to claim win in stage 1

- Multiple winners are possible

Other stations know they lost on sensing a busy tone

Gain over total pipelining?

No packets transmitted on busy tone channel

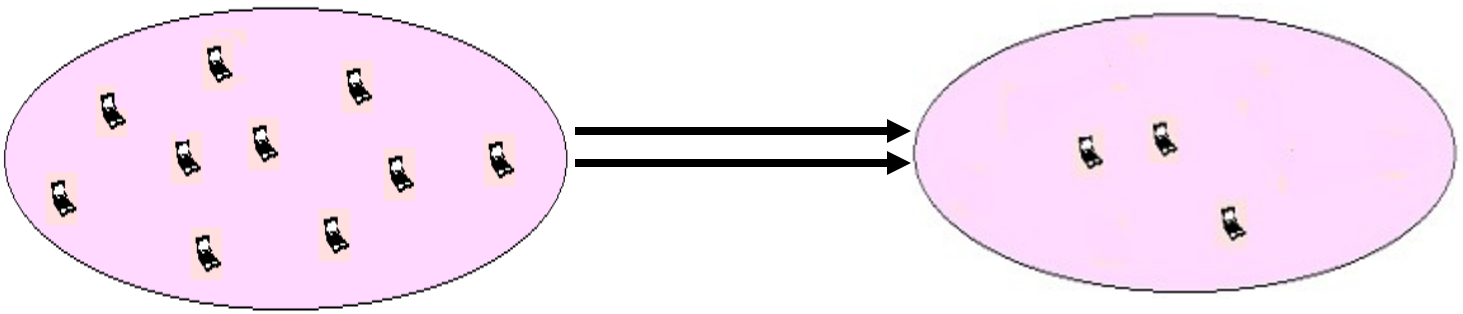
- bandwidth can be small
- the difficulty of deciding optimum bandwidth division in “total pipelining” is avoided

Length of stage 1 is elastic so the two stages can be kept balanced

Benefits of Partial Pipeline

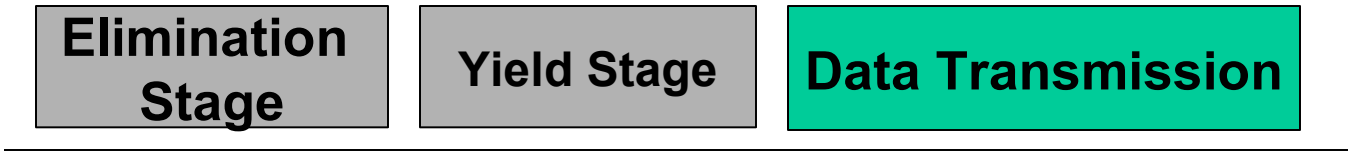
Only winners of stage 1 can contend channel in stage 2

- reduces the data channel contention
- reduces collision probability on the data channel

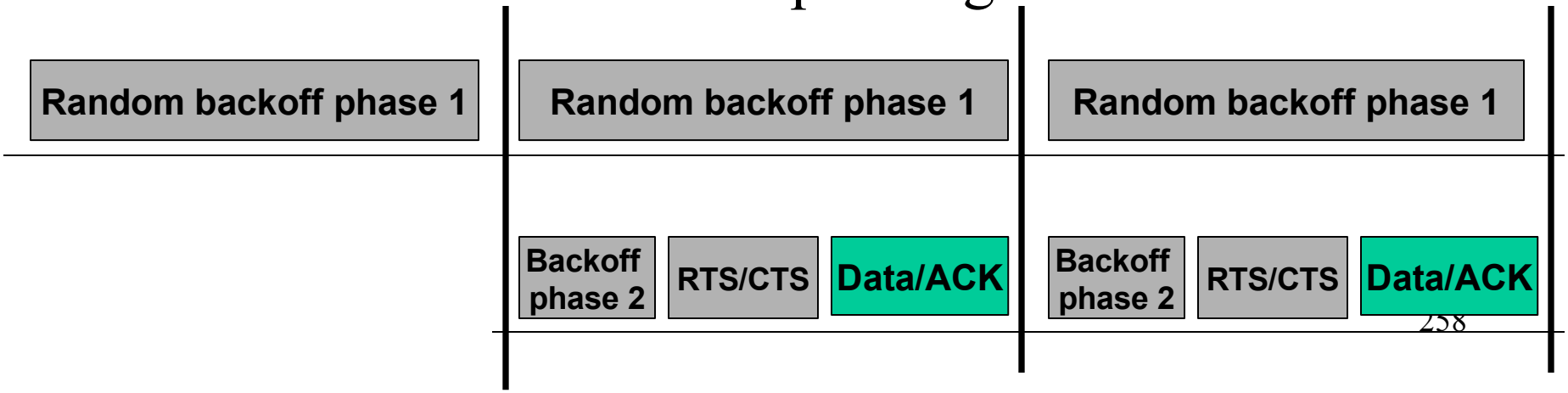


Sounds like HIPERLAN/1?

HIPERLAN / 1 (*no pipelining*)

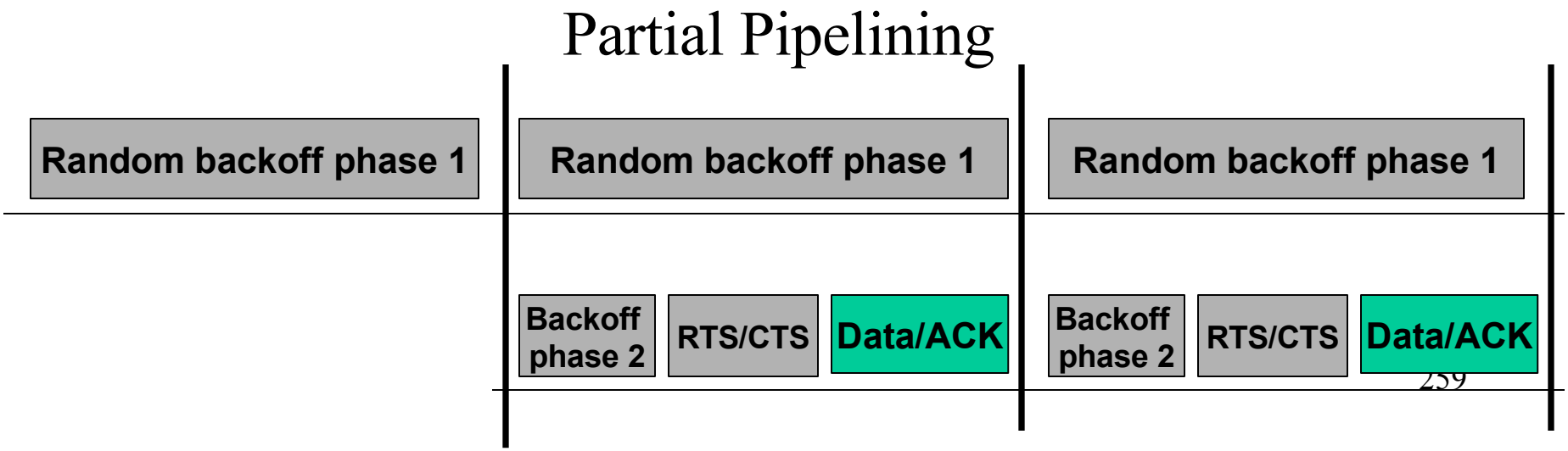


Partial Pipelining



Benefits of Partial Pipeline

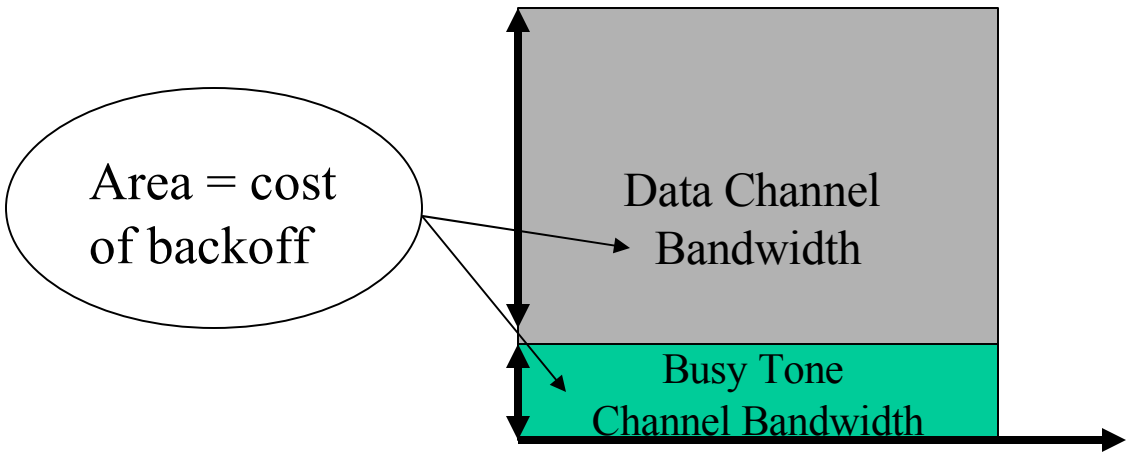
Because of pipelining, stages 1 and 2 proceed in parallel. Stage 1 costs little except for a narrow band busy tone channel



Benefits of Partial Pipeline

By migrating most of the backoff to busy tone channel,bandwidth cost of random backoff is reduced

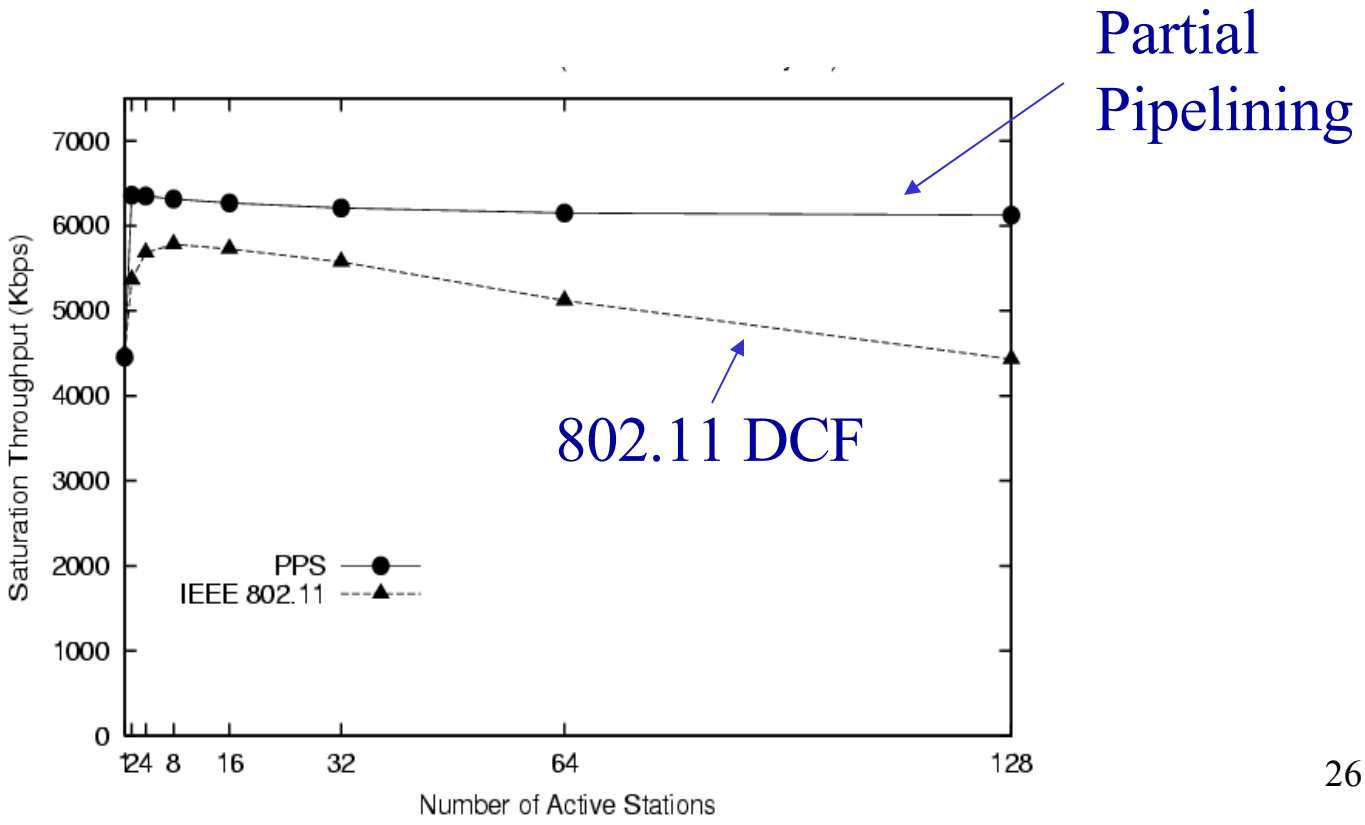
Cost of backoff = Channel bandwidth * backoff duration



Using IEEE 802.11 DSSS, the backoff duration could be several milliseconds

Results of Partial Pipelining

Improved throughput and stability over 802.11 DCF

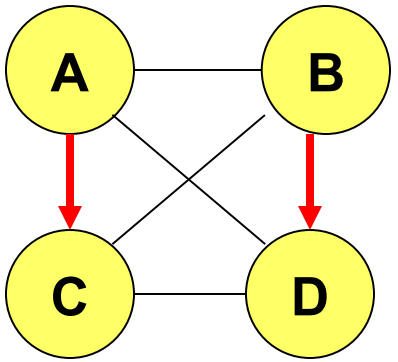


Fairness

Fairness Issue

Many definitions of fairness plausible

Simplest definition: All nodes should receive *equal* bandwidth



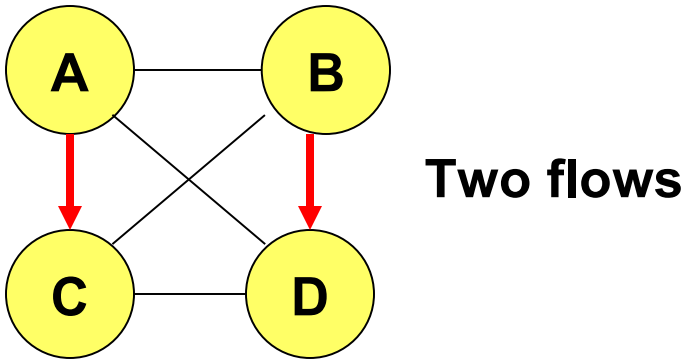
Two flows

Fairness Issue

Assume that initially, A and B both choose a backoff interval in range [0,31] but their RTSs collide

Nodes A and B then choose from range [0,63]

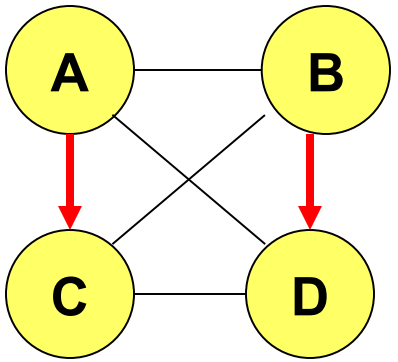
- Node A chooses 4 slots and B choose 60 slots
- After A transmits a packet, it next chooses from range [0,31]
- It is possible that A may transmit several packets before B transmits its first packet



Two flows

Fairness Issue

Unfairness occurs when one node has backed off much more than some other node



Two flows

MACAW Solution for Fairness

When a node transmits a packet, it appends the *cw* value to the packet, all nodes hearing that *cw* value use it for their future transmission attempts

Since *cw* is an indication of the level of congestion in the vicinity of a specific receiver node, MACAW proposes maintaining *cw* independently for each receiver

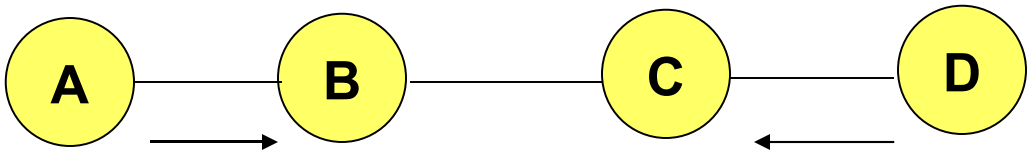
Using per-receiver *cw* is particularly useful in multi-hop environments, since congestion level at different receivers can be very different

Another MACAW Proposal

For the scenario below, when node A sends an RTS to B, while node C is receiving from D, node B cannot reply with a CTS, since B knows that D is sending to C

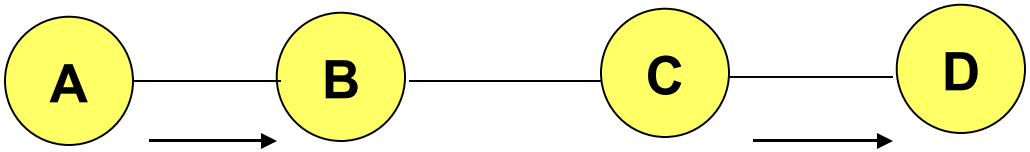
When the transfer from C to D is complete, node B can send a Request-to-send-RTS to node A [Bharghavan94Sigcomm]

Node A may then immediately send RTS to node B



This approach, however, does not work in the scenario below

Node B may not receive the RTS from A at all, due to interference with transmission from C



Weighted Fair Queueing [Keshav97book]

Assign a weight to each node

Bandwidth used by each node should be proportional to the weight assigned to the node

Distributed Fair Scheduling (DFS) [Vaidya00Mobicom]

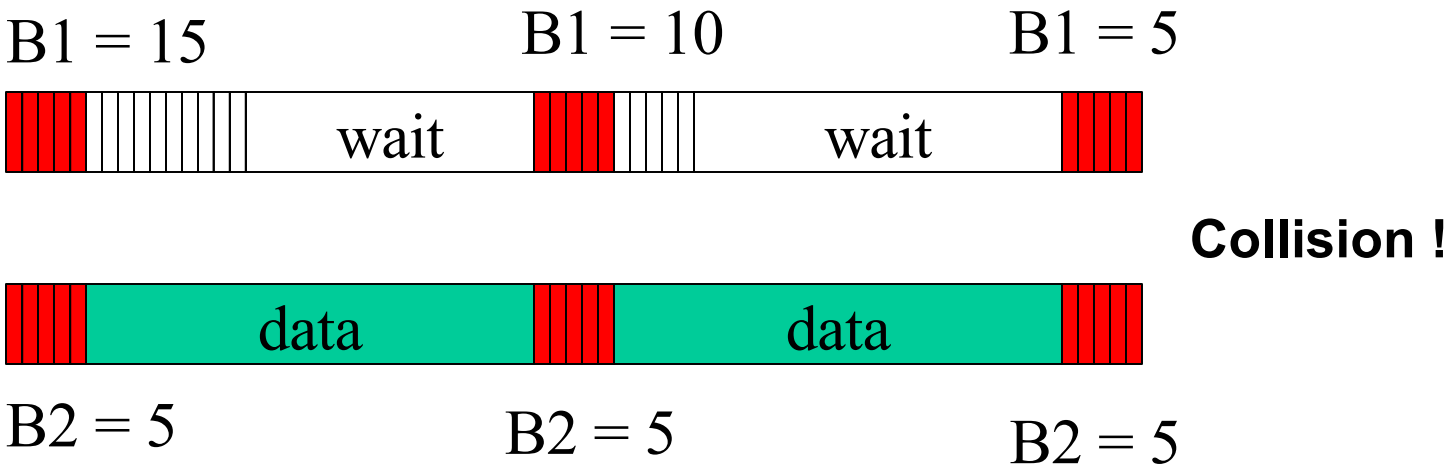
A fully distributed algorithm for achieving weighted fair queueing

Chooses backoff intervals proportional to *(packet size / weight)*

DFS attempts to mimic the centralized Self-Clocked Fair Queueing algorithm [Golestani]

Works well on a LAN

Distributed Fair Scheduling (DFS)



Weight of node 1 = 1
Weight of node 2 = 3

Assume equal packet size

B1 = 15 (DFS actually picks a random value with mean 15)

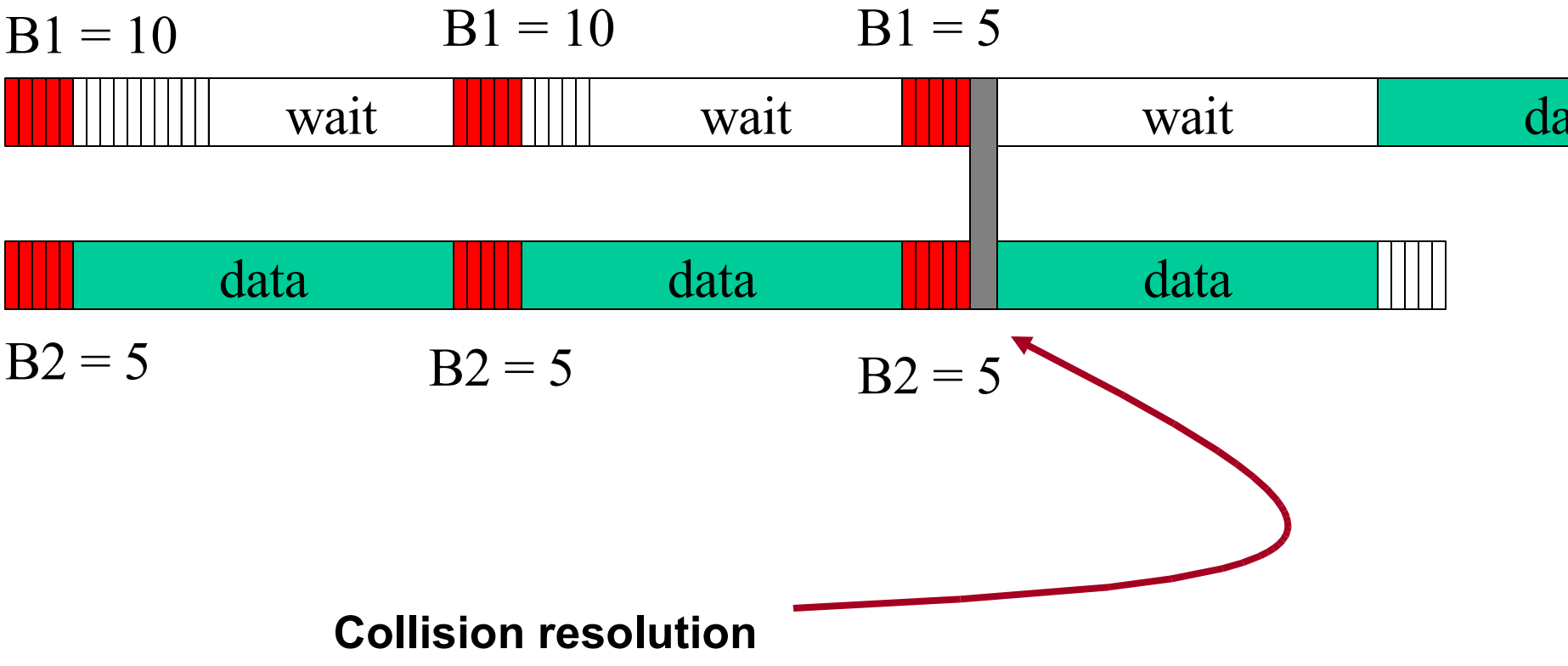
B2 = 5 (DFS picks a value with mean 5)

Impact of Collisions

After collision resolution, either node 1 or node 2 may transmit a packet

The two alternatives may have different fairness properties (since collision resolution can result in priority inversion)

Distributed Fair Scheduling (DFS)



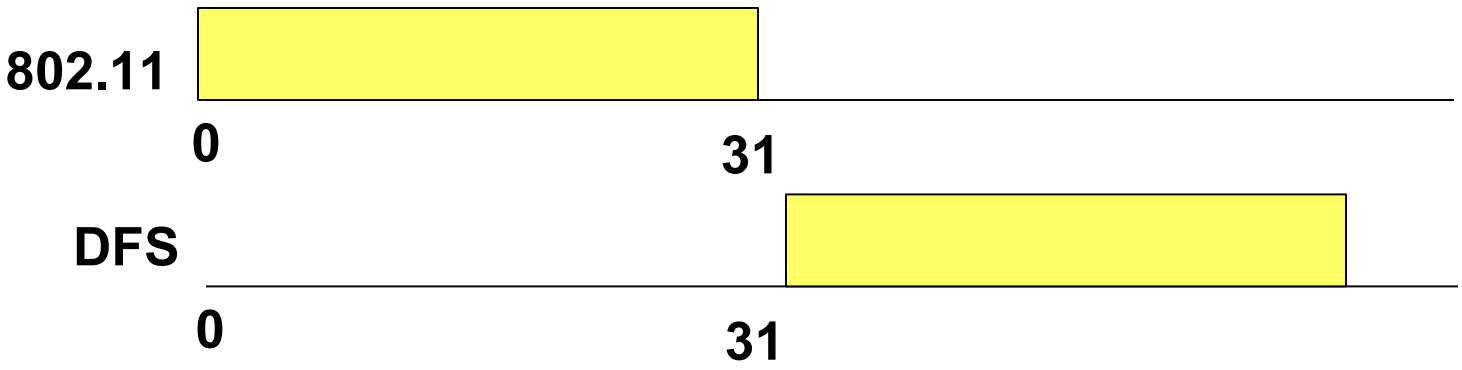
Distributed Fair Scheduling

DFS uses randomization to reduce collisions

- Alleviates negative impact of *synchronization*

DFS also uses a ***shifted*** contention window for choosing initial backoff interval

- Reduces *priority inversion* (which leads to short-term unfairness)



DFS

Due to large cw, DFS can potentially yield lower throughput than IEEE 802.11

- trade-off between fairness and throughput

On multi-hop network, properties of DFS still need to be characterized

Fairness in multi-hop case affected by hidden terminals

- May need use of a copying technique, analogous to window copying in MACAW, to share some protocol state

Fairness in Multi-Hop Networks

Several definitions of fairness
[Ozugur98,Vaidya99MSR,Luo00Mobicom, Nandagopal00Mobicom]

Hidden terminals make it difficult to achieve a desired notion of fairness

Estimation-Based Fair MAC
[Bansou00MobiHoc]

Attempts to equalize *throughput/weight* ratio for all nodes

Two parts of the algorithm

- Fair share estimation
- Window adjustment

Each node estimates how much bandwidth (W) it is able to use, and the amount of bandwidth used by each station in its vicinity

- Estimation based on overheard RTS, CTS, DATA packets

Estimation-Based Fair MAC

Fair share estimation: Node estimates how much bandwidth (W_i) it is able to use, and the amount of bandwidth (W_o) used by by all other neighbors combined

- Estimation based on overheard RTS, CTS, DATA packets

Estimation-Based Fair MAC

Define:

Ti = Wi / weight of i

To = Wo / weight assigned to the group of neighbors of i

Fairness index = Ti / To

Window adjustment:

If fairness index is too large, cw = cw * 2

Else if fairness index is too small, cw = cw / 2

Else no change to cw (contention window)

SKIP

Proportional Fair Contention Resolution (PFCR)
[Nandagopal00Mobicom]

Proportional fairness: Allocate bandwidth Ri to node i such that any other allocation Si has the following property

$\sum_i \ (Si-Ri) / Ri < 0$

Link access probability is dynamically changed depending on success/failure at transmitting a packet

- On success: Link access probability is increased by an additive factor α
- On failure: Link access probability is decreased by a multiplicative factor $(1-\beta)$

Sender-Initiated Protocols

The protocols discussed so far are sender-initiated protocols

The sender initiates a packet transfer to a receiver

Receive-Initiated Mechanism
[Talucci97,Garcia99]

In most protocols, sender initiates a transfer

Alternatively, a receiver may send a
Ready-To-Receive (RTR) message to a sender requesting it to being a packet transfer

Sender node on receiving the RTR transmits data

How does a receiver determine when to poll a sender with RTR?

- Based on history, and prediction of traffic from the sender

Energy Conservation

Energy Conservation

Since many mobile hosts are operated by batteries, MAC protocols which conserve energy are of interest

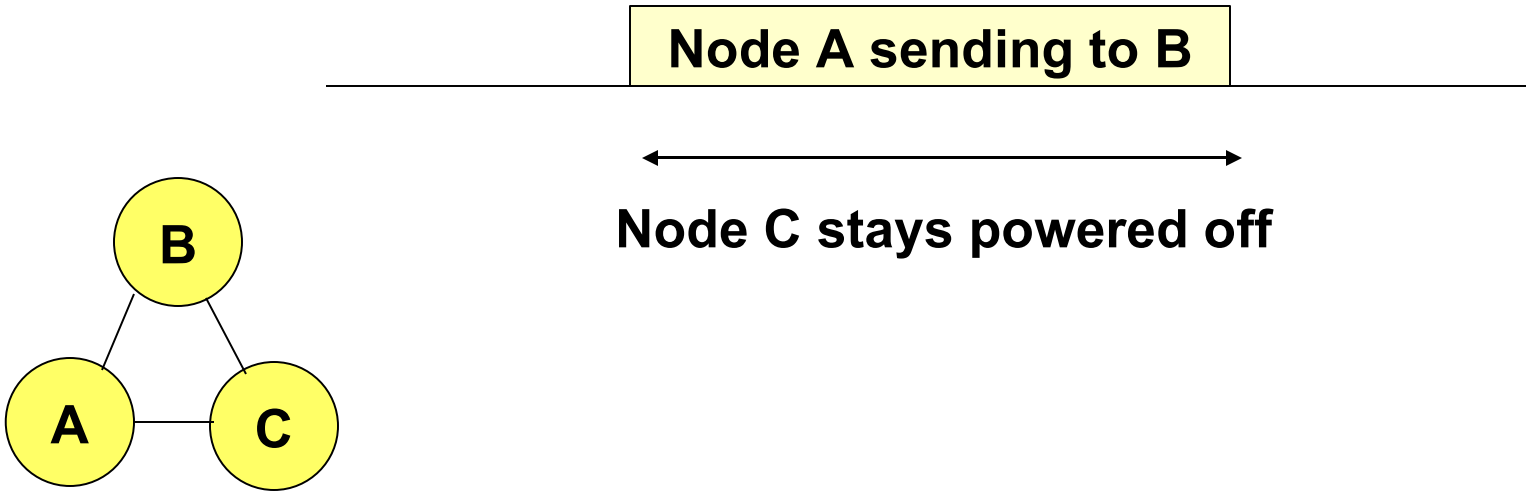
Two approaches to reduce energy consumption

- Power save: Turn off wireless interface when desirable
- Power control: Reduce transmit power

Power Aware Multi-Access Protocol (PAMAS)

[Singh98]

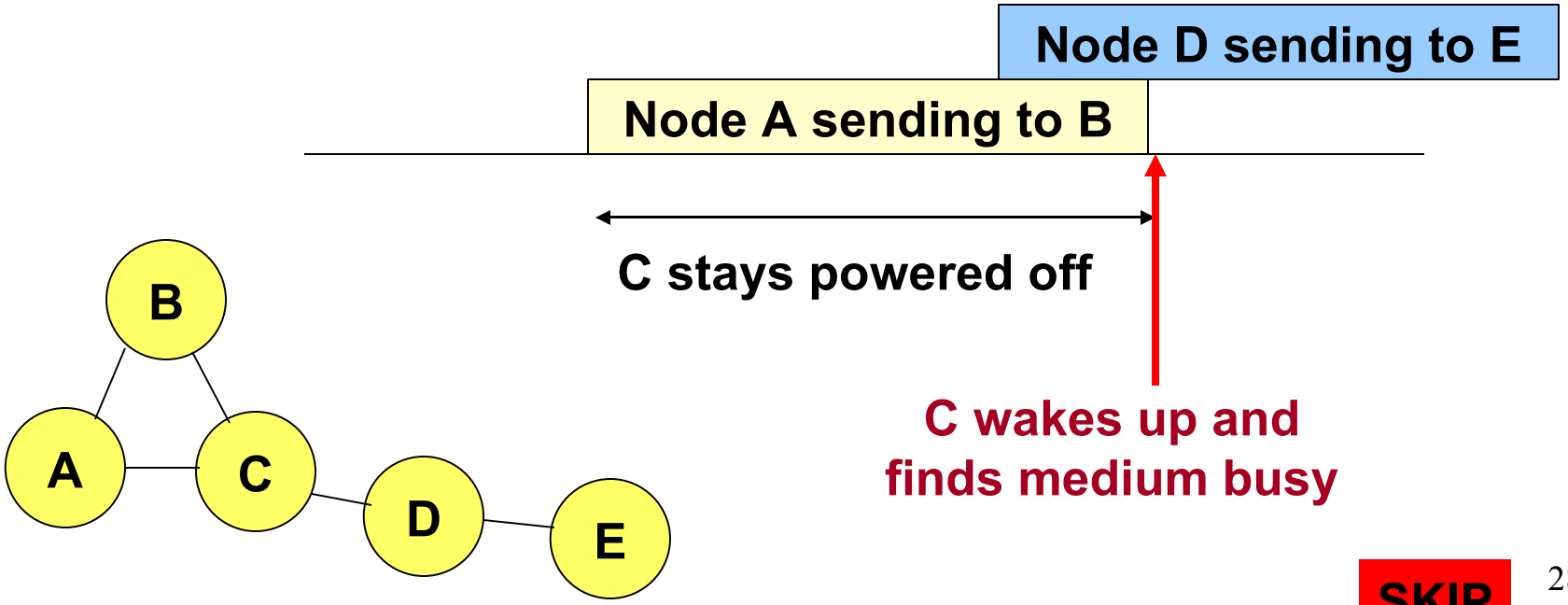
A node powers off its radio while a neighbor is transmitting to someone else



Power Aware Multi-Access Protocol (PAMAS)

What should node C do when it wakes up and finds that D is transmitting to someone else

C does not know how long the transfer will last



SKIP

PAMAS

PAMAS uses a control channel separate from the data channel

Node C on waking up performs a binary probe to determine the length of the longest remaining transfer

- C sends a probe packet with parameter L
- All nodes which will finish transfer in interval [L/2,L] respond
- Depending on whether node C see silence, collision, or a unique response it takes varying actions

Node C (using procedure above) determines the duration of time to go back to sleep

Disadvantages of PAMAS

Use of a separate control channel

Nodes have to be able to receive on the control channel while they are transmitting on the data channel

- And also transmit on data and control channels simultaneously

A node (such as C) should be able to determine when probe responses from multiple senders collide

Another Proposal in PAMAS

To avoid the probing, a node should switch off the interface for data channel, but not for the control channel (which carries RTS/CTS packets)

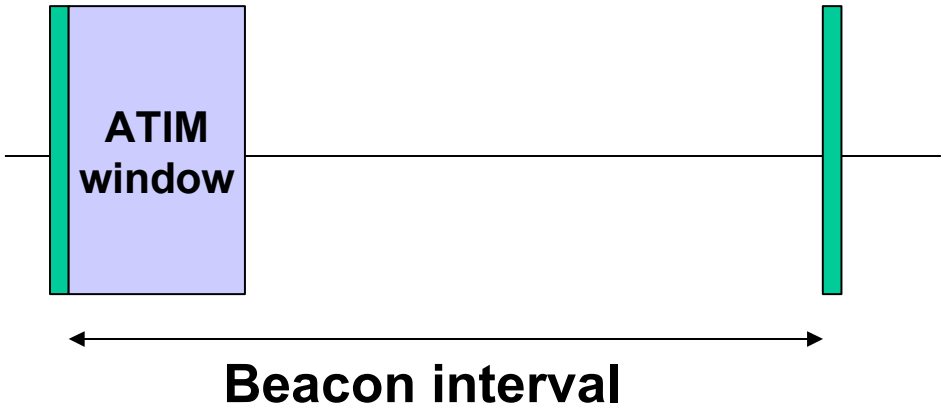
Advantage: Each sleeping node always know how long to sleep by watching the control channel

Disadvantage: This may not be useful when hardware is shared for the control and data channels

- It may not be possible turn off much hardware due to the sharing

Power Save in IEEE 802.11 Ad Hoc Mode

Time is divided into beacon intervals



Each beacon interval begins with an ATIM window

ATIM =

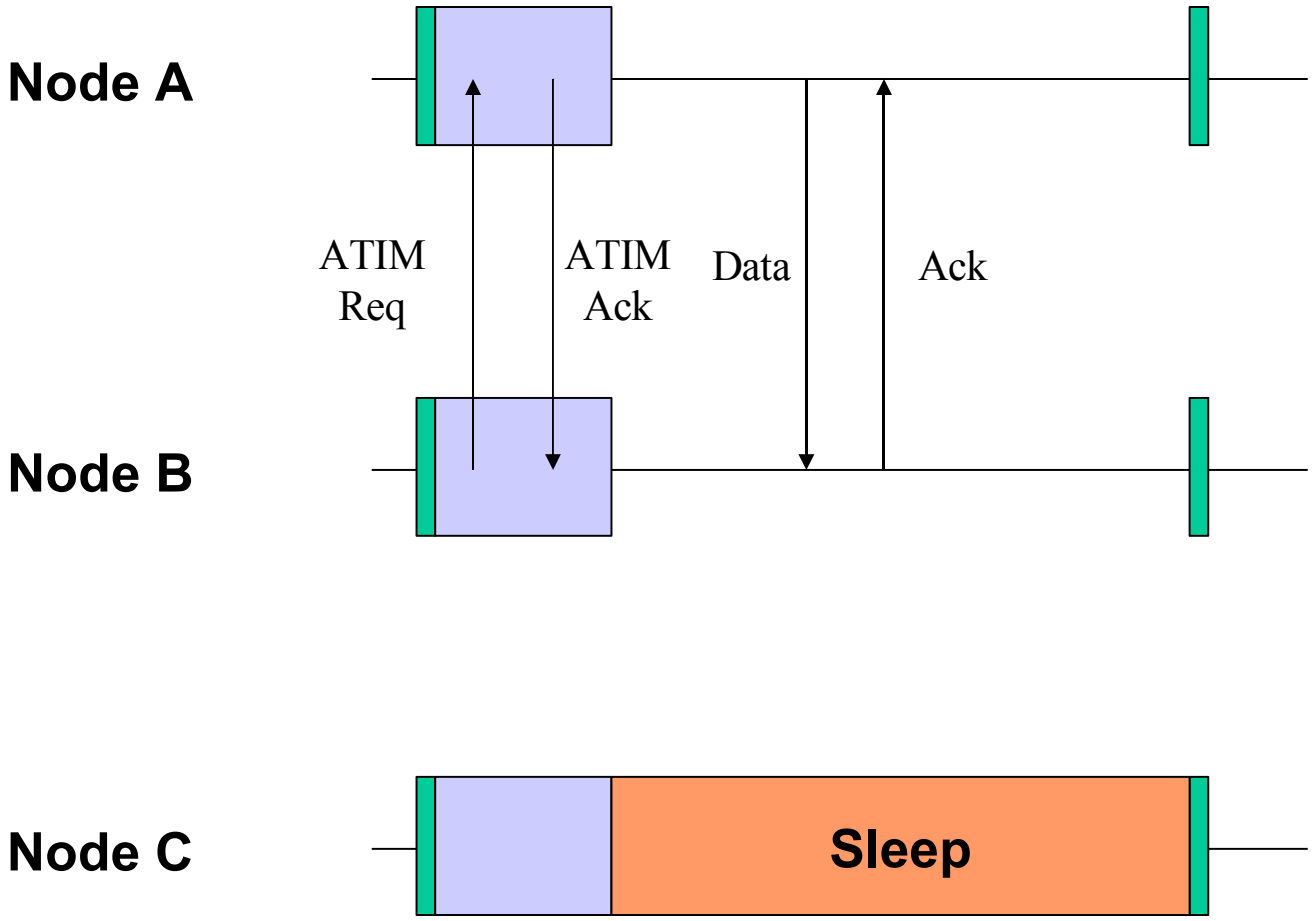
Power Save in IEEE 802.11 Ad Hoc Mode

If host A has a packet to transmit to B, A must send an ATIM Request to B during an ATIM Window

On receipt of ATIM Request from A, B will reply by sending an ATIM Ack, and stay up during the rest of the beacon interval

If a host does not receive an ATIM Request during an ATIM window, and has no pending packets to transmit, it may sleep during rest of the beacon interval

Power Save in IEEE 802.11 Ad Hoc Mode



Power Save in IEEE 802.11 Ad Hoc Mode

Size of ATIM window and beacon interval affects performance [Woesner98]

If ATIM window is too large, reduction in energy consumption reduced

- Energy consumed during ATIM window

If ATIM window is too small, not enough time to send ATIM request

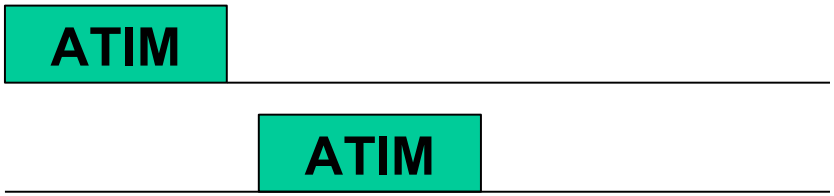
Power Save in IEEE 802.11 Ad Hoc Mode

How to choose ATIM window dynamically?

- Based on observed load [Jung02infocom]

How to synchronize hosts?

- If two hosts' ATIM windows do not overlap in time, they cannot exchange ATIM requests
- Coordination requires that each host stay awake long enough (at least periodically) to discover out-of-sync neighbors [Tseng02infocom]



Impact on Upper Layers

If each node uses the 802.11 power-save mechanism, each hop will require one beacon interval

- This delay could be intolerable

Allow upper layers to dictate whether a node should enter the power save mode or not [Chen01mobicom]

Power Save Using Wake-Up Channels

Motivation

Sleep mode power consumption << Idle power consumption

Radio State	Power Consumption (mW)
Transmit	81
Receive/Idle	30
Sleep	0.003

Power Characteristics for a Mica2 Mote Sensor

Design Alternatives

Synchronous: Once a host enters sleep mode, it wakes up at a pre-determined time

- Timer-based

Asynchronous: A sleeping host can be woken up at any time by a neighbor

Hybrid: Synchronous + Asynchronous

Using Wake-up Radio [Miller04WCNC]

Add second, low-power radio to wakeup neighbors on-demand

Low-power wake-up can be achieved using

- Simpler hardware with a lower bit-rate and/or less decoding capability, or
- A periodic duty cycle (e.g., as in STEM [UCLA]) using a "normal" radio
  - Latter approach used in the illustration here

Actions of a Sleeping Host

Periodically listen to a wake-up channel

- Duty cycle affects energy consumption

If wake-up channel sensed busy:

Turn on data radio

Receive a “filter” packet on data radio

If filter intended for another host, go back to sleep

Actions of a Sender Host

Transmit a wake-up signal “long enough” if the intended receiver is expected to be sleeping

Transmit a filter packet specifying intended receiver

Transmit data to the receiver

Purely Asynchronous Mechanism

In a purely asynchronous approach, each packet burse is preceded by a “wake-up” signal

Might wake-up too many hosts near the transmitter – referred as “full” wakeup

➔ *Energy cost*

Add a Synchronous Component

Each sleeping host will wake-up after a pre-defined interval of time ("timeout")

- Referred as "triggered" wakeup

If a transmitter cannot wait until then, it may send a wake-up signal

- Send wake-up signal if queue size exceeds threshold L or a delay bound

Timeout is computed based on recent traffic rate

Timeout for Triggered Wakeups

If too small, host may wake-up when there are no packets pending for it

If too large, too many “full” wakeups

Power Save Protocol [Miller04WCNC]

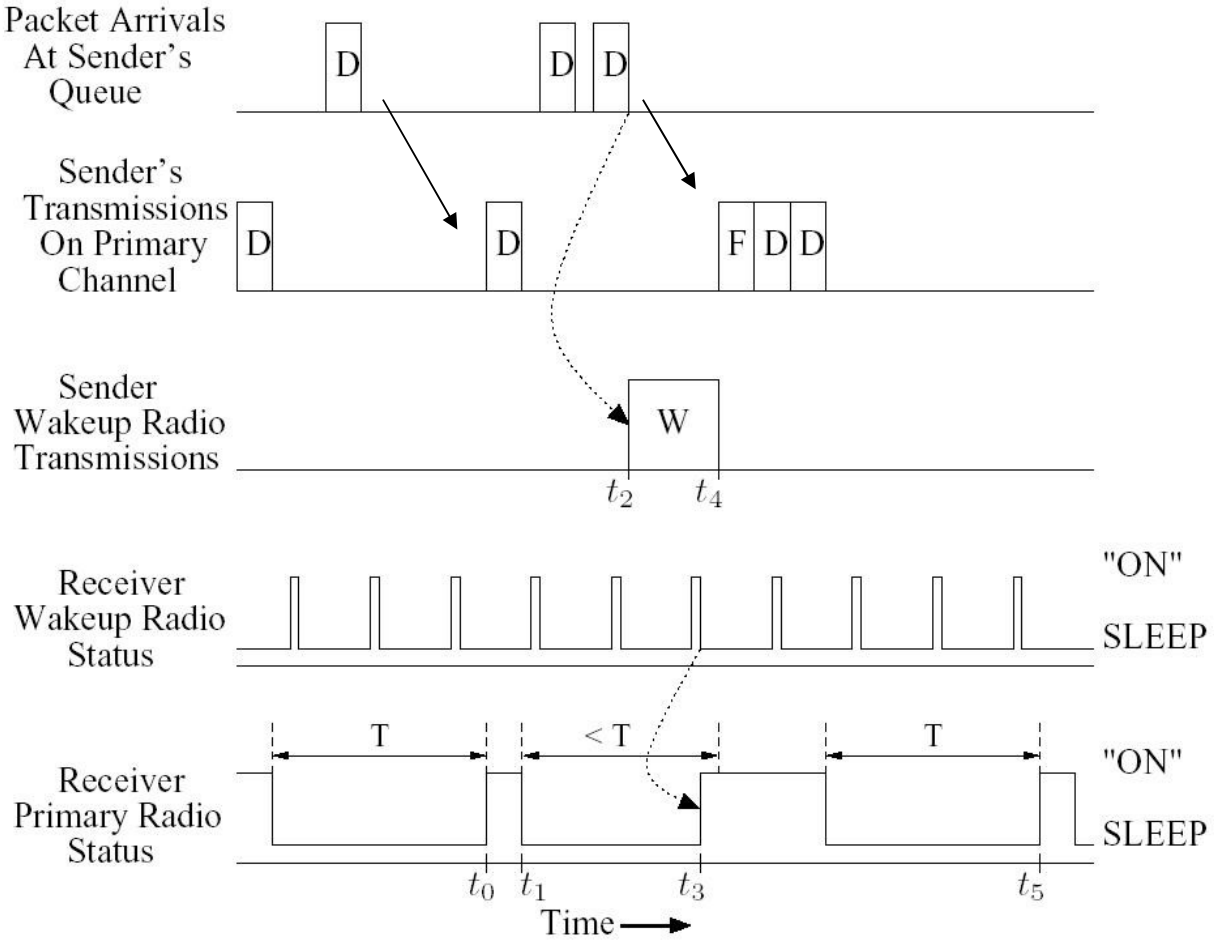


Fig. 2. Static T and $L = 2$ (**D** = data packet, **F** = filter packet, **W** = wakeup signal).

Energy Conservation

Power save

Power control

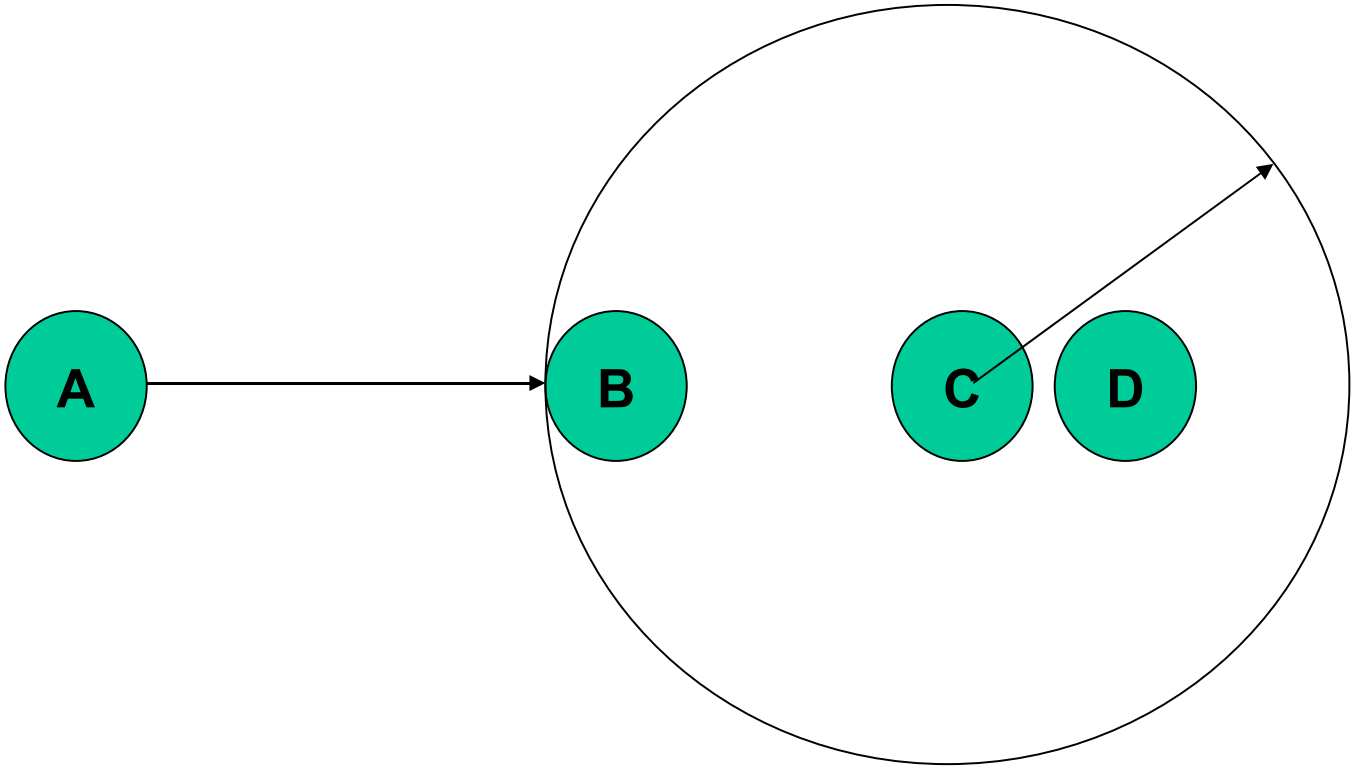
Power Control

Power control has two potential benefit

- Reduced interference & increased spatial reuse
- Energy saving

Power Control

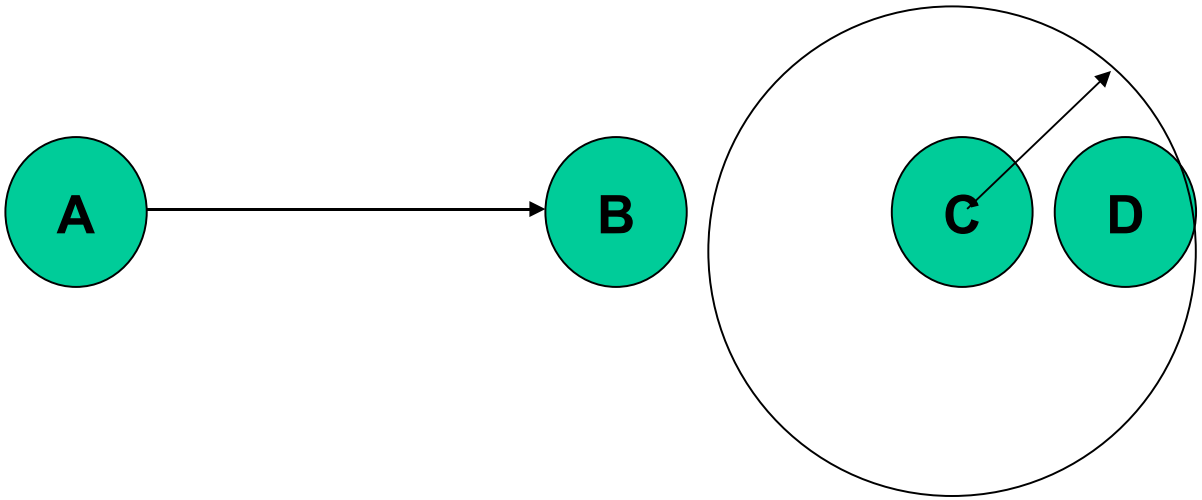
When C transmits to D at a high power level, B cannot receive A's transmission due to interference from C



Power Control

If C reduces transmit power, it can still communicate with D

- Reduces energy consumption at node C
- Allows B to receive A's transmission (spatial reuse)



Power Control

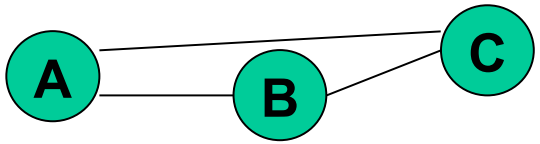
Received power level is proportional to $1/d^{\alpha}$, $\alpha >= 2$

If power control is utilized, energy required to transmit to a host at distance d is proportional to

d^{α} + constant

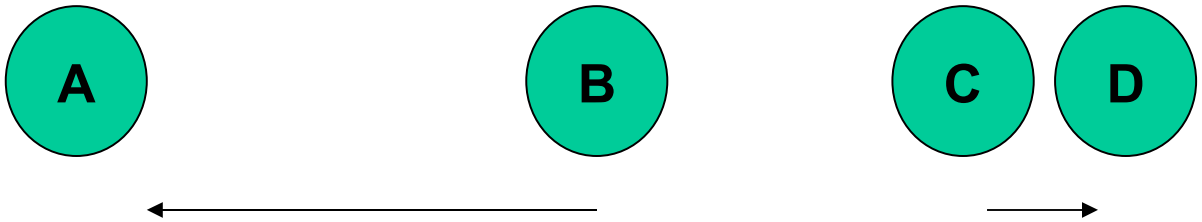
Shorter hops typically preferred for energy consumption (depending on the constant) [Rodoplu99]

Transmit to C from A via B, instead of directly from A to C



Power Control with 802.11

Transmit RTS/CTS/DATA/ACK at least power level needed to communicate with the received

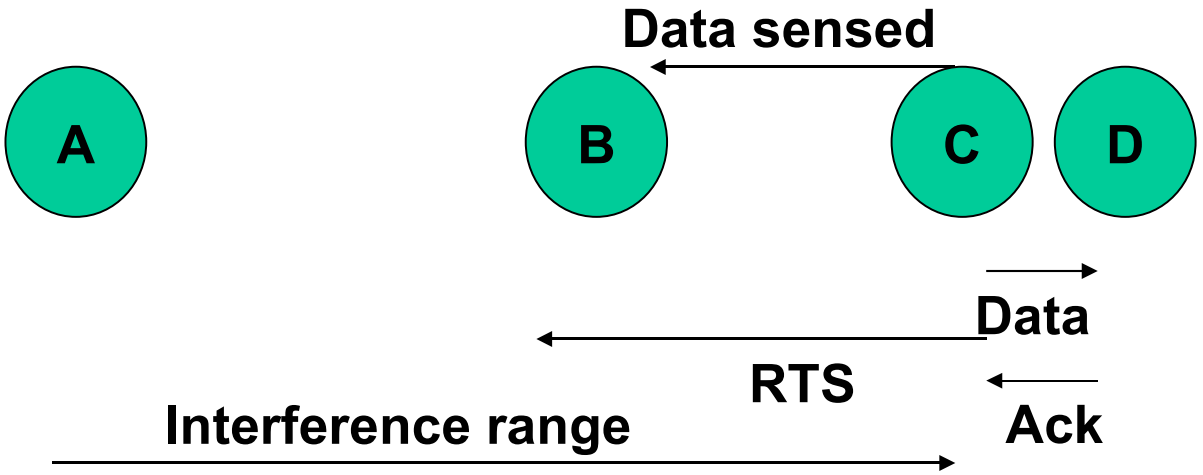


A/B do not receive RTS/CTS from C/D. Also do not sense D's data transmission

B's transmission to A at high power interferes with reception of ACK at C

A Plausible Solution

RTS/CTS at highest power, and DATA/ACK at smallest necessary power level



A cannot sense C's data transmission, and may transmit DATA to some other host

This DATA will interfere at C

This situation unlikely if DATA transmitted at highest power level

Interference range ~ sensing range

Transmitting RTS at the highest power level also reduces spatial reuse

Nodes receiving RTS/CTS have to defer transmissions

Caveat

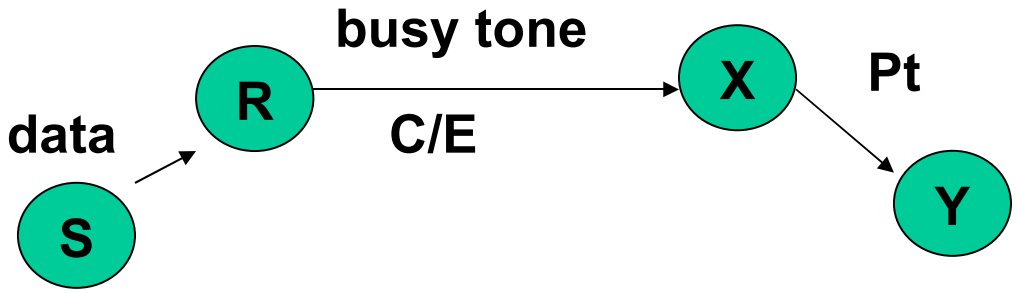
Energy saving by power control is limited to savings in transmit energy

Other energy costs may not change, and may represent a significant fraction of total energy consumption

Power Controlled Multiple Access (PCMA)
[Monks01infocom]

If receiver node R can tolerate interference E, it sends a busy tone at power level C/E, where C is an appropriate constant

When some node X receives a busy-tone a power level Pr, it may transmit at power level Pt <= C/Pr



Power Controlled Multiple Access (PCMA)

[Monks01infocom]

If receiver node R can tolerate noise E, it sends a busy tone at power level C/E, where C is an appropriate constant

When some node X receives a busy-tone a power level Pr, it may transmit at power level Pt <= C/Pr

Explanation:

- Gain of channel RX = gain of channel XR = g
- Busy tone signal level at X = Pr = g * C / E
- Node X may transmit at level = Pt = C/Pr = E/g
- Interference received by R = Pt * g = E

PCMA

Advantage

- Allows higher spatial reuse, as well as power saving using power control

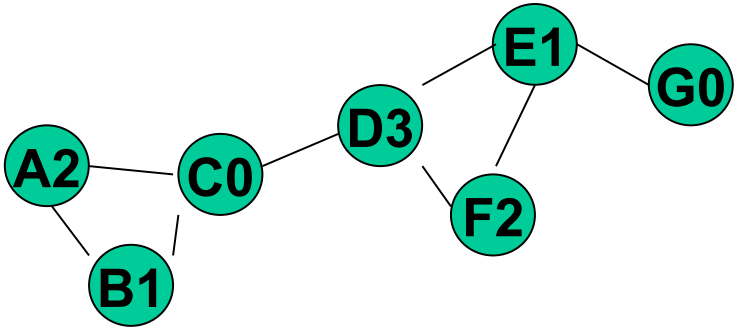
Disadvantages:

- Need a separate channel for the busy tone
- Since multiple nodes may transmit the busy tones simultaneously, spatial reuse is less than optimal

Small Addresses Save Energy
[Schurgers01mobihoc]

In sensor networks, packet sizes are small, and MAC addresses may be a substantial fraction of the packet

Observation: MAC addresses need only be unique within two hops



Fewer addresses are sufficient: Address size can be smaller. [Schurgers00mobihoc] uses Huffman coding to assign variable size encoding to the addresses

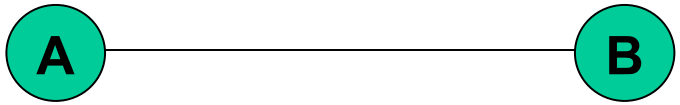
Energy consumption reduced due to smaller addresses

Adaptive Modulation

Adaptive Modulation

Channel conditions are time-varying

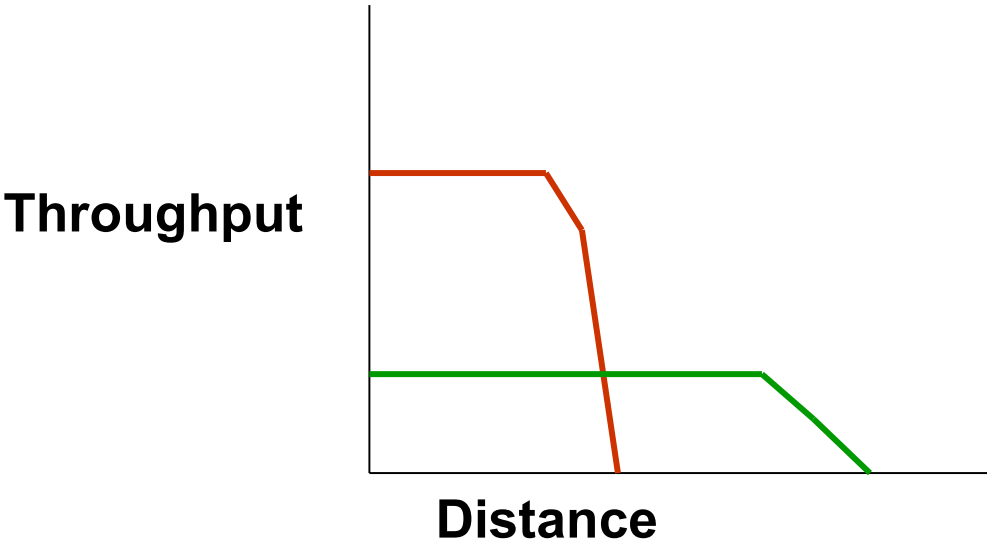
Received signal-to-noise ratio changes with time



Adaptive Modulation

Multi-rate radios are capable of transmitting at several rates, using different modulation schemes

Choose modulation scheme as a function of channel conditions



Modulation schemes provide a trade-off between throughput and range

Adaptive Modulation

If physical layer chooses the modulation scheme transparent to MAC

- MAC cannot know the time duration required for the transfer

Must involve MAC protocol in deciding the modulation scheme

- Some implementations use a sender-based scheme for this purpose [Kamerman97]
- Receiver-based schemes can perform better

Sender-Based “Autorate Fallback” [Kamerman97]

Probing mechanisms

Sender decreases bit rate after X consecutive transmission attempts fail

Sender increases bit rate after Y consecutive transmission attempt succeed

Autorate Fallback

Advantage

Can be implemented at the sender, without making any changes to the 802.11 standard specification

Disadvantage

Probing mechanism does not accurately detect channel state

Channel state detected more accurately at the receiver

Performance can suffer

- Since the sender will periodically try to send at a rate higher than optimal
- Also, when channel conditions improve, the rate is not increased immediately

Receiver-Based Autorate MAC
[Holland01mobicom]

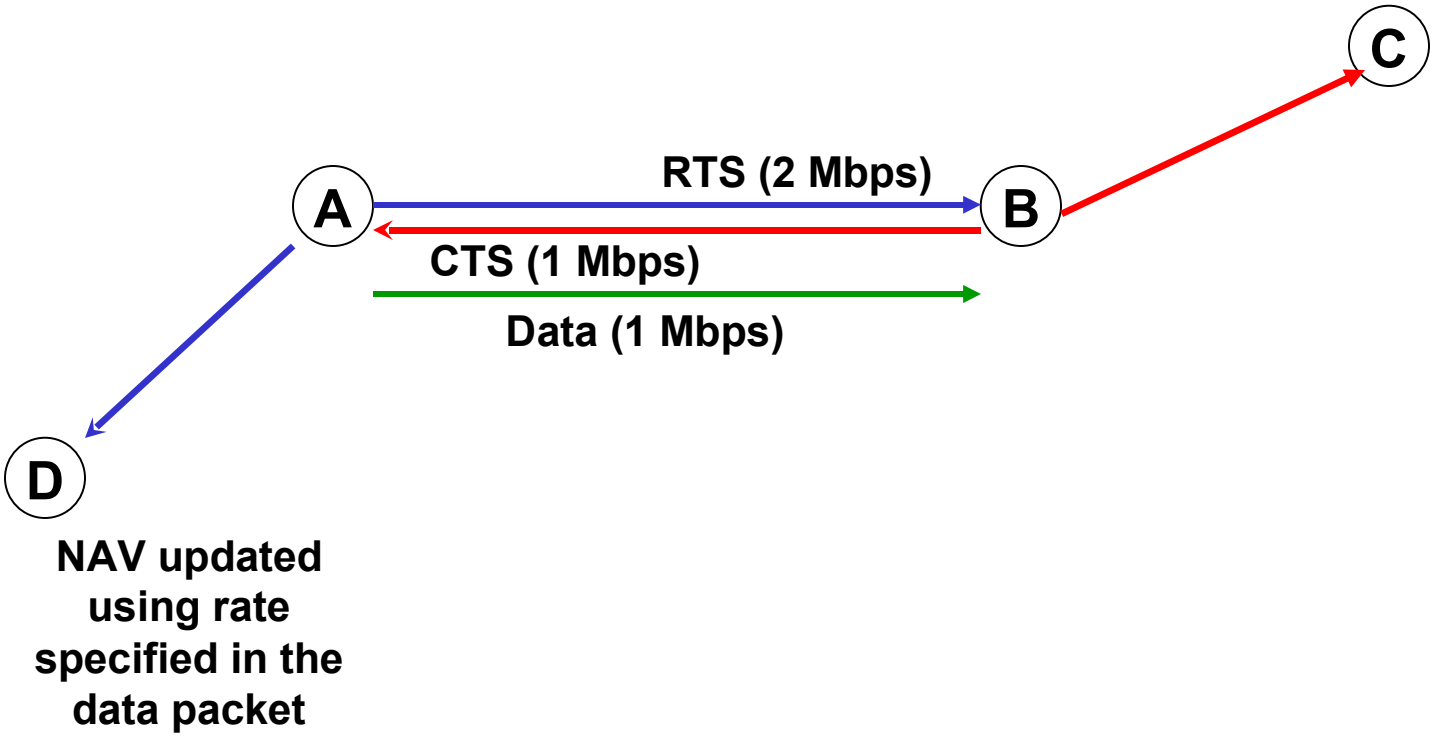
Sender sends RTS containing its best rate estimate

Receiver chooses best rate for the conditions and sends it in the CTS

Sender transmits DATA packet at new rate

Information in data packet header implicitly updates nodes that heard old rate

Receiver-Based Autorate MAC Protocol



Multiple Channels

Multiple Channels

Multiple channels in ad hoc networks: typically defined FDMA

TDMA requires time synchronization among hosts in ad hoc network

- Difficult

Many MAC protocols have been proposed

Multi-Channel MAC: A simple approach

Divide bandwidth into multiple channels

Choose any one of the idle channels

Use a single-channel protocol on the chosen channel

- for instance, 802.11

Multi-Channel MAC with Soft Reservation [Nasipuri00]

Similar to the simple scheme, channel used recently for a successful transmission preferred

Tends to “reserve” channels

Another Protocol

Use one (control) channel for RTS/CTS and remaining (data) channels for DATA/ACK

Each host maintains NAV table, with one entry for each data channel

Sender sends RTS to destination, specifying the channels that are free per sender's table

Receiver replies with CTS specifying a channel that it also thinks is free

- A channel is used only if both sender and receiver conclude that it is free

Impact of Directional Antennas on MAC and Routing

Impact of Antennas on MAC

Wireless hosts traditionally use
single-mode antennas

Typically, the
single-mode = omni-directional

Recently, antennas with *multiple modes* (often, but not necessarily, directional) have been develop

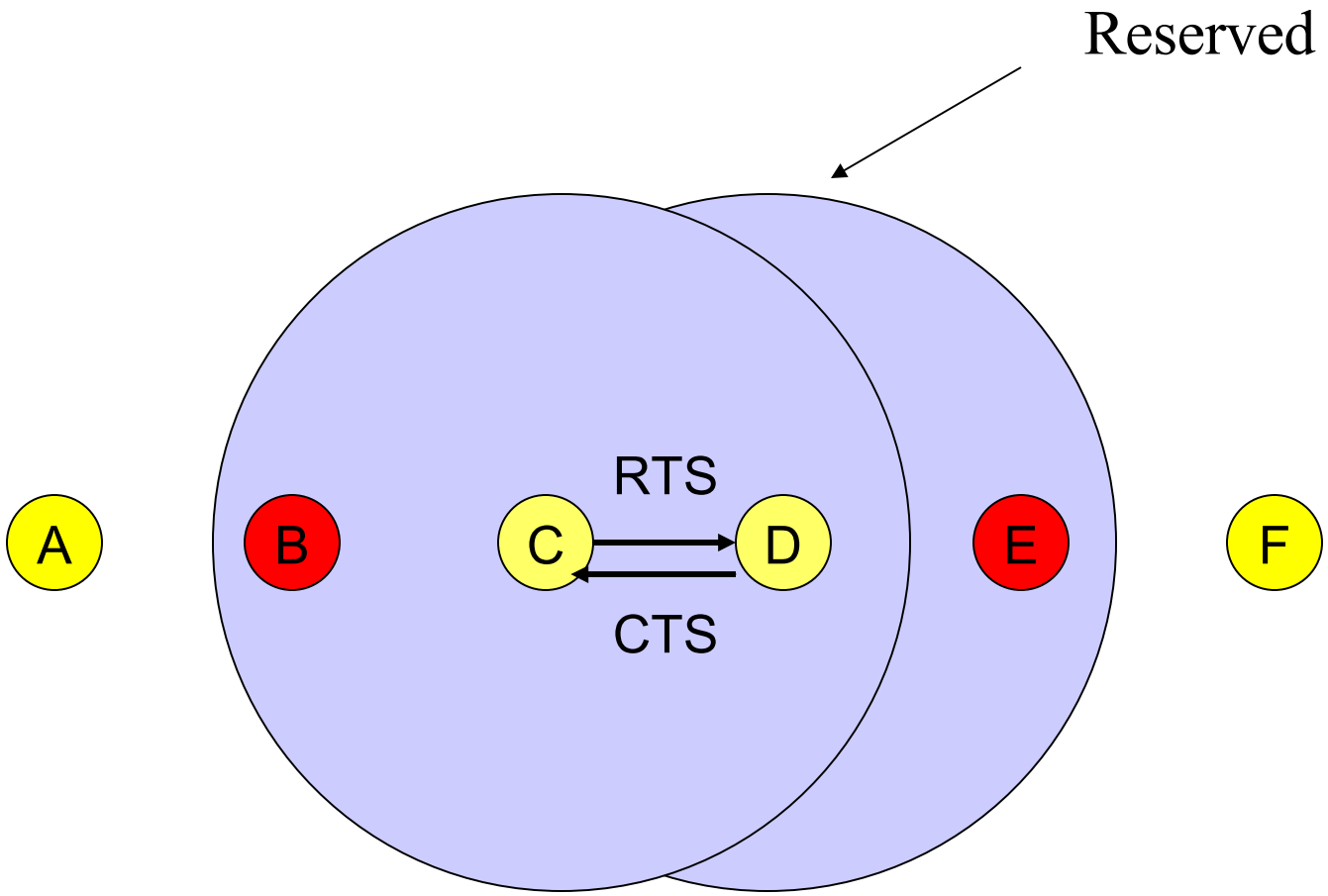
We will now focus on directional antennas with multiple modes

IEEE 802.11

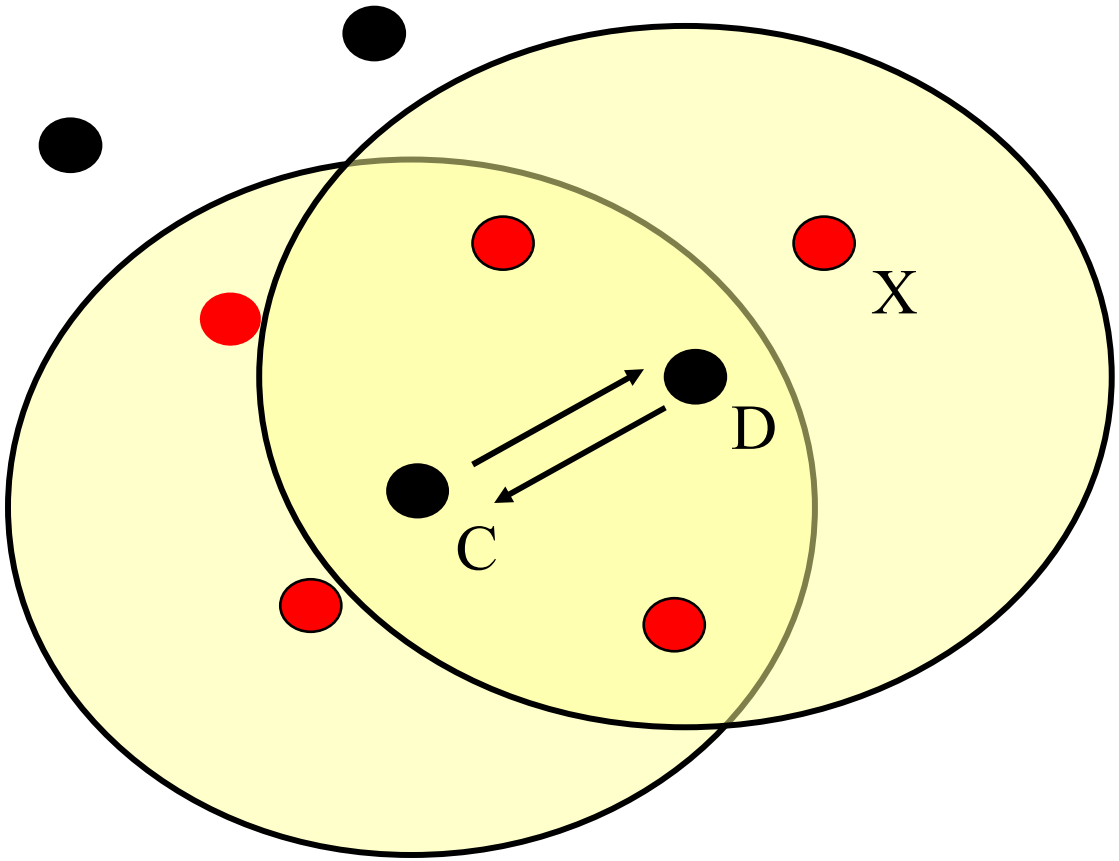
Implicitly assumes single mode antennas

Typically, omnidirectional antennas (though not necessarily)

IEEE 802.11

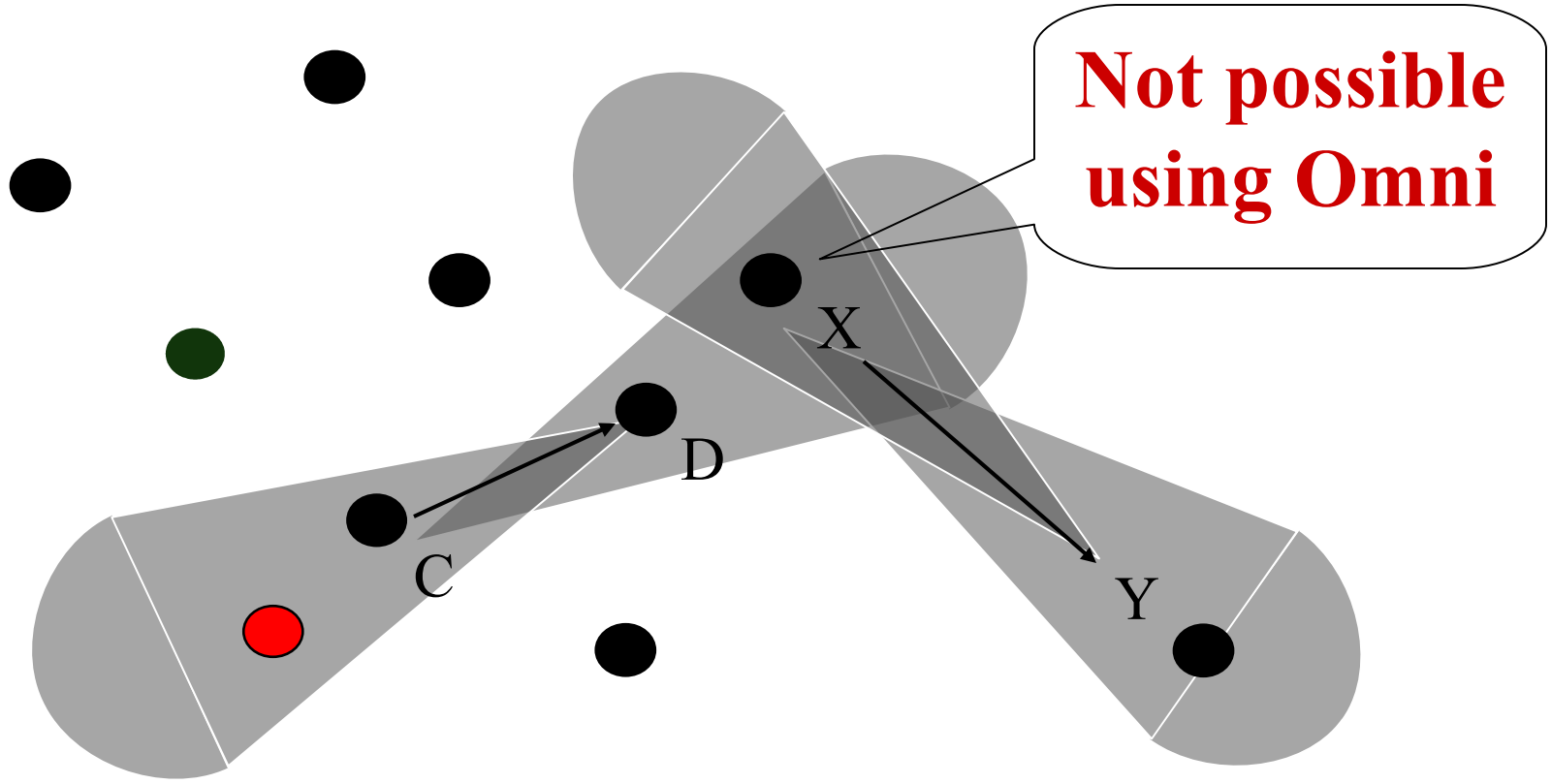


Omni-Directional Antennas



Red nodes
Cannot
Communicate
presently

Directional Antennas



Question

How to exploit directional antennas in ad hoc networks ?

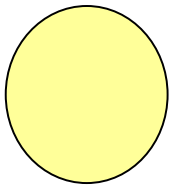
- Medium access control
- Routing

Antenna Model

In Omni Mode:

Nodes receive signals with gain $\mathbf{G^o}$

While idle a node stays in omni mode

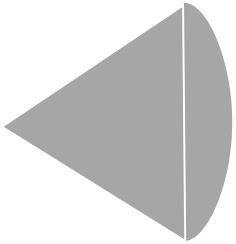


In Directional Mode:

Capable of beamforming in specified direction

Directional Gain $\mathbf{G^d}$ $\mathbf{(G^d > G^o)}$

Directional mode has sidelobes



Symmetry: Transmit gain = Receive gain

Directional Communication

Received Power

$\propto$

(Transmit power) *(Tx Gain) * (Rx Gain)

Directional gain is higher

Potential Benefits of Directional Antennas

Increase “range”, keeping transmit power constant

Reduce transmit power, keeping range comparable with omni mode

- Several proposal focus on this benefit
- Assume that range of omni-directional and directional transmission is equal
- ➔Directional transmissions at lower power

Caveats

Only most important features of the protocols discussed here

Antenna characteristics assumed are often different in different papers

Simple Tone Sense (STS) Protocol

[Yum1992IEEE Trans. Comm.]

STS Protocol

Based on busy tone signaling:

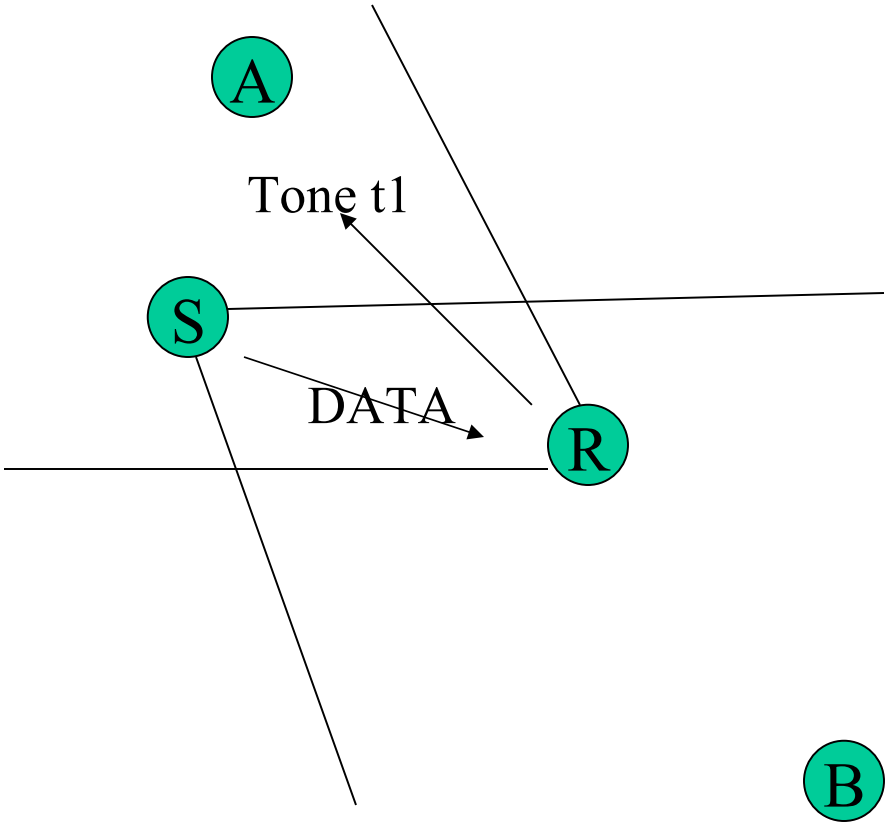
- Each host is assigned a *tone* (sinusoidal wave at a certain frequency)
- Tone frequency unique in each host's neighborhood
- When a host detects a packet destined to itself, it transmit a tone
- If a host receive a tone on directional antenna A,it assumes that some host in that direction is receiving a packet
  - Cannot transmit using antenna A presently
  - OK to transmit using other antennas

STS Protocol

Tone duration used to encode information

- Duration t1 implies transmitting node is busy
- Duration t2 implies the transmitting node successfully received a transmission from another node

Example



Node A cannot Initiate a transmission.

But B can send to C

Because B does not receive t1

STS Protocol

Issues:

- Assigning tones to hosts
- Assigning hosts to antennas: It is assumed that the directions/angles can be chosen
  - distribute neighbor hosts evenly among the antennas
  - choose antenna angles such that adjacent antennas have some minimum separation

Busy Tone Directional MAC
[Huang2002MILCOM]

Extends the busy tone (DBTMA) protocol originally proposed by omni-directional antennas [Deng98ICUPC]

Three channels

- Data channel
- Two Busy Tone channels
  - Receive tone (BTr)
  - Transmit tone (BTt)

DBTMA

Sender:

Sense BTr. If sensed busy, defer transmission.

If BTr idle, transmit RTS to receiver

Receiver

On receiving RTS, sense BTt.

If BTt idle, reply with a CTS, and transmit BTr until DATA is completely received

Sender

On receiving CTS, transmit DATA and BTt both

DBTMA + Directional Antennas

DBTMA reduces reduction in throughput caused by collisions by hidden terminals

Directional antennas can be used to transmit the busy tones directionally

- RTS/CTS, DATA, *busy tones* all may be sent directionally
- Trade-offs similar to directional versus omni-directional transmission of RTS/CTS

Another Directional MAC protocol
[Roychoudhury02mobicom]

Derived from IEEE 802.11 (similar to [Takai02mobihoc])

A node listens omni-directionally when idle

Sender transmits Directional-RTS (DRTS) towards receiver

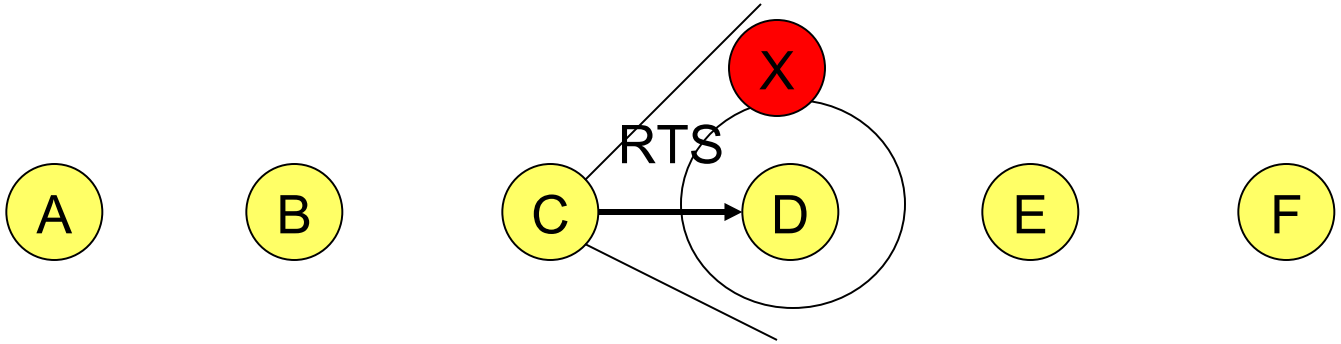
RTS received in Omni mode (idle receiver in when idle)

Receiver sends Directional-CTS (DCTS)

DATA, ACK transmitted and received directionally

Directional MAC

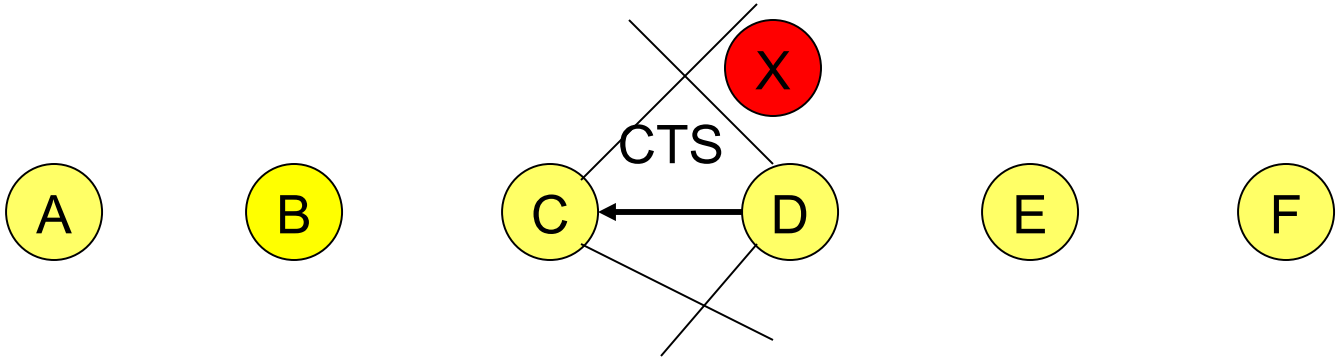
RTS = Request-to-Send



Pretending a circular range for omni

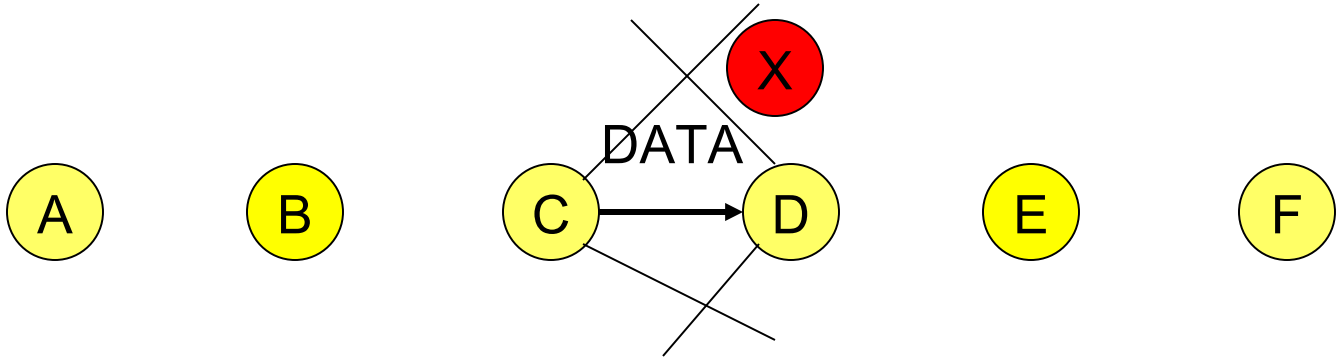
Directional MAC

CTS = Clear-to-Send

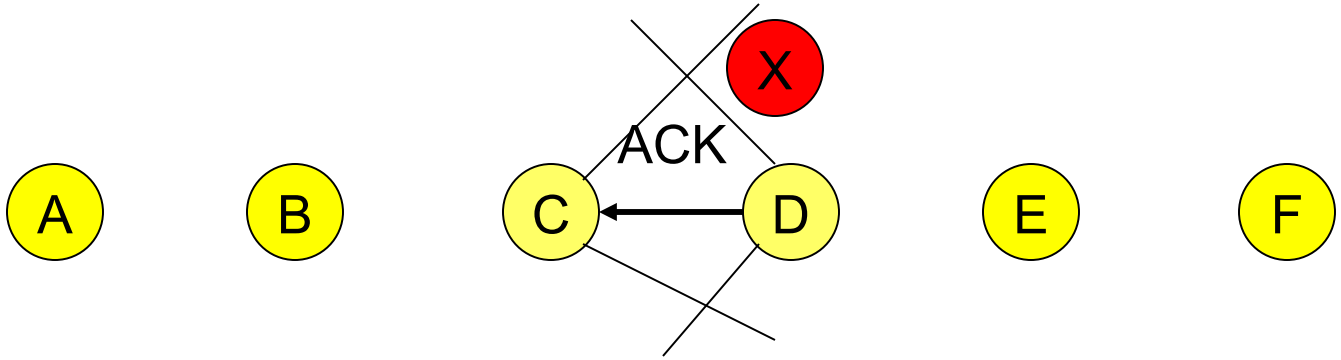


Directional MAC

- DATA packet follows CTS. Successful data reception acknowledged using ACK.

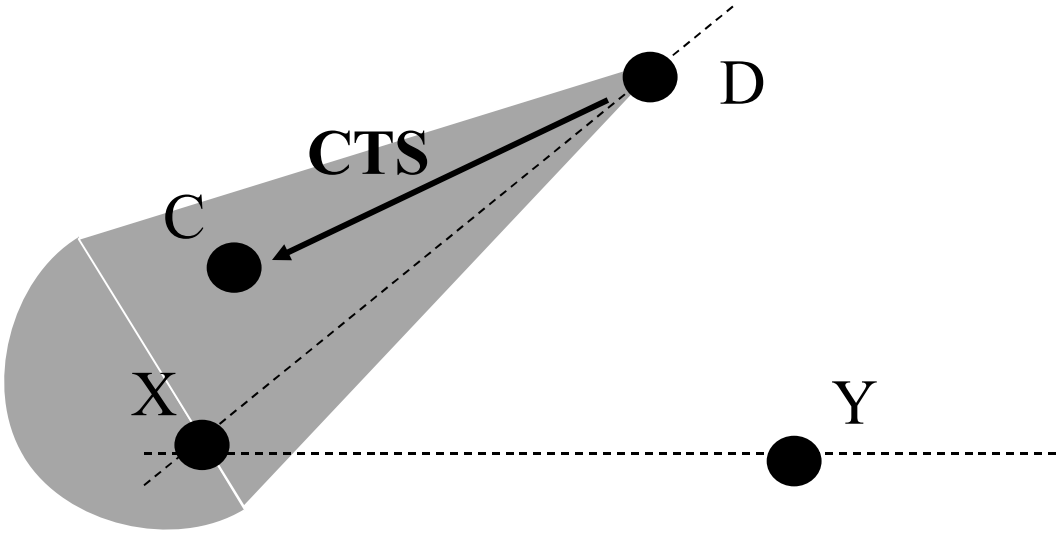


Directional MAC



Directional NAV (DNAV)
[Roychoudhury02mobicom]

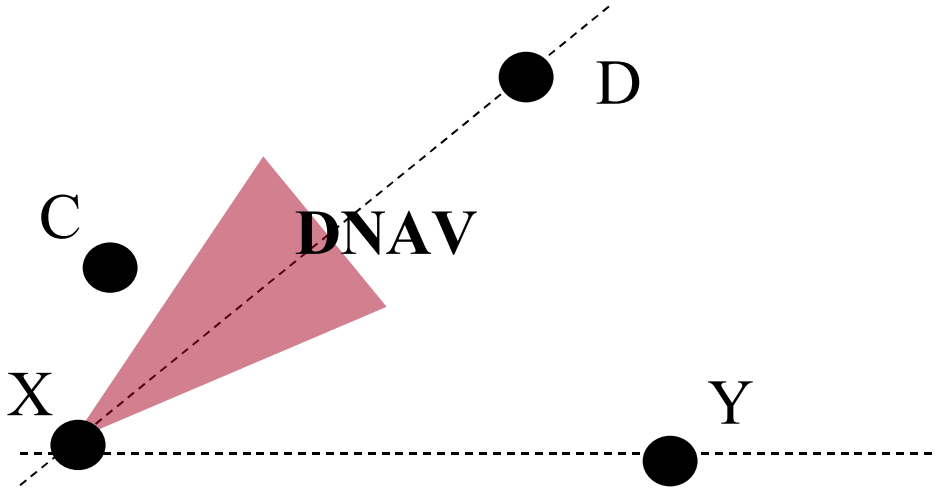
Nodes overhearing RTS or CTS set up directional NAV (DNAV) for that Direction of Arrival (DoA)



Similar DNAV mechanism proposed in [Takai02mobihoc]

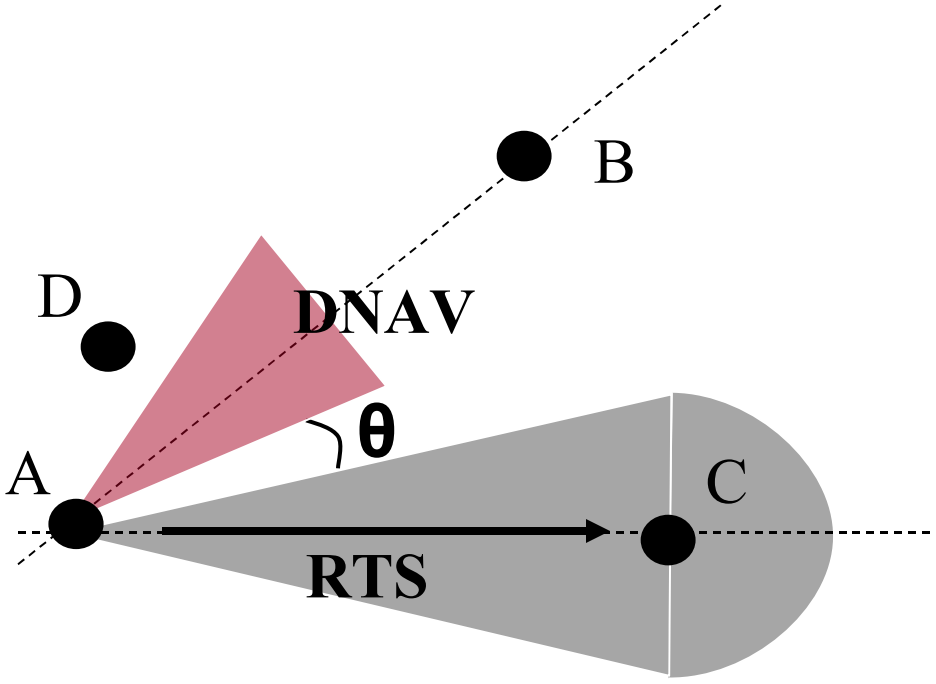
Directional NAV (DNAV)

Nodes overhearing RTS or CTS set up directional NAV (DNAV) for that Direction of Arrival (DoA)

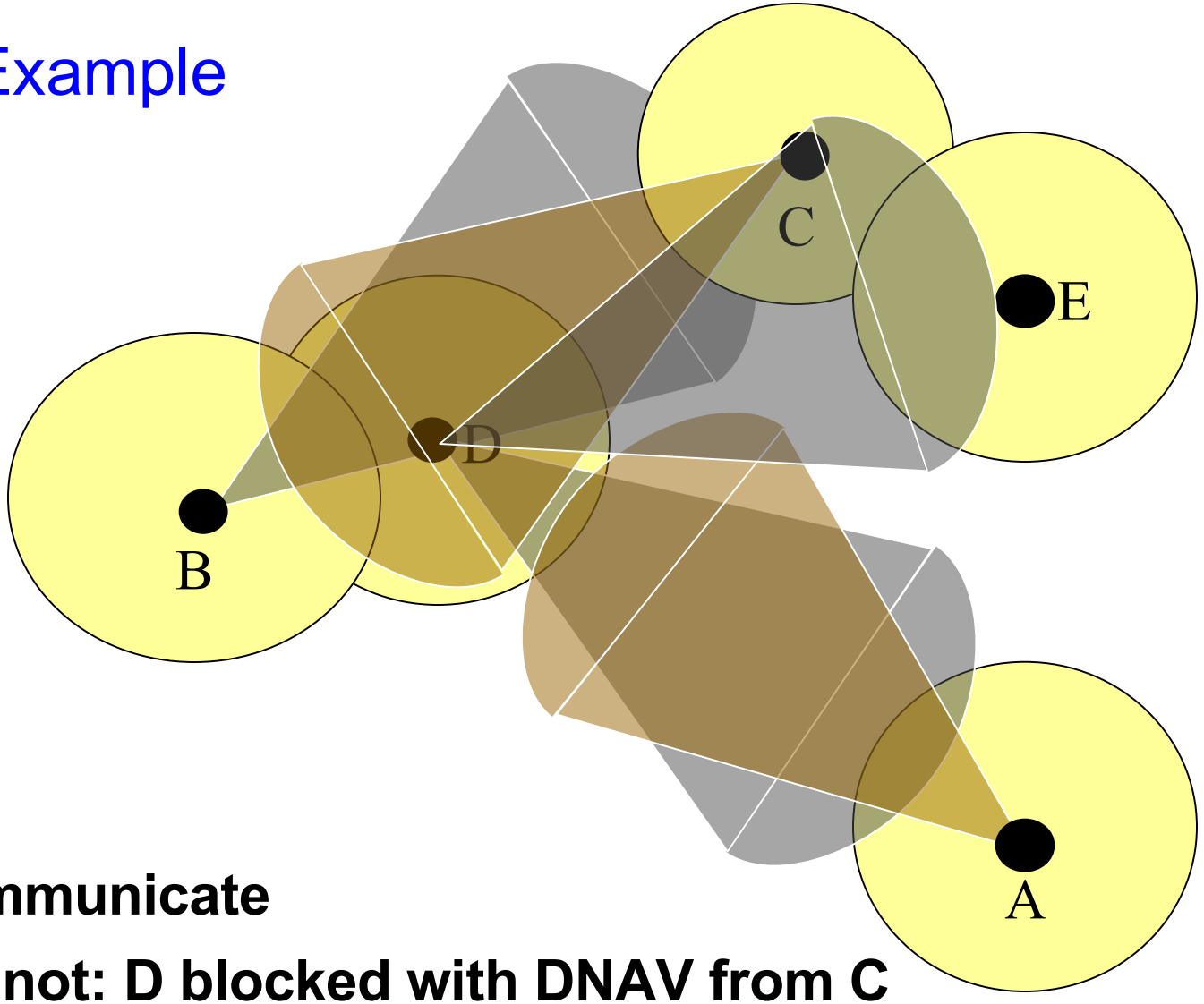


Directional NAV (DNAV)

New transmission initiated only if direction of transmission does not overlap with DNAV,
i.e., if (θ > 0)



DMAC Example



B and C communicate

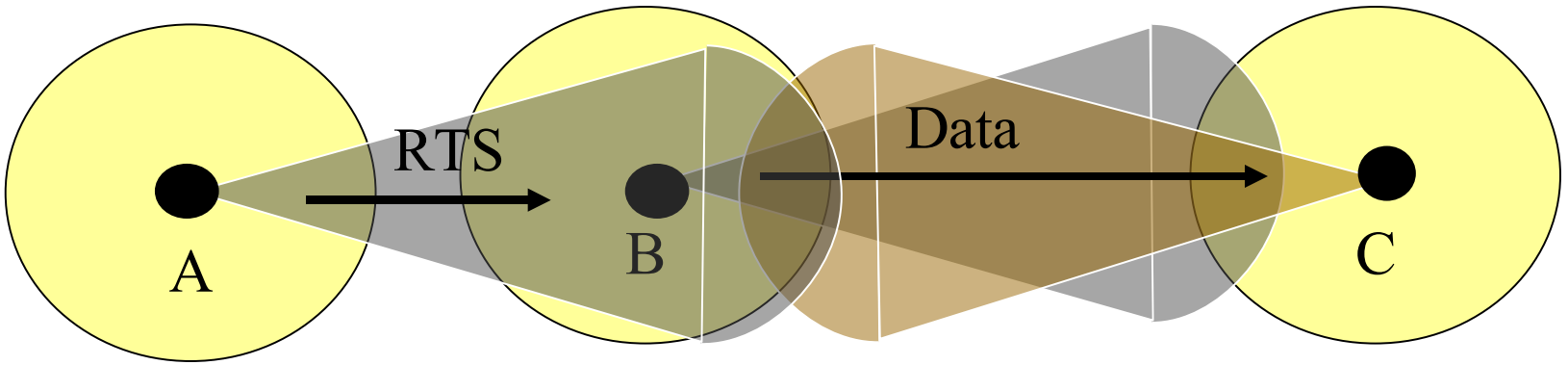
D and E cannot: D blocked with DNAV from C

D and A communicate

Issues with DMAC

Two types of Hidden Terminal Problems

Due to asymmetry in gain



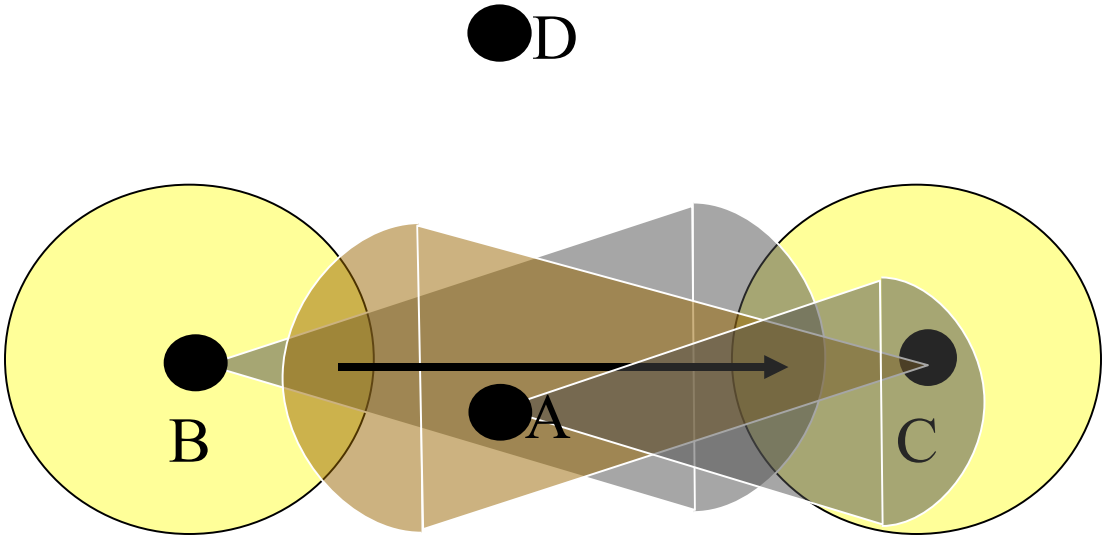
A is unaware of communication between B and C

A's RTS may interfere with C's reception of DATA

Issues with DMAC

Two types of Hidden Terminal Problems

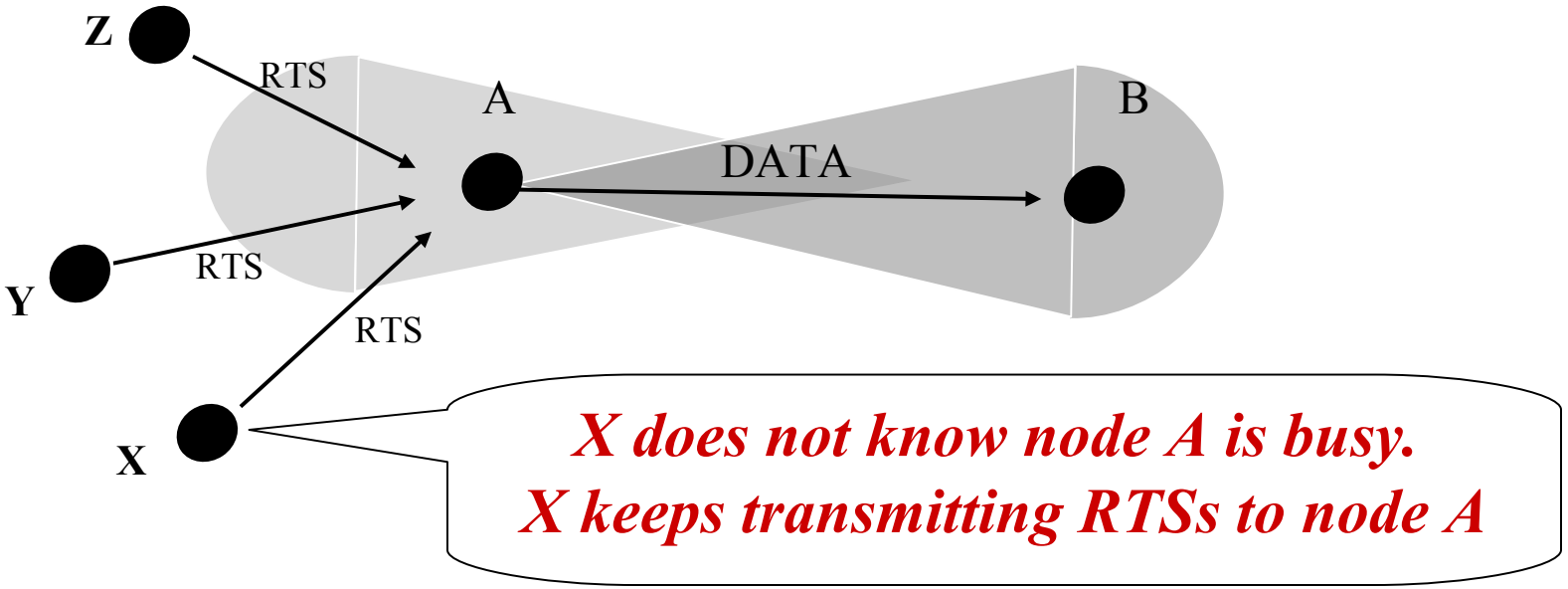
Due to unheard RTS/CTS



Node A may now interfere at node **C** by transmitting in C's direction

Issues with DMAC

- Deafness



Using omni antennas, X would be aware that A is busy, and defer its own transmission

Issues with DMAC

Uses DO links, but not DD links

DMAC Tradeoffs

Benefits

Better Network Connectivity

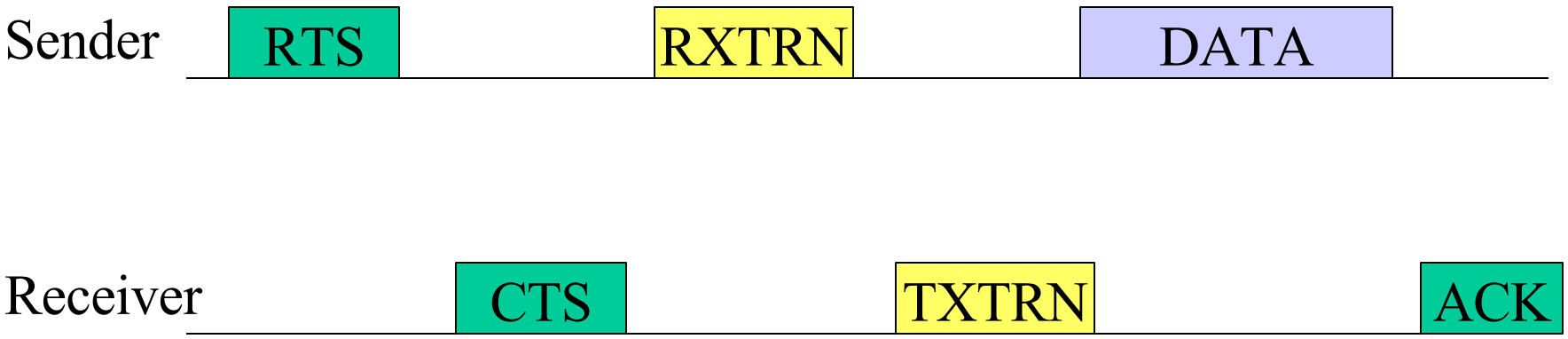
Spatial Reuse

- **Disadvantages**
  - ***Hidden terminals***
  - ***Deafness***
  - ***No DD Links***

Using Training Sequences

[Bellofiore2002IEEETrans.Ant.Prop]

Training packets used for DoA determination, after RTS/CTS exchange omni-directionally



Performance depends on the TXTRN and RXTRN delays

If direction is known a priori, then these delays can potentially be avoided

- But mobility can change direction over time

Another Variation
[Nasipuri2000WCNC]

Similar to 802.11, but adapted for directional antennas

Assumptions:

- Antenna model: Several directional antennas which can all be used simultaneously
- Omni-directional reception is possible (by using all directional antennas together)
- Direction of arrival (DoA) can be determined when receiving omni-directionally
- Range of directional and omni transmissions are identical

Protocol Description

Sender sends omni-directional RTS

Receiver sends omni-directional CTS

- Receiver also records direction of sender by determining the antenna on which the RTS signal was received with highest power level
- Similarly, the sender, on receiving CTS, records the direction of the receiver

All nodes overhearing RTS/CTS defer transmissions

Sender then sends DATA directionally to the receiver

Receiver sends directional ACK

Discussion

Protocol takes advantage of reduction in interference due to directional transmission/reception of DATA

All neighbors of sender/receiver defer transmission on receiving omni-directional RTS/CTS

➔ spatial reuse benefit not realized

Enhancing DMAC

Are improvements possible to make DMAC more effective ?

Possible improvements:

- Make Use of DD Links [Roychoudhury02MobiCom]
- Overcome deafness [Roychoudhury03techrep]

Directional MAC: Summary

Directional antennas need adaptation of the MAC protocols

MAC protocols show improvement in aggregate throughput and delay

- But not always

Performance dependent on topology

Routing with Directional Antennas

Routing Protocols

Many routing protocols for ad hoc networks rely on broadcast messages

- For instance, flood of route requests (RREQ)

Using omni antennas for broadcast will not discover all possible links

Need to implement broadcast using directional transmissions

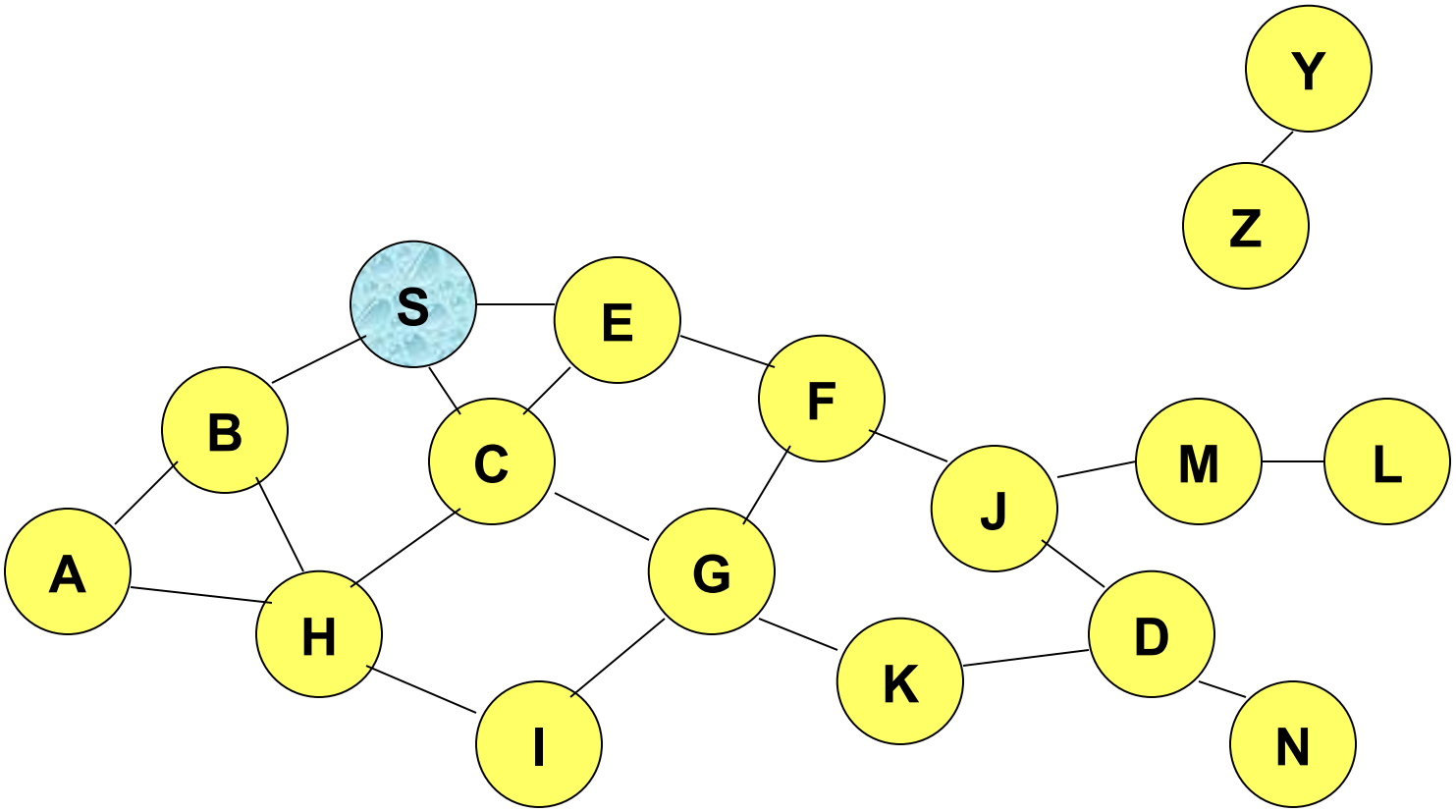
Dynamic Source Routing [Johnson]

Sender floods RREQ through the network

Nodes forward RREQs after appending their names

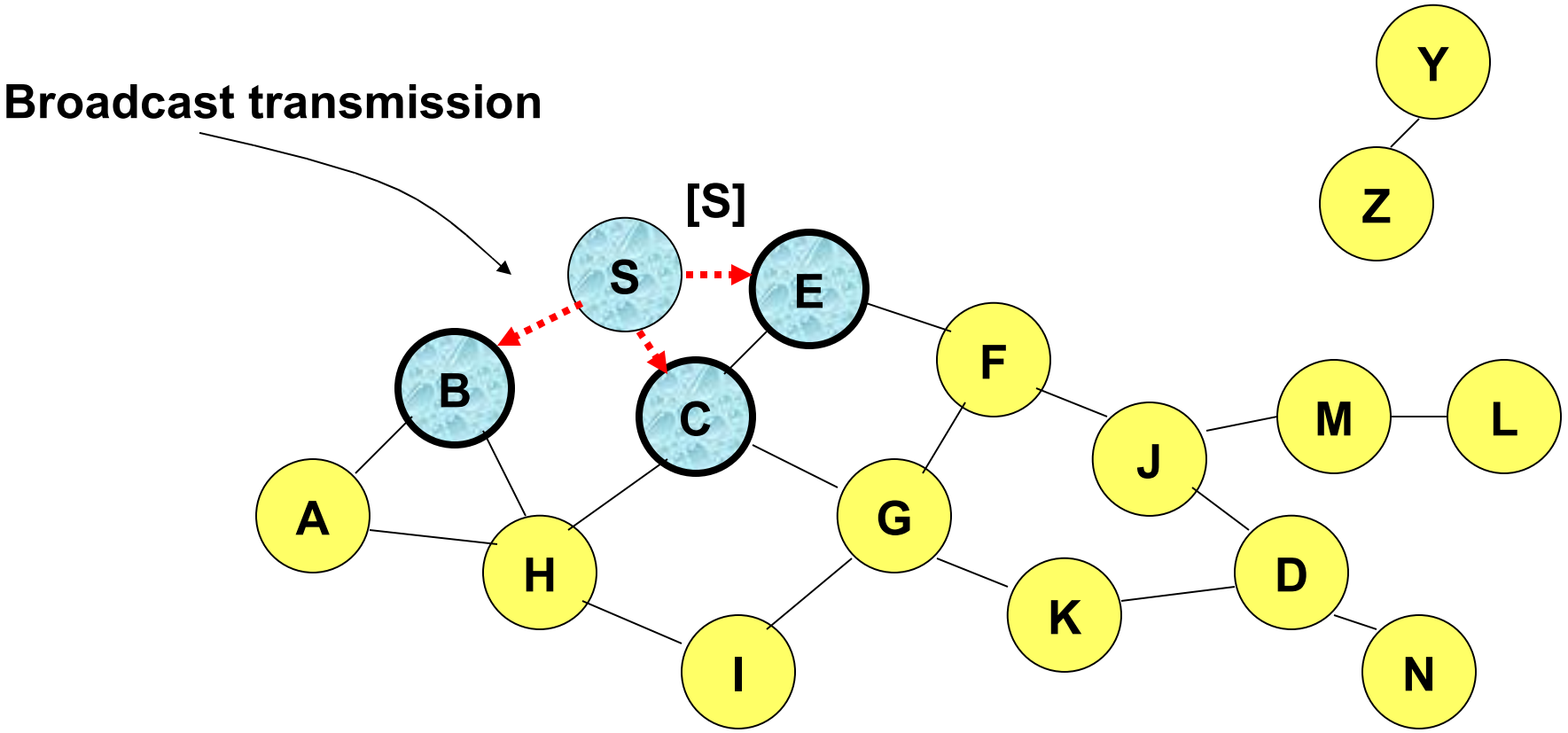
Destination node receives RREQ and unicasts a RREP back to sender node, using the route in which RREQ traveled

Route Discovery in DSR



Represents a node that has received RREQ for D from S

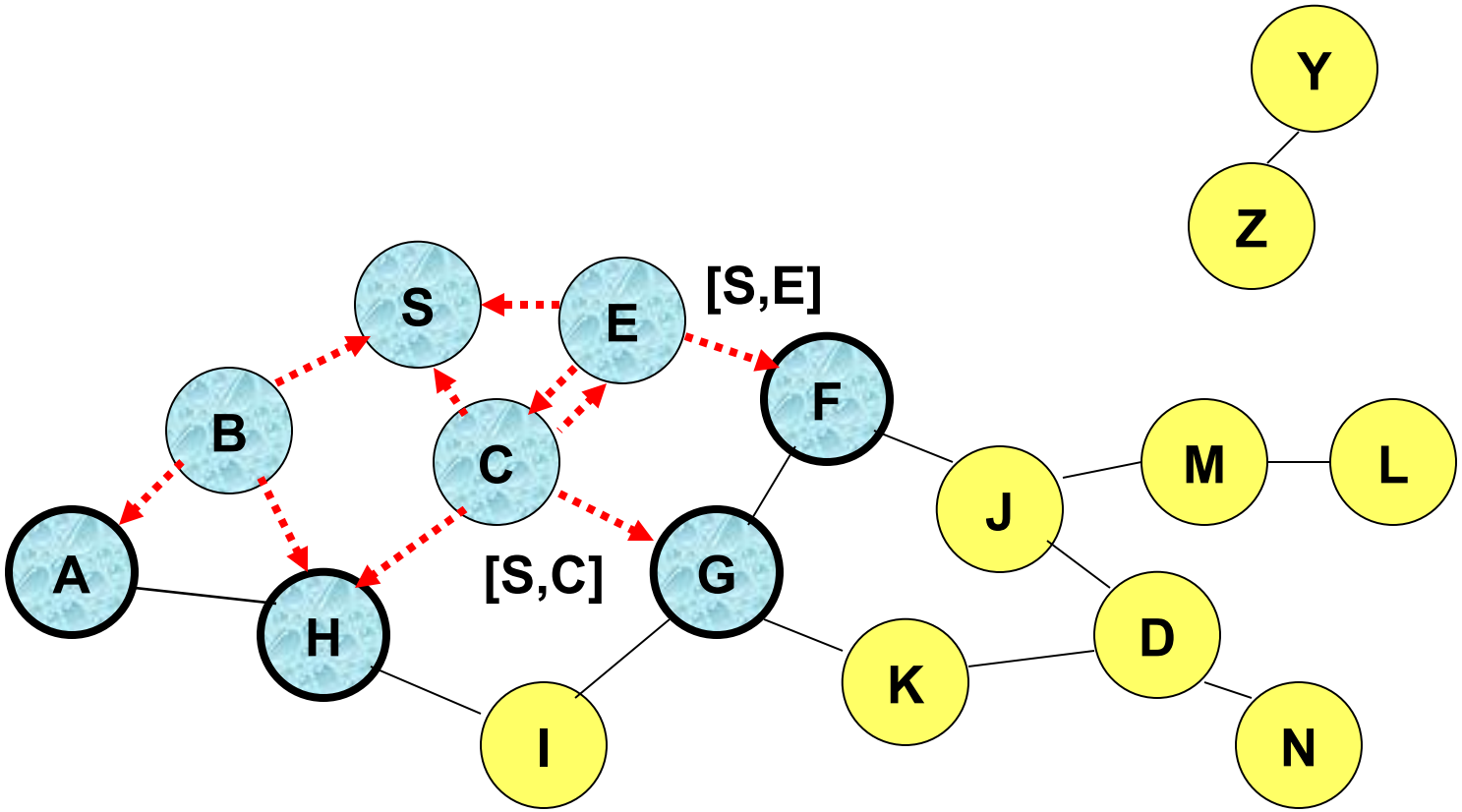
Route Discovery in DSR



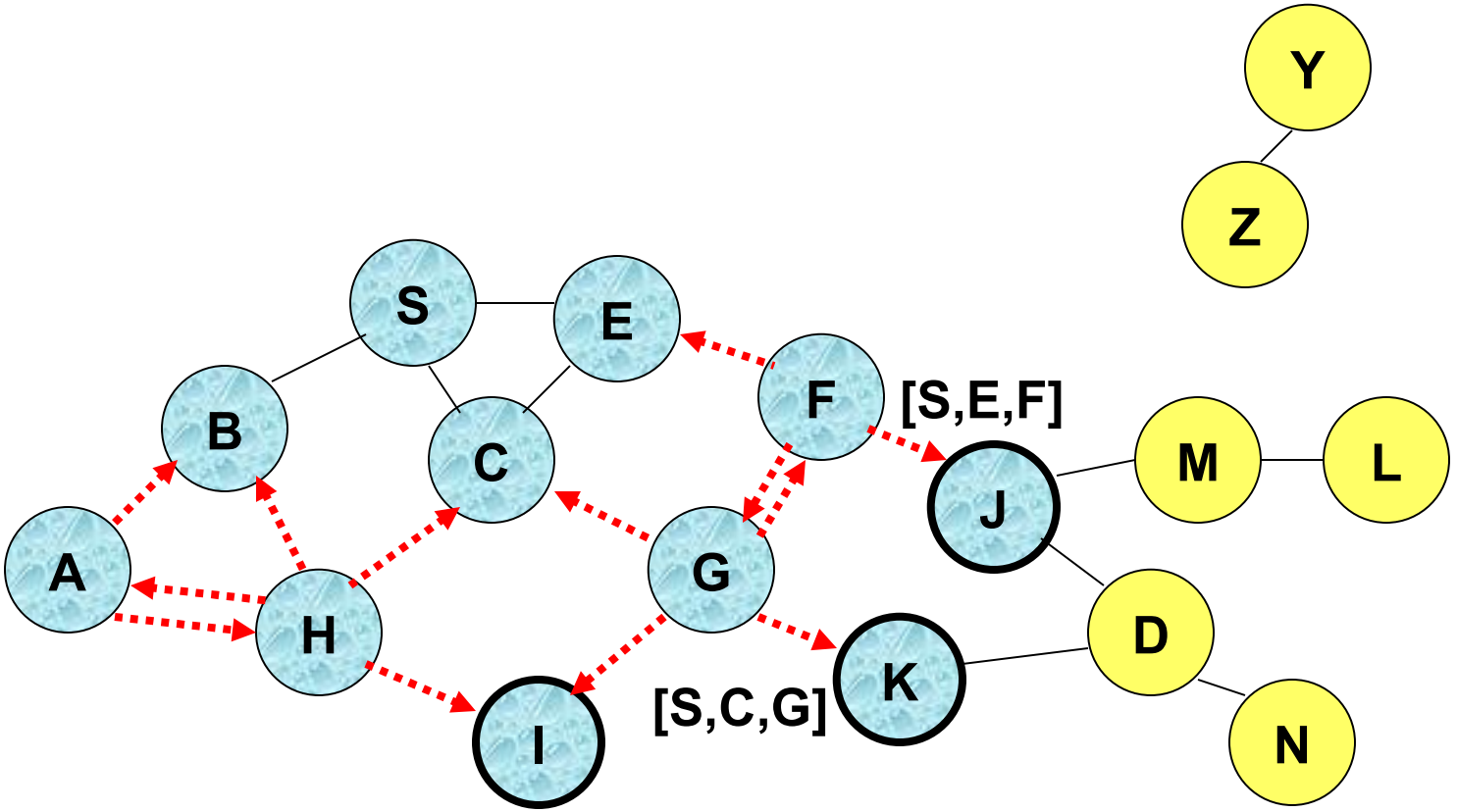
Represents transmission of RREQ

[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR

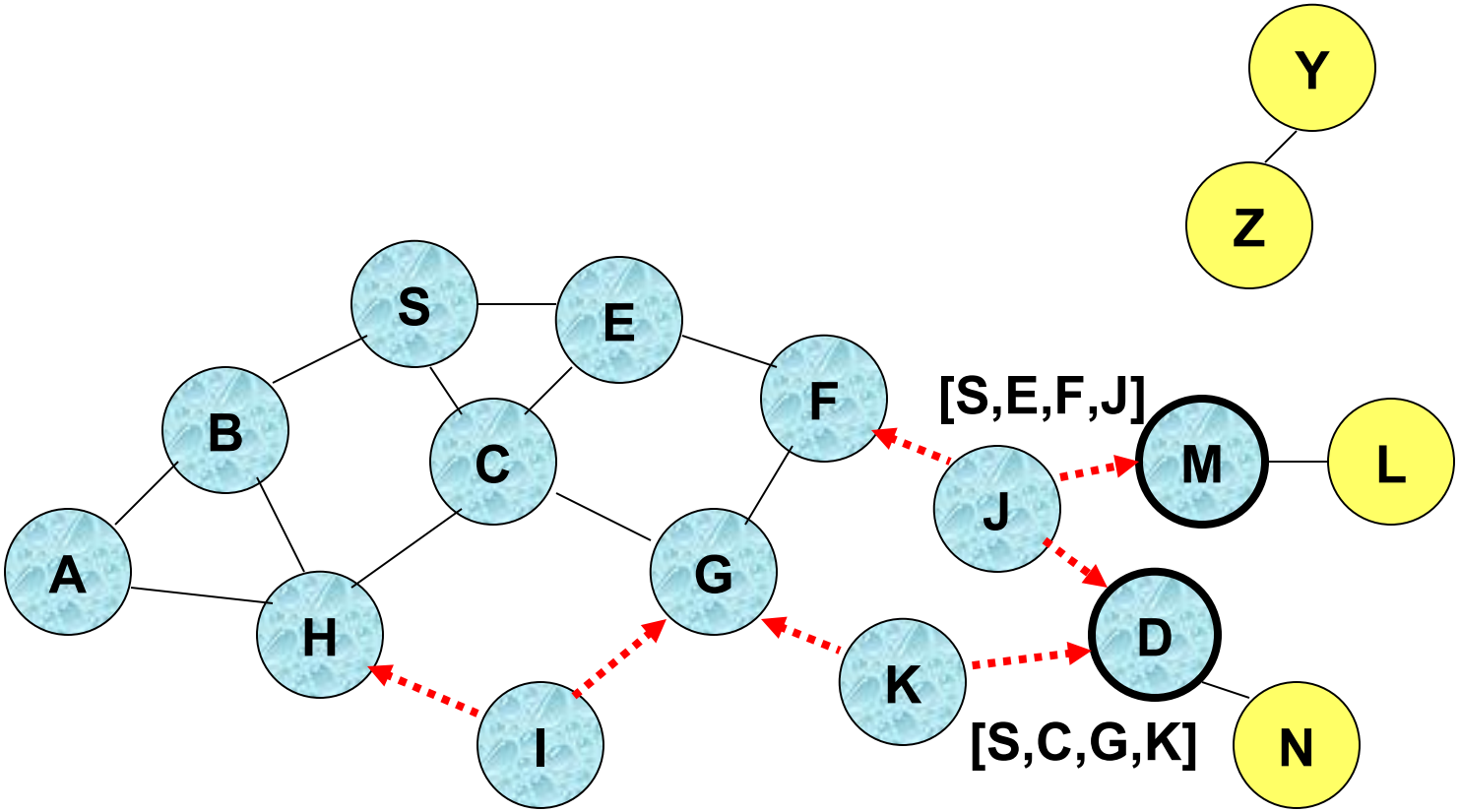


Route Discovery in DSR



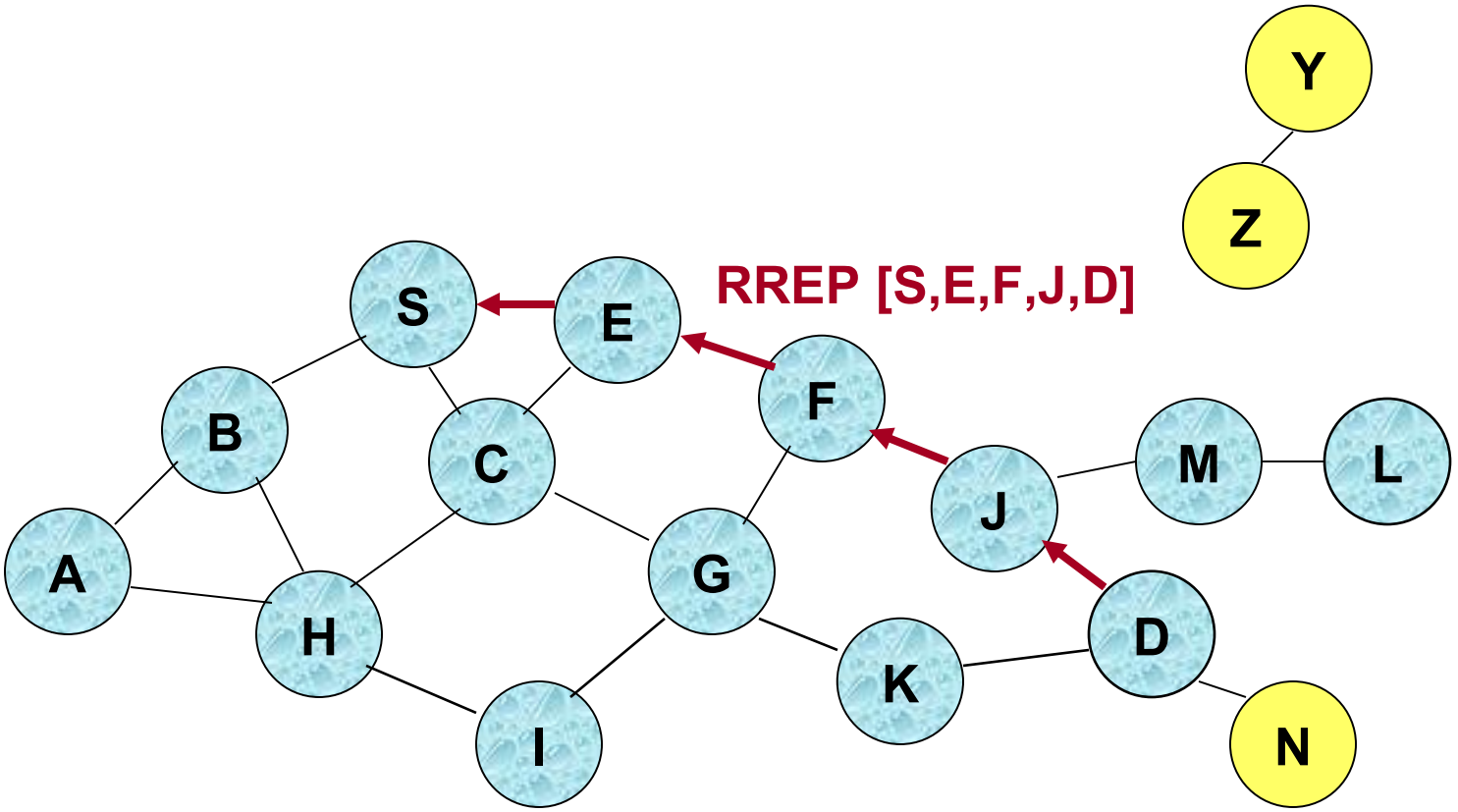
- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

Route Discovery in DSR



- **Nodes J and K both broadcast RREQ to node D**

Route Reply in DSR



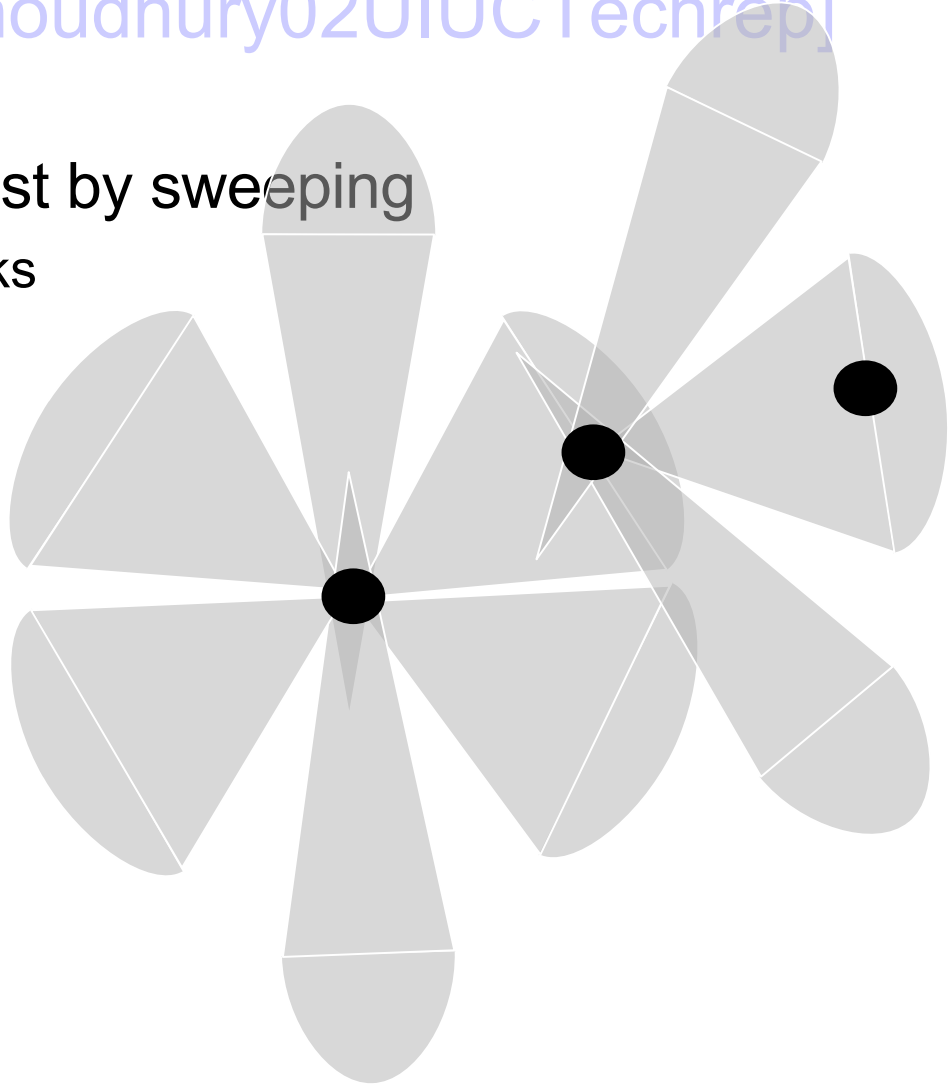
Represents RREP control message

DSR over Directional Antennas

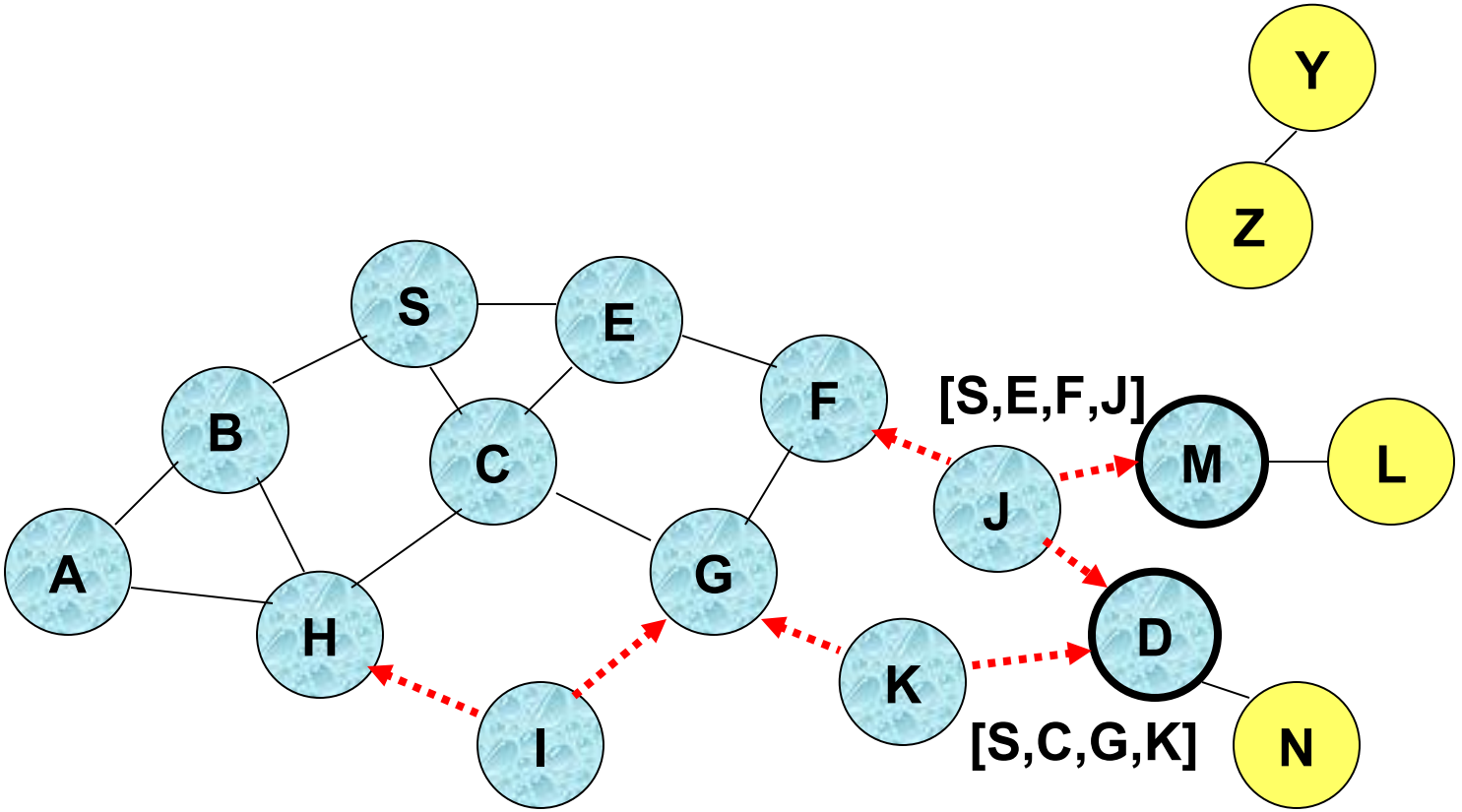
[Roychoudhury03PWC, Roychoudhury02UIUCTechrep]

RREQ broadcast by sweeping

To use DD links



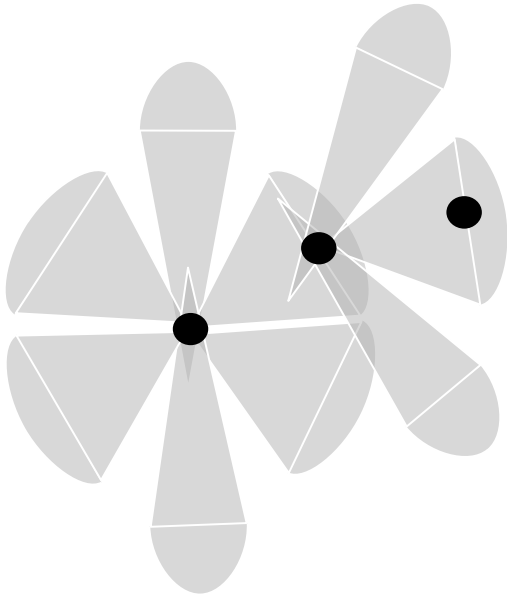
Route Discovery in DSR



- **Nodes J and K both broadcast RREQ to node D**

Directional Routing

Broadcast by sweeping

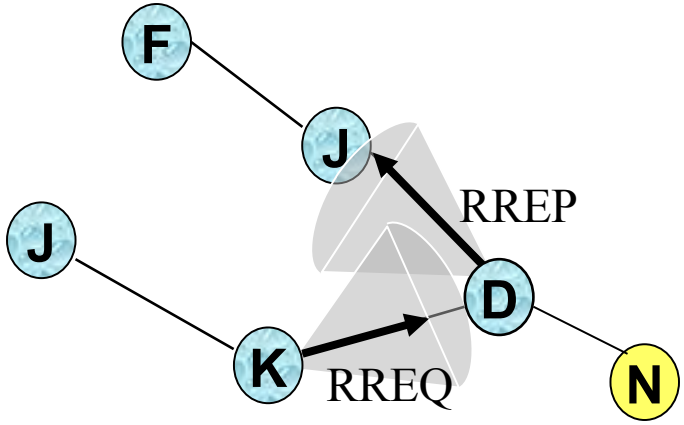


Tradeoffs

Larger Tx Range	⇒	Fewer Hop Routes
Few Hop Routes	⇒	Low Data Latency
Small Beamwidth	⇒	High Sweep Delay
More Sweeping	⇒	High Overhead

Issues

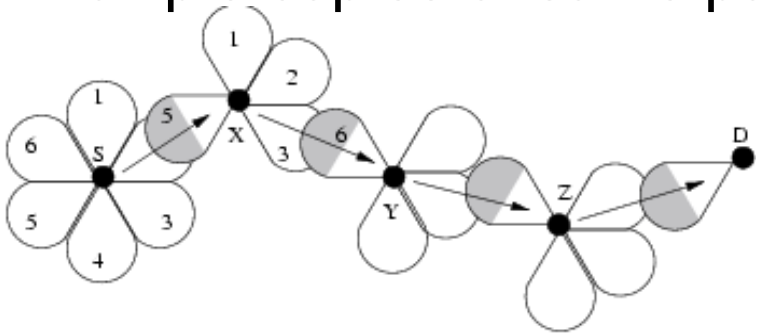
Sub-optimal routes may be chosen if destination node misses shortest request, while beamformed



D misses request from K

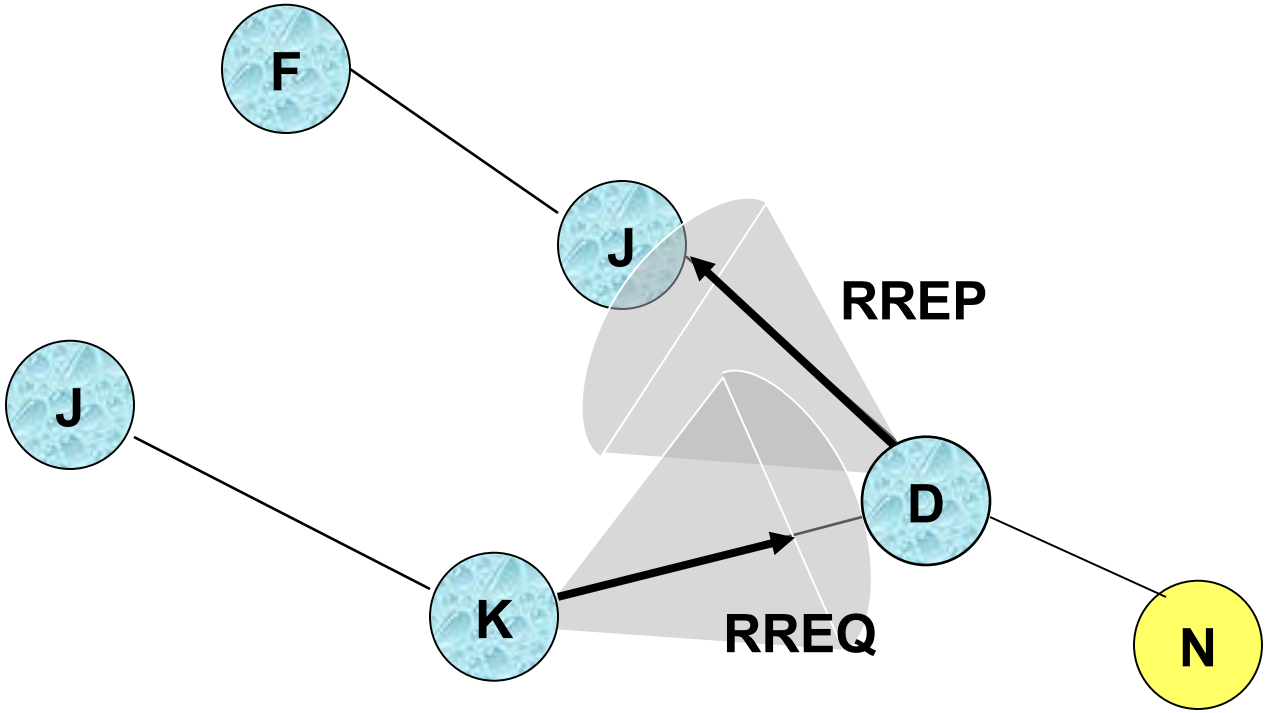
Optimize by having destination wait before replying

Broadcast storm: Using broadcasts, nodes receive multiple copies of same packet



Use K antenna elements to forward broadcast packet

Route Discovery in DSR



D receives RREQ from J, and replies with RREP

D misses RREQ from K

Delayed RREP Optimization

Due to sweeping – earliest RREQ need not have traversed shortest hop path.

- RREQ packets sent to different neighbors at different points of time

If destination replies to first arriving RREP, it might miss shorter-path RREQ

Optimize by having DSR destination wait before replying with RREP

Routing Overhead

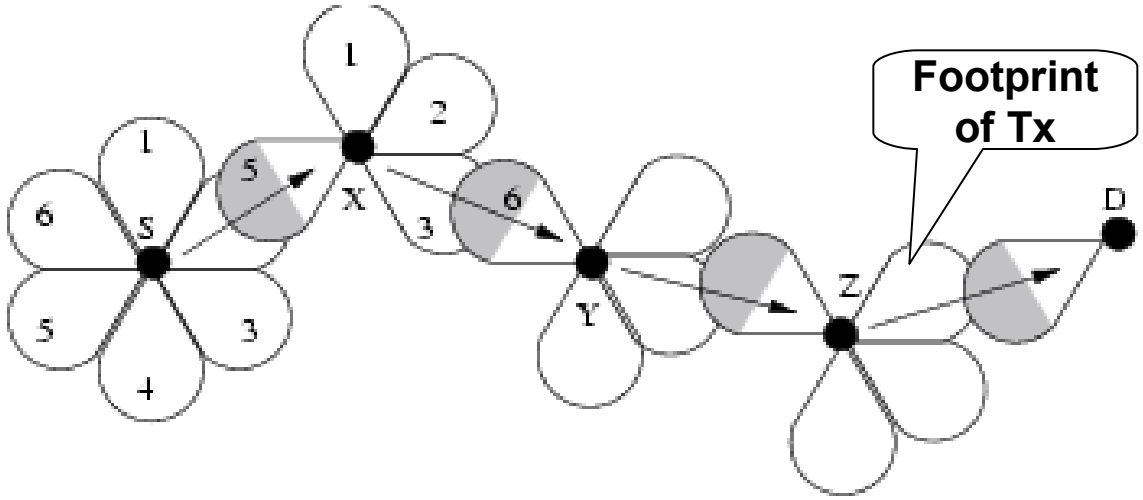
Using omni broadcast, nodes receive multiple copies of same packet - Redundant !!!

- Broadcast Storm Problem

Using directional Antennas – can do better ?

Routing Overhead

Use K antenna elements to forward broadcast packet



$$\text{New measure for control Overhead} = \frac{\Sigma \text{ (number of control packets) } \times \text{ (footprint of transmissions)}}{\Sigma \text{ No. Data Packets}}$$

Mobility

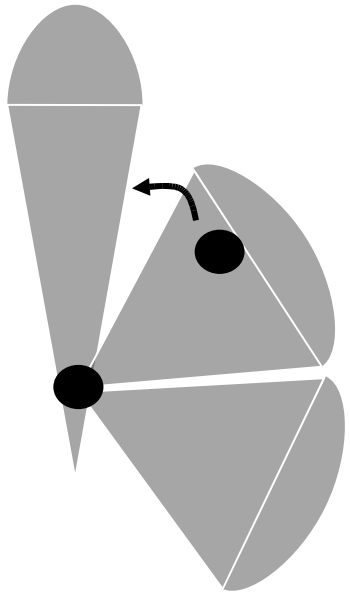
Link lifetime increases using directional antennas.

- Higher transmission range - link failures are less frequent

Nodes moving out of beam coverage in order of packet-transmission-time

- Low probability

Mobility



Antenna handoff

- If no response to RTS, MAC layer uses N adjacent antenna elements to transmit same packet
- Route error avoided if communication re-established [RoyChoudhury02UIUC Techrep]

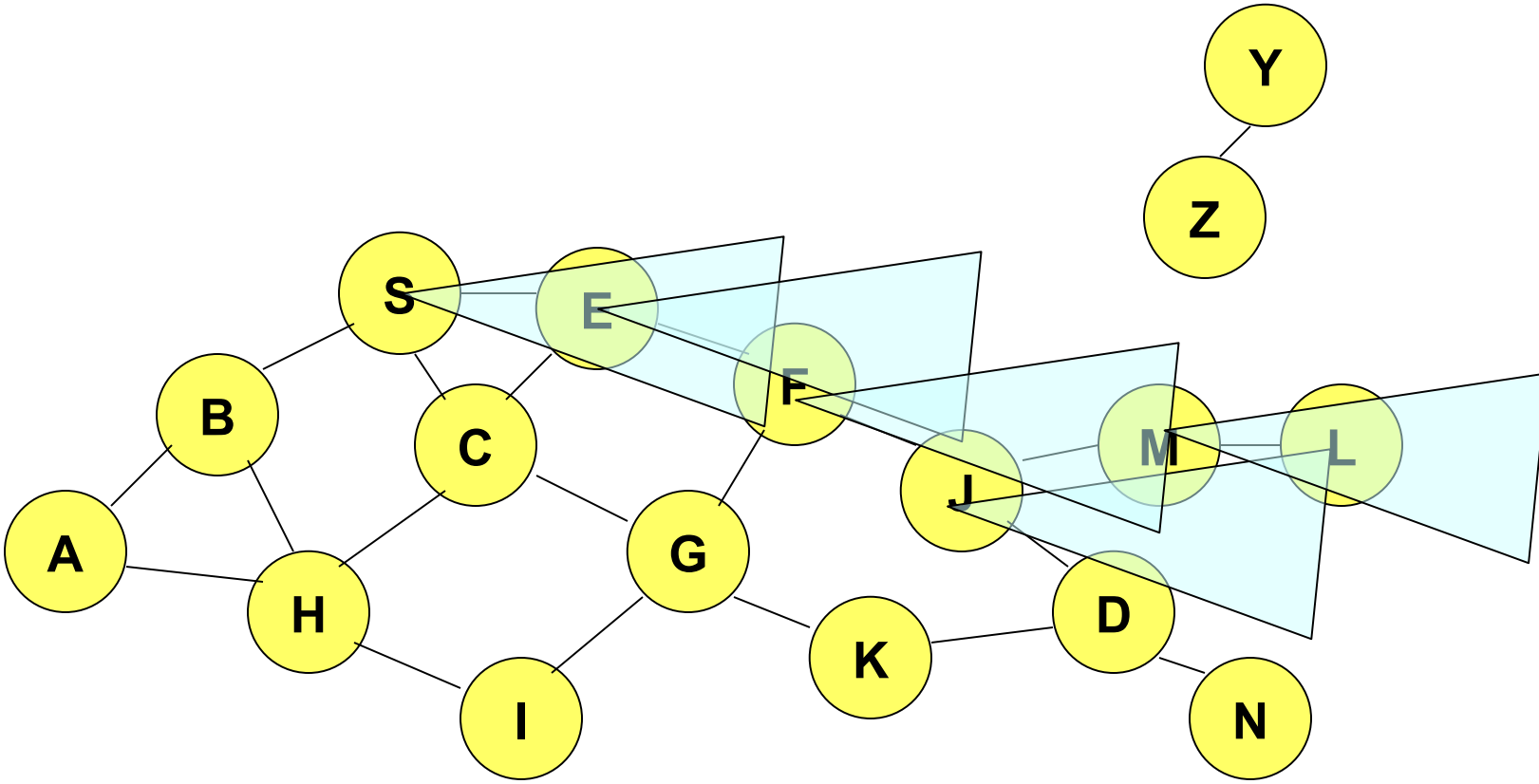
Other Approaches to Routing with Directional Antennas [Nasipuri2000ICCCN]

Modified version of DSR

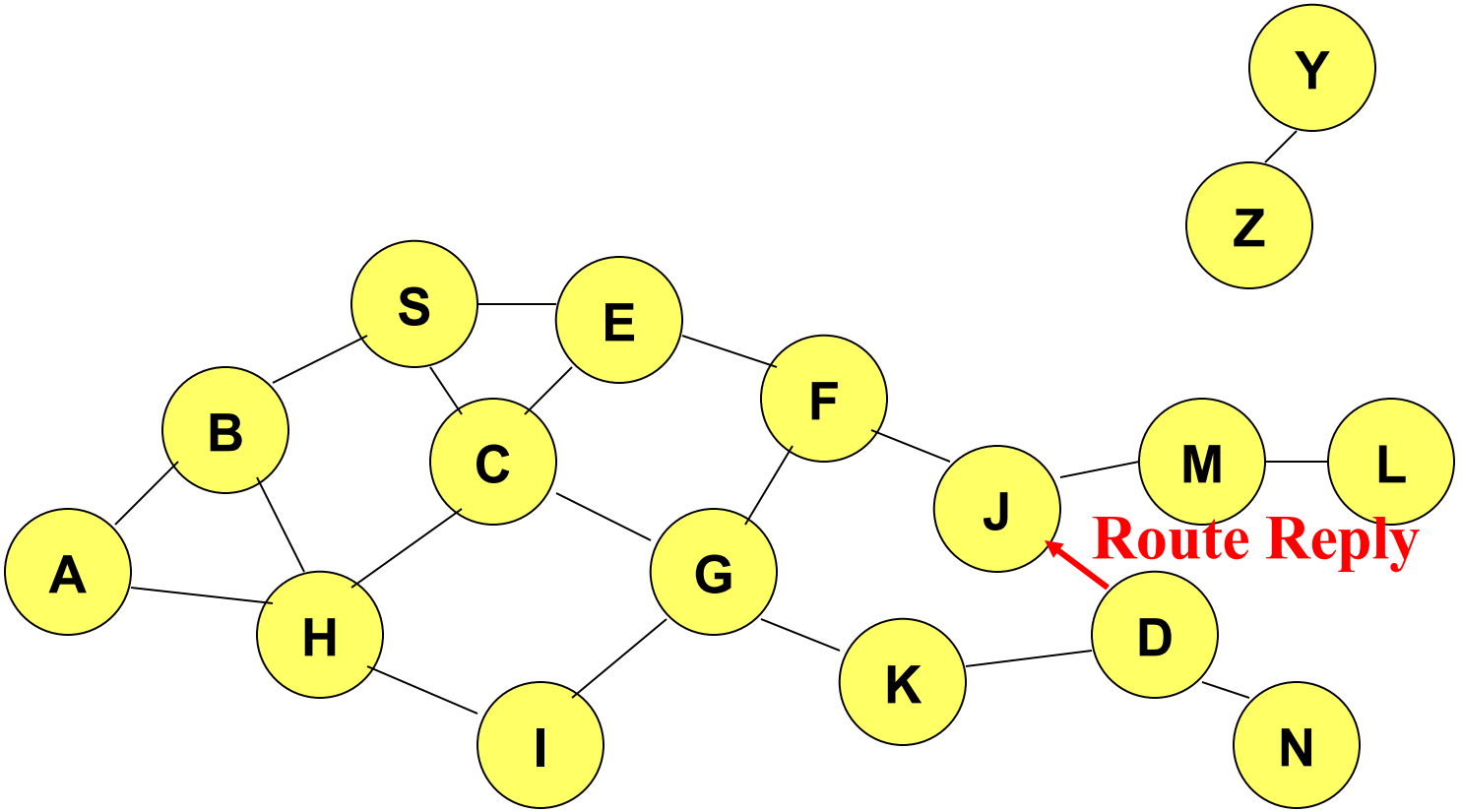
Transmit Route Request in the last known direction of the receiver

If the source S perceives receiver R to have been in direction *d,* then all nodes forward the route request from S in direction *d.*

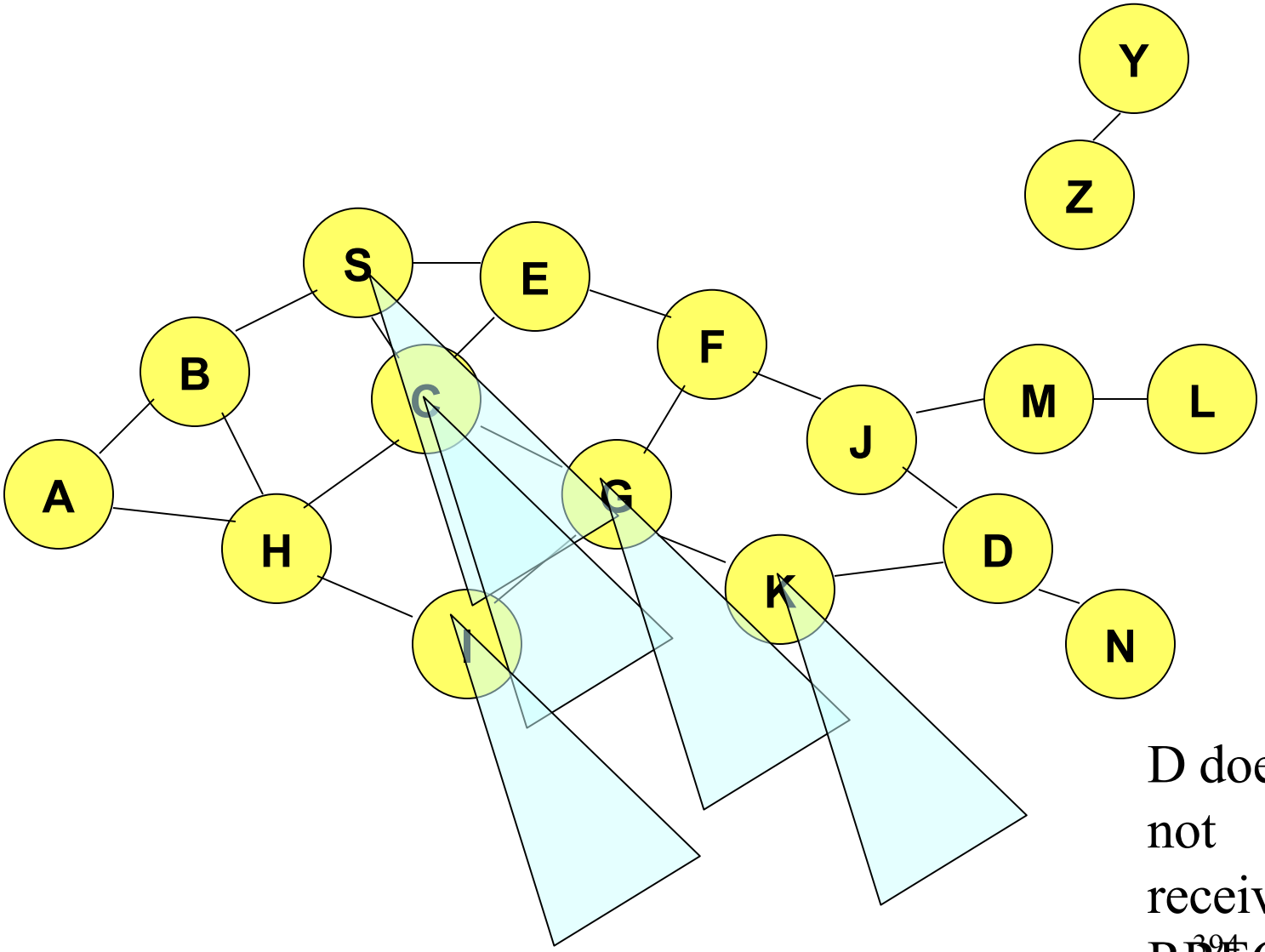
Example 1



Example 1



Example 2



D does not receive RREQ

Limited Forwarding

Benefit: Limits the forwarding of the Route Request

Disadvantage: Effectively assumes that each node has a sense of *orientation*

Routing with Directional Antennas: Conclusion

Directional antennas can improve routing performance

But suitable protocol adaptations necessary

Directional Antennas: Conclusion

Directional antennas can potentially benefit

But also create difficulties in MAC and routing protocol design

UDP on
Mobile Ad Hoc Networks

User Datagram Protocol (UDP)

UDP provides unreliable delivery

Studies comparing different routing protocols for MANET typically measure UDP performance

Several performance metrics are often used

- Routing overhead per data packet
- Packet loss rate
- Packet delivery delay

UDP Performance

Several relevant studies [Broch98Mobicom,Das9ic3n,Johansson99Mobicom,Das00Infocom,Jacquet00Inria]

Results comparing a specific pair of protocols do not always agree, but some general (and intuitive) conclusions can be drawn

- Reactive protocols may yield lower routing overhead than proactive protocols when communication density is low
- Reactive protocols tend to loose more packets (assuming than network layer drops packets if a route is not known)
- Proactive protocols perform better with high mobility and dense communication graph

UDP Performance

Many variables affect performance

- Traffic characteristics
  - one-to-many, many-to-one, many-to-many
  - small bursts, large file transfers, real-time, non-real-time
- Mobility characteristics
  - low/high rate of movement
  - do nodes tend to move in groups
- Node capabilities
  - transmission range (fixed, changeable)
  - battery constraints
- Performance metrics
  - delay
  - throughput
  - latency
  - routing overhead
- Static or dynamic system characteristics (listed above)

UDP Performance

Difficult to identify a single scheme that will perform well in all environments

Holy grail: Routing protocol that dynamically adapts to all environments so as to optimize “performance”

- Performance metrics may differ in different environments

TCP on
Mobile Ad Hoc Networks

Overview of Transmission Control Protocol / Internet Protocol (TCP/IP)

Internet Protocol (IP)

Packets may be delivered out-of-order

Packets may be lost

Packets may be duplicated

Transmission Control Protocol (TCP)

Reliable ordered delivery

Implements congestion avoidance and control

Reliability achieved by means of retransmissions if necessary

End-to-end semantics

- Acknowledgements sent to TCP sender confirm delivery of data received by TCP receiver
- Ack for data sent only **after** data has reached receiver

TCP Basics

Cumulative acknowledgements

An acknowledgement ack's all contiguously received data

TCP assigns byte sequence numbers

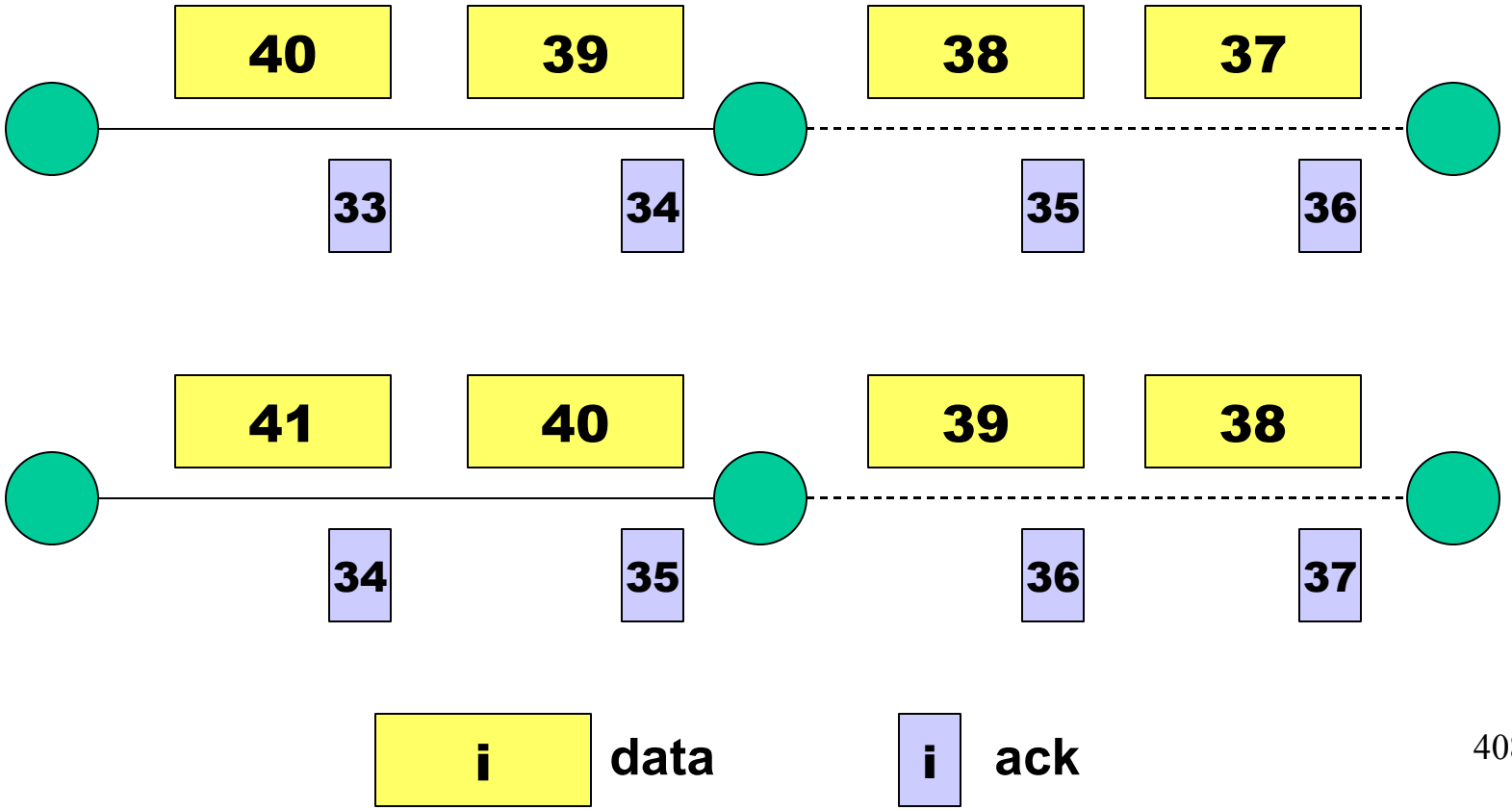
For simplicity, we will assign packet sequence numbers

Also, we use slightly different syntax for acks than normal TCP syntax

In our notation, *ack i* acknowledges receipt of packets through packet *i*

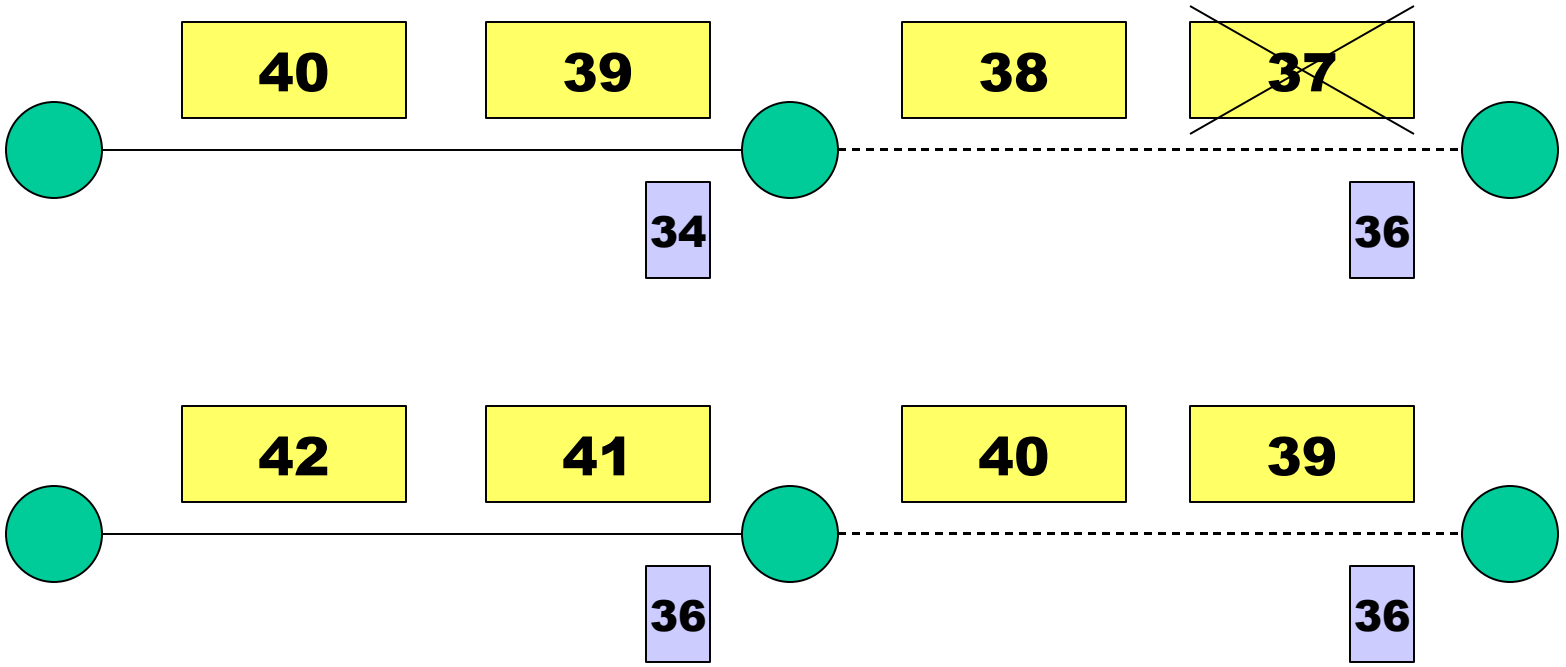
Cumulative Acknowledgements

A new cumulative acknowledgement is generated only on receipt of a **new in-sequence** packet



Duplicate Acknowledgements

A dupack is generated whenever an out-of-order segment arrives at the receiver



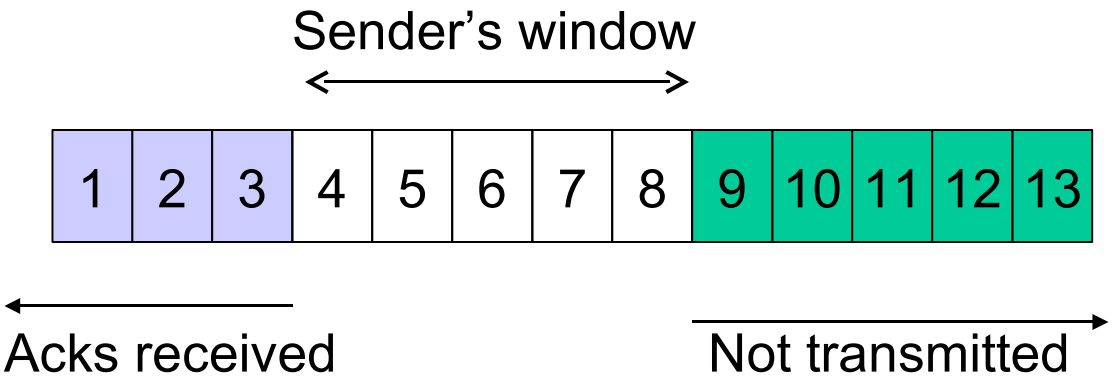
(Above example assumes *delayed acks*)

Window Based Flow Control

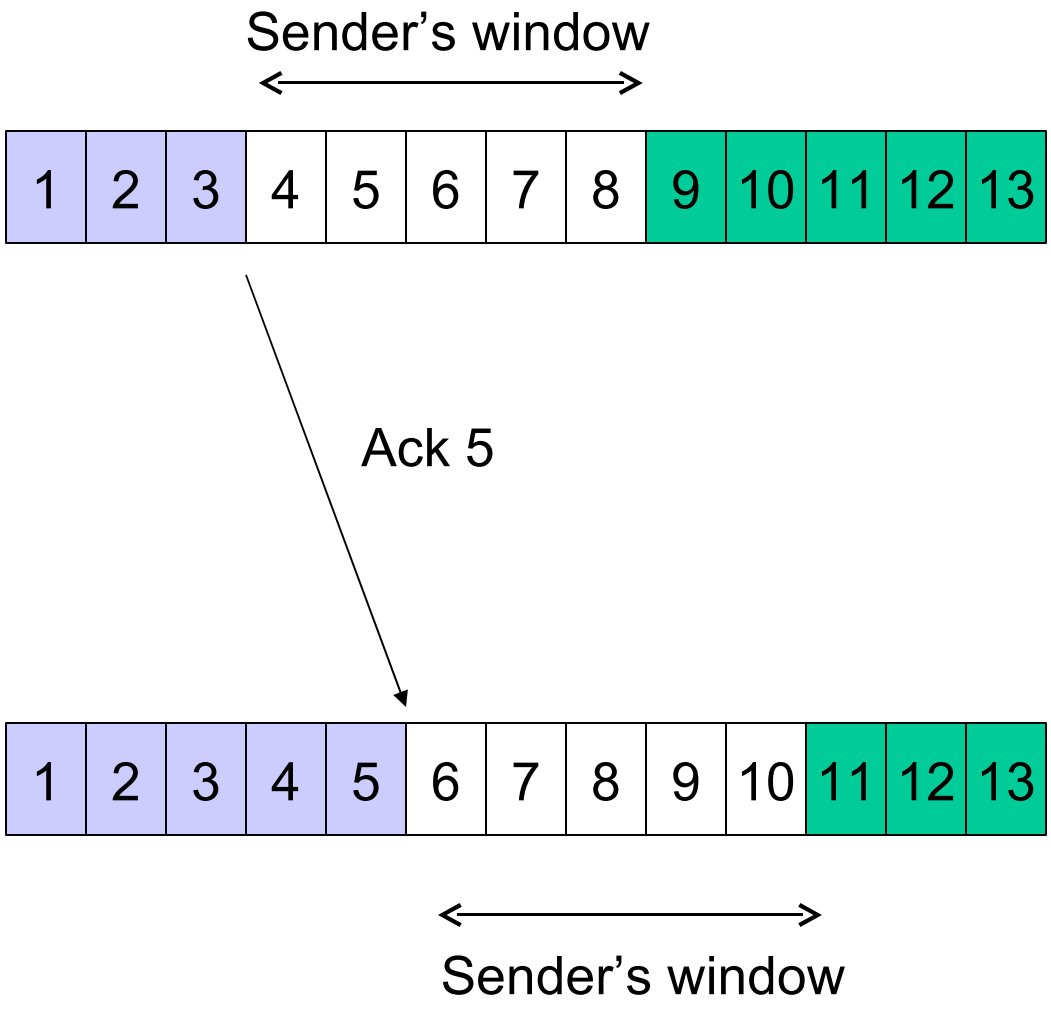
Sliding window protocol

Window size minimum of

- receiver's advertised window - determined by available buffer space at the receiver
- congestion window - determined by the sender, based on feedback from the network



Window Based Flow Control



Window Based Flow Control

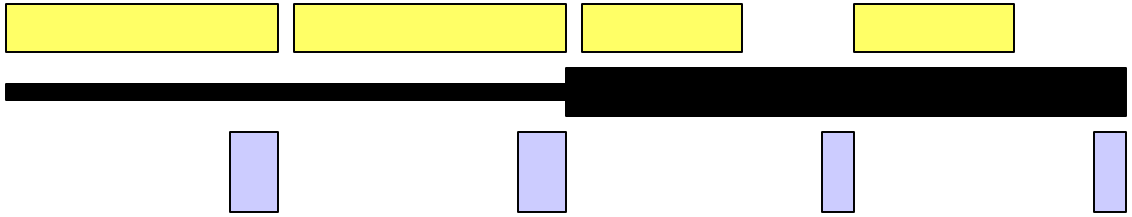
Congestion window size bounds the amount of data that can be sent per round-trip time

Throughput <= W / RTT

Ideal Window Size

Ideal size = delay * bandwidth

delay-bandwidth product



What if window size < delay*bw ?

Inefficiency (wasted bandwidth)

What if > delay*bw ?

Queuing at intermediate routers

- increased RTT due to queuing delays

Potentially, packet loss

How does TCP detect a packet loss?

Retransmission timeout (RTO)

Duplicate acknowledgements

Detecting Packet Loss Using Retransmission Timeout (RTO)

At any time, TCP sender sets retransmission timer for only one packet

If acknowledgement for the timed packet is not received before timer goes off, the packet is assumed to be lost

RTO dynamically calculated

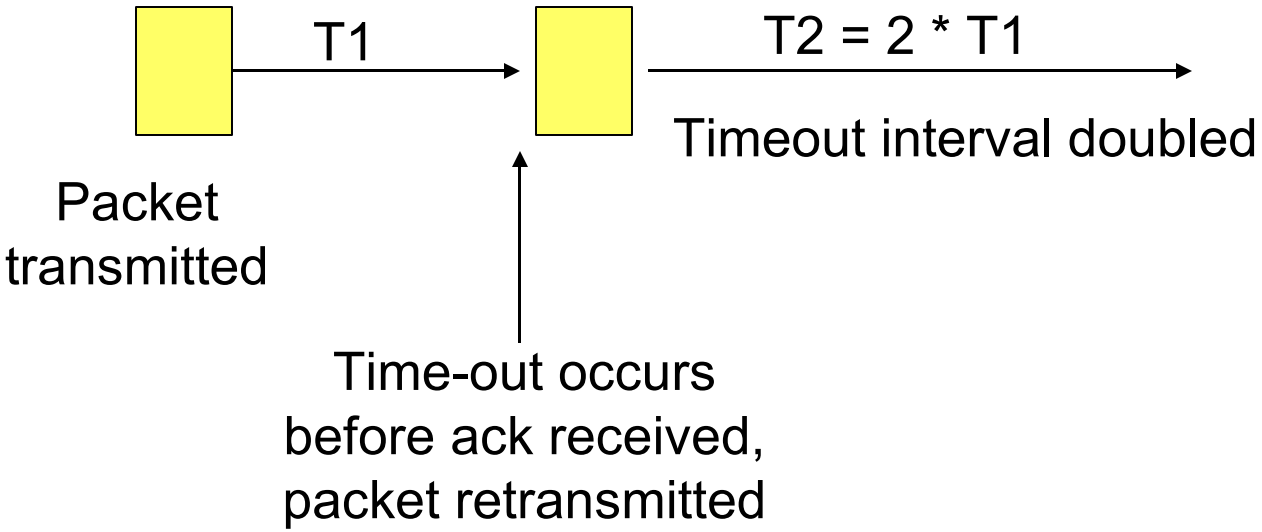
Retransmission Timeout (RTO) calculation

RTO = mean + 4 mean deviation

- Standard deviation σ : σ^2 = average of (sample – mean)^2
- Mean deviation δ = average of |sample – mean|
- Mean deviation easier to calculate than standard deviation
- Mean deviation is more conservative: $\delta >= \sigma$

Exponential Backoff

Double RTO on each timeout



Fast Retransmission

Timeouts can take too long

- how to initiate retransmission sooner?

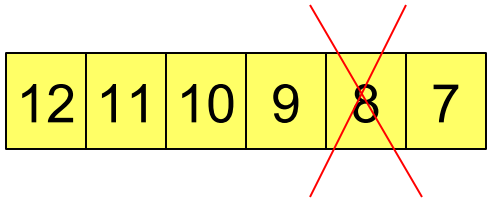
Fast retransmit

Detecting Packet Loss Using Dupacks: Fast Retransmit Mechanism

Dupacks may be generated due to

- packet loss, or
- out-of-order packet delivery

TCP sender assumes that a packet loss has occurred if it receives three **dupacks** consecutively



Receipt of packets 9, 10 and 11 will each generate a *dupack* from the receiver. The sender, on getting these dupacks, will *retransmit* packet 8.

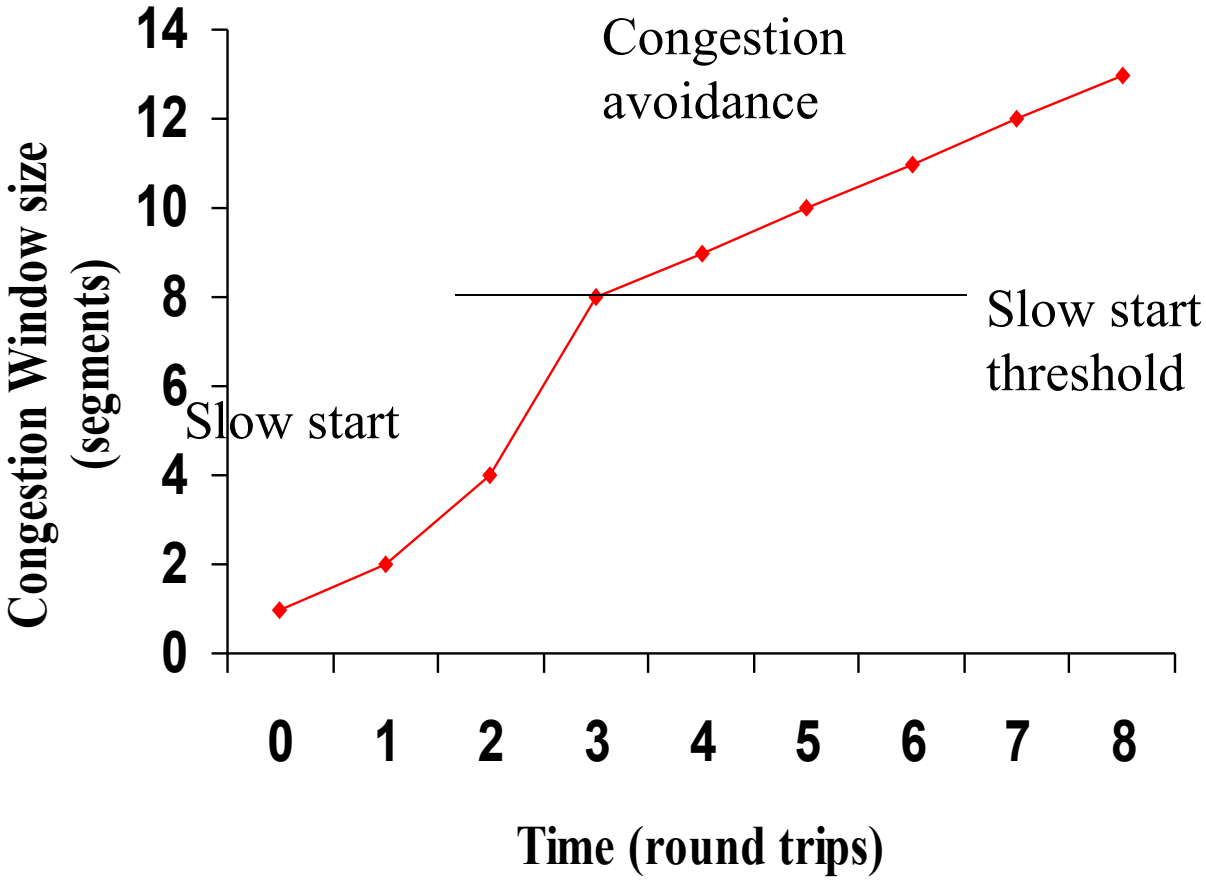
Congestion Avoidance and Control

Slow Start: cwnd grows **exponentially** with time during slow start

When cwnd reaches slow-start threshold, congestion avoidance is performed

Congestion avoidance: cwnd increases **linearly** with time during congestion avoidance

Rate of increase could be lower if sender does not always have data to send



Example assumes that acks are not delayed

Congestion Control

On detecting a packet loss, TCP sender assumes that network congestion has occurred

On detecting packet loss, TCP sender drastically reduces the congestion window

Reducing congestion window reduces amount of data that can be sent per RTT

Congestion Control -- Timeout

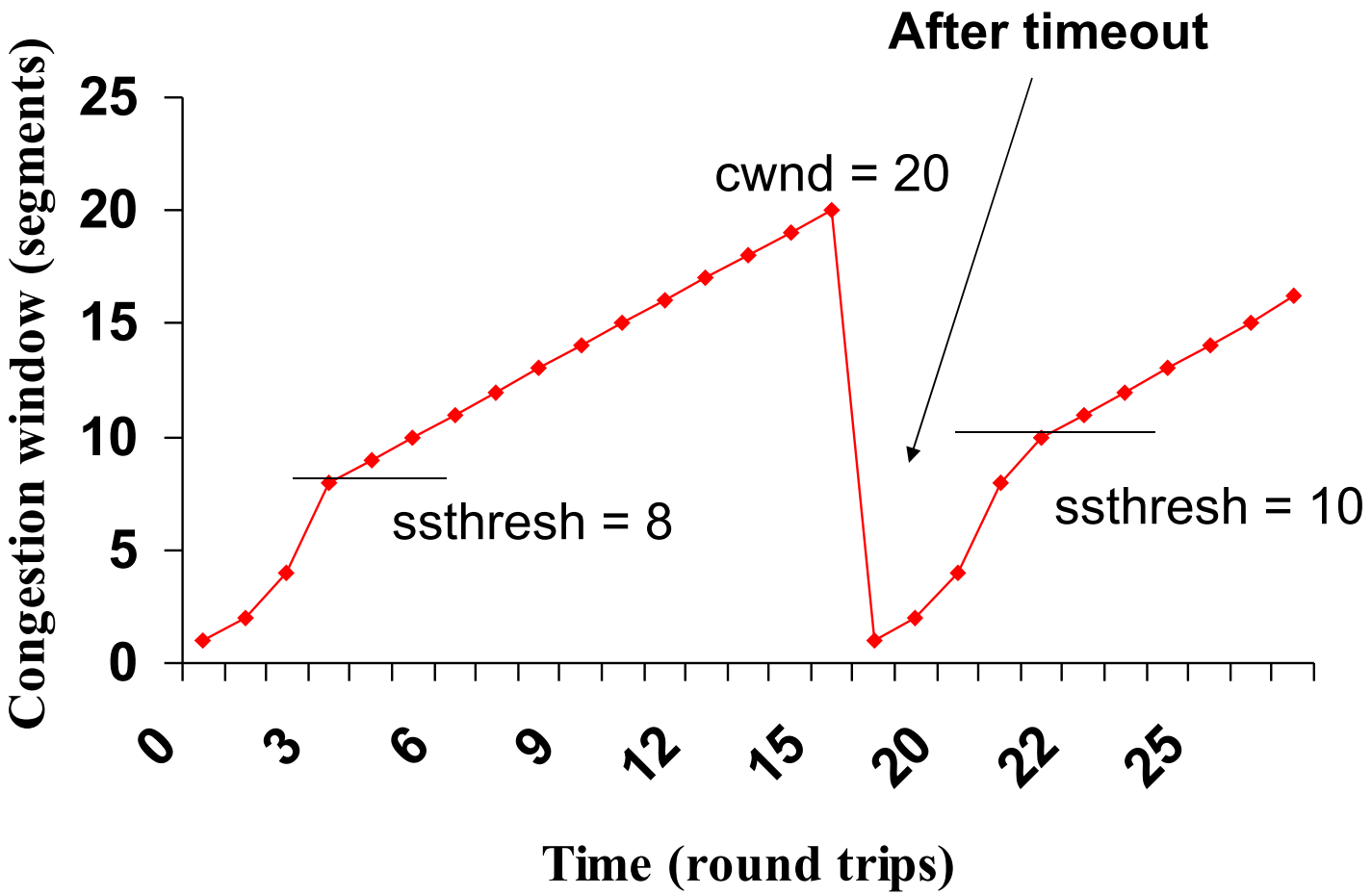
On a timeout, the congestion window is reduced to the initial value of **1 MSS**

The slow start threshold is set to half the window size before packet loss

more precisely,

ssthresh = maximum of min(cwnd,receiver's advertised window)/2 and 2 MSS

Slow start is initiated


After timeout
cwnd = 20
ssthresh = 8
ssthresh = 10
Congestion window (segments)
0
5
10
15
20
25
0
3
6
9
12
15
20
22
25
Time (round trips)

Congestion Control - Fast retransmit

Fast retransmit occurs when multiple (>= 3) dupacks come back

Fast recovery follows fast retransmit

Different from timeout : slow start follows timeout

- timeout occurs when no more packets are getting across
- fast retransmit occurs when a packet is lost, but latter packets get through
- ack clock is still there when fast retransmit occurs
- no need to slow start

Fast Recovery

ssthresh =

min(cwnd, receiver's advertised window)/2
(at least 2 MSS)

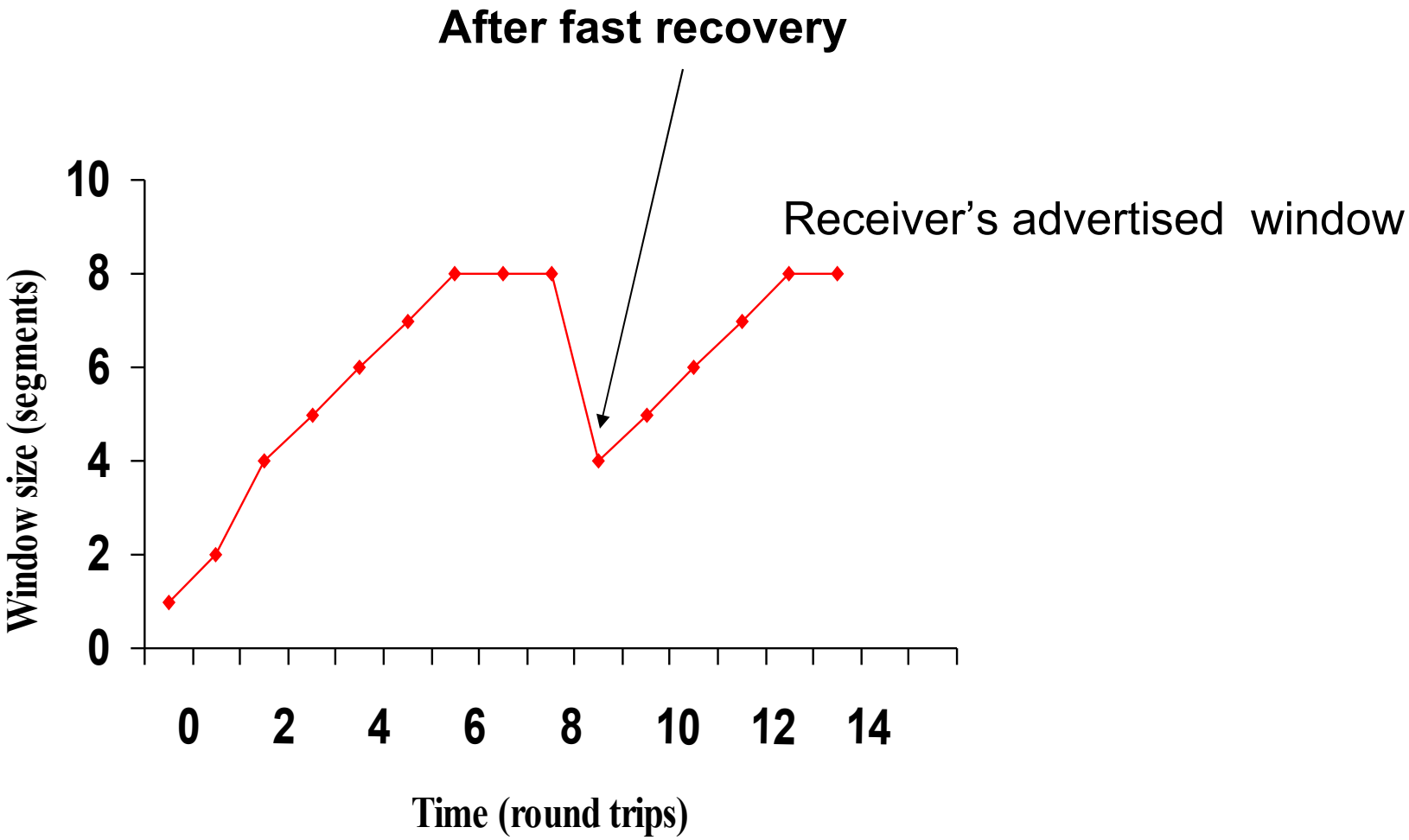
retransmit the missing segment (fast retransmit)

cwnd = ssthresh + number of dupacks

when a new ack comes: cwnd = ssthreh

enter congestion avoidance

Congestion window cut into half



After fast retransmit and fast recovery window size is reduced in half.

TCP Reno

Slow-start

Congestion avoidance

Fast retransmit

Fast recovery

TCP Performance
in
Mobile Ad Hoc Networks

Performance of TCP

Several factors affect TCP performance in MANET:

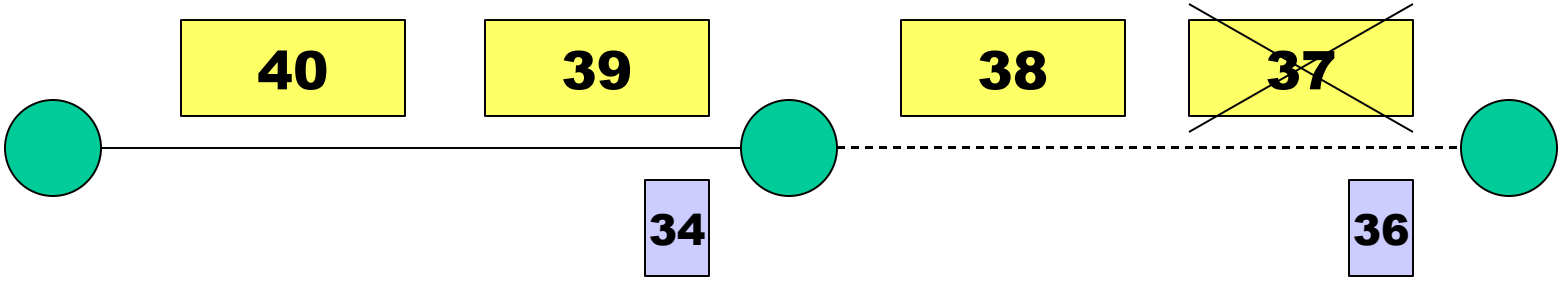
- Wireless transmission errors
- Multi-hop routes on shared wireless medium
  - For instance, adjacent hops typically cannot transmit simultaneously
- Route failures due to mobility

Random Errors

If number of errors is small, they may be corrected by an error correcting code

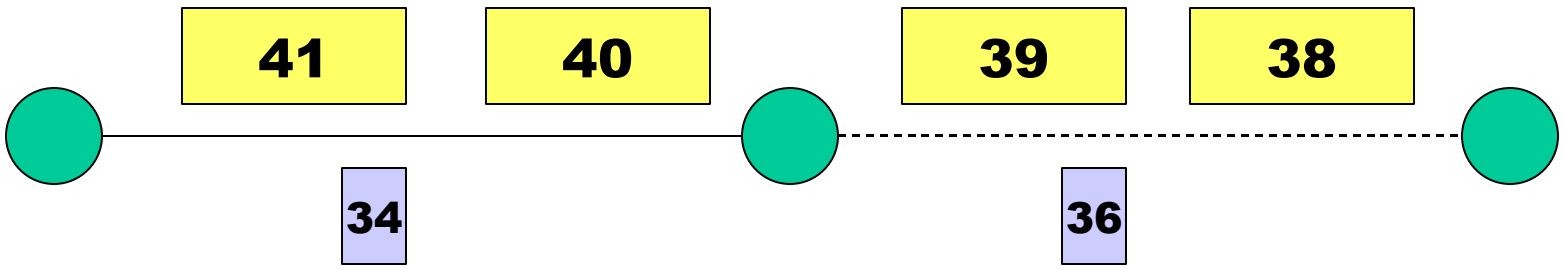
Excessive bit errors result in a packet being discarded, possibly before it reaches the transport layer

Random Errors May Cause Fast Retransmit



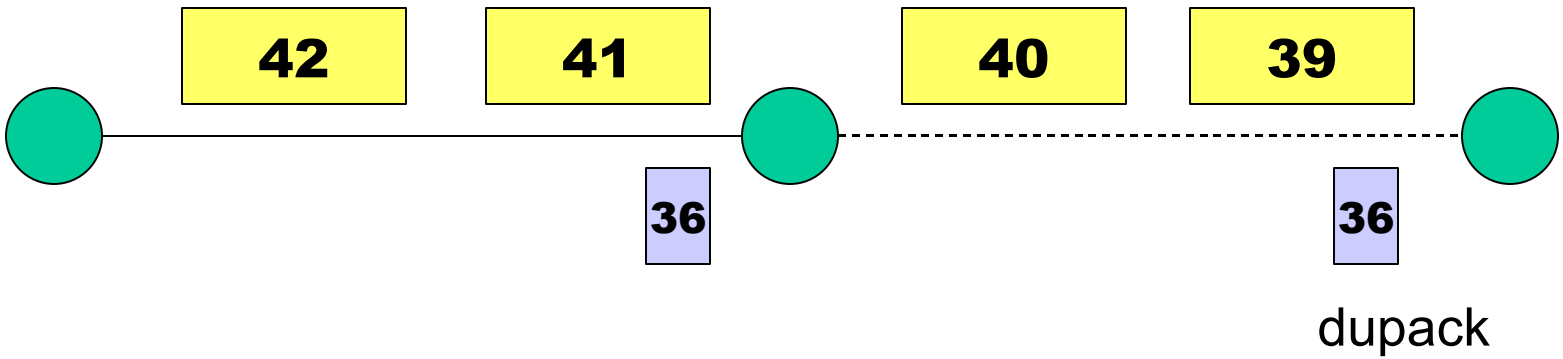
Example assumes delayed ack - every other packet ack'd

Random Errors May Cause Fast Retransmit



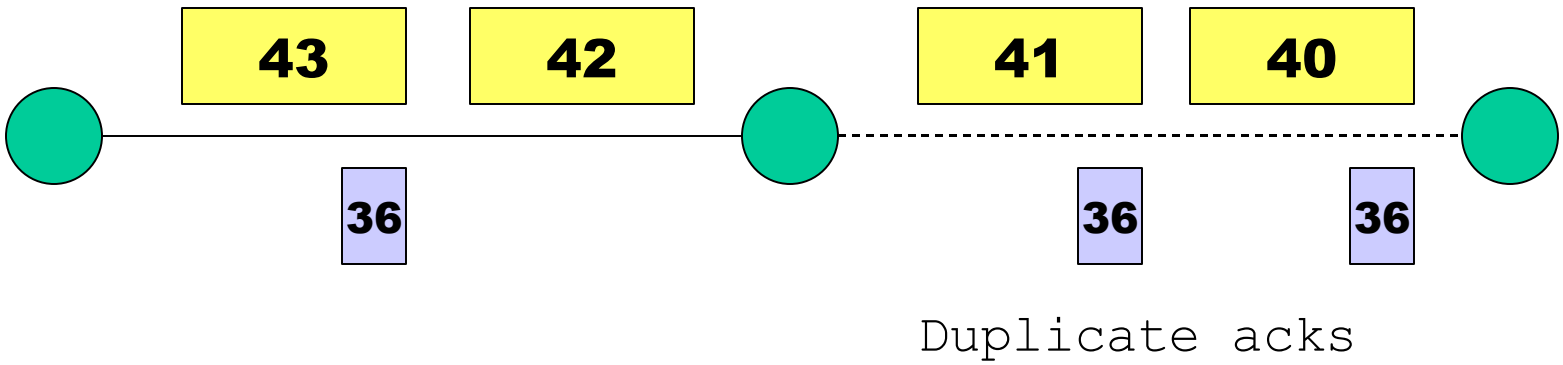
Example assumes delayed ack - every other packet ack'd

Random Errors May Cause Fast Retransmit

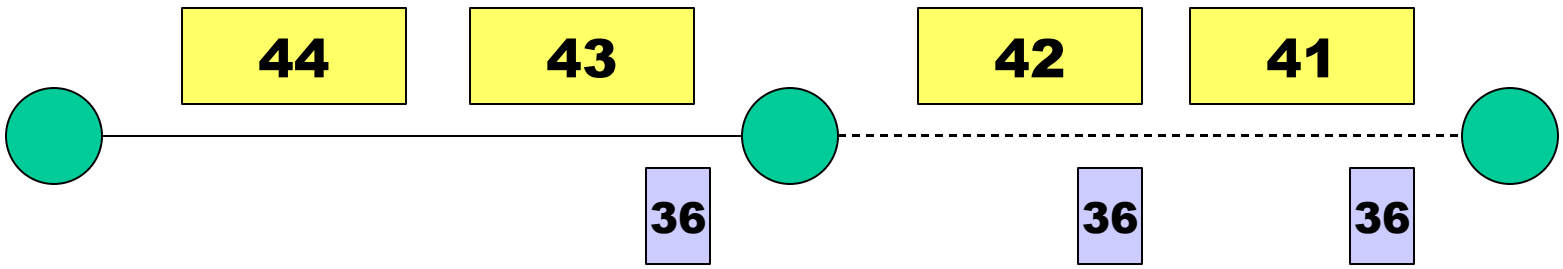


Duplicate acks are not delayed

Random Errors May Cause Fast Retransmit



Random Errors May Cause Fast Retransmit



3 duplicate acks trigger
fast retransmit at sender

Random Errors May Cause Fast Retransmit

Fast retransmit results in

- retransmission of lost packet
- reduction in congestion window

Reducing congestion window in response to errors is unnecessary

Reduction in congestion window reduces the throughput

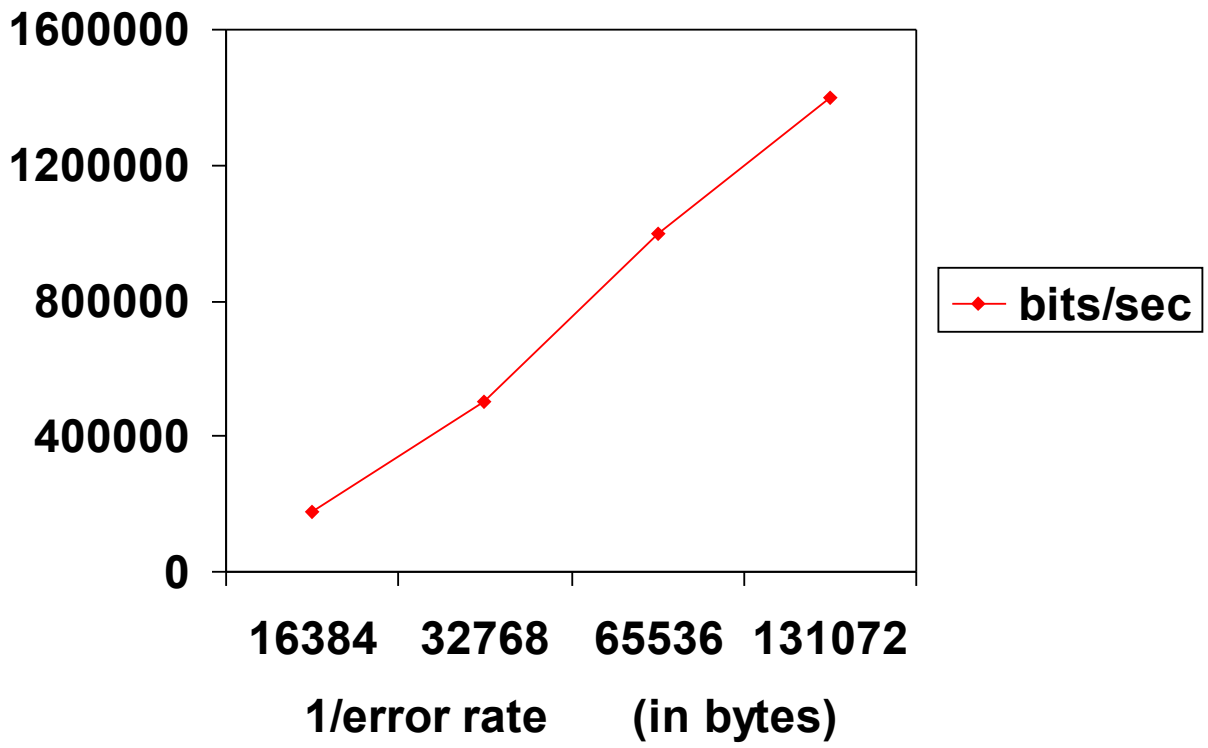
Sometimes Congestion Response May be Appropriate in Response to Errors

On a CDMA channel, errors occur due to interference from other user, and due to noise [Karn99pilc]

- Interference due to other users is an indication of congestion. If such interference causes transmission errors, it is appropriate to reduce congestion window
- If noise causes errors, it is not appropriate to reduce window

When a channel is in a bad state for a long duration, it might be better to let TCP backoff, so that it does not unnecessarily attempt retransmissions while the channel remains in the bad state [Padmanabhan99pilc]

Impact of Random Errors [Vaidya99]



Exponential error model
2 Mbps wireless full duplex link
No congestion losses

Burst Errors May Cause Timeouts

If wireless link remains unavailable for extended duration, a window worth of data may be lost

- driving through a tunnel
- passing a truck

Timeout results in slow start

Slow start reduces congestion window to 1 MSS, reducing throughput

Reduction in window in response to errors unnecessary

Random Errors May Also Cause Timeout

Multiple packet losses in a window can result in timeout when using TCP-Reno (and to a lesser extent when using SACK)

Impact of Transmission Errors

TCP cannot distinguish between packet losses due to congestion and transmission errors

Unnecessarily reduces congestion window

Throughput suffers

This Tutorial

This tutorial does *not* consider techniques to improve TCP performance in presence of transmission errors

Please refer to the *Tutorial on TCP for Wireless and Mobile Hosts* presented by Vaidya at MobiCom 1999, Seattle

> The tutorial slides are presently available from *http://www.cs.tamu.edu/faculty/vaidya/* (follow the link to *Seminars*)

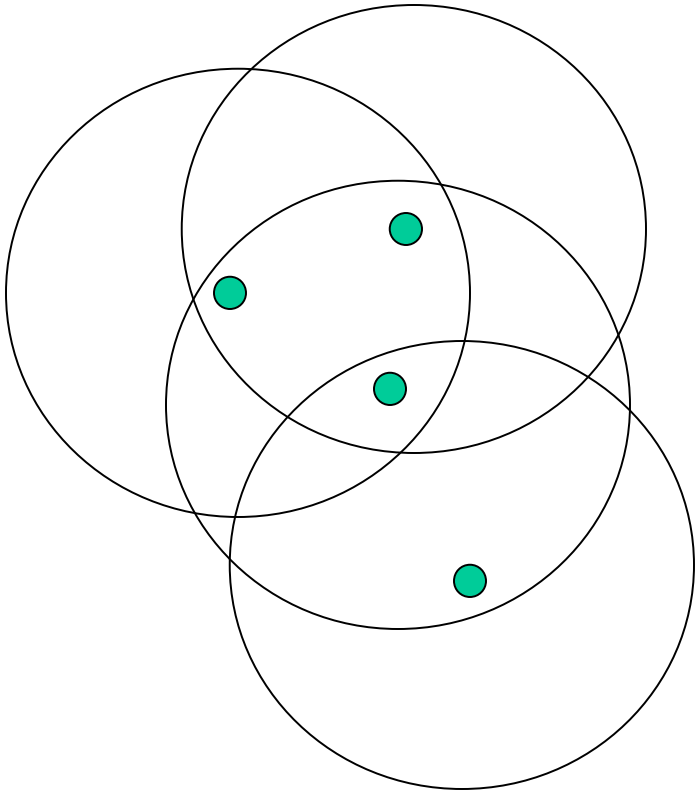
[Montenegro00-RFC2757] discusses related issues

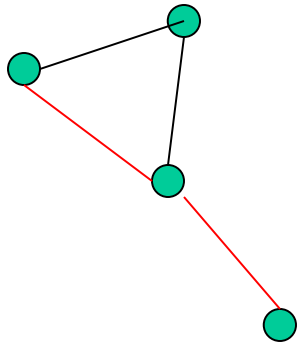
This Tutorial

This tutorial considers impact of multi-hop routes and route failures due to mobility

Mobile Ad Hoc Networks

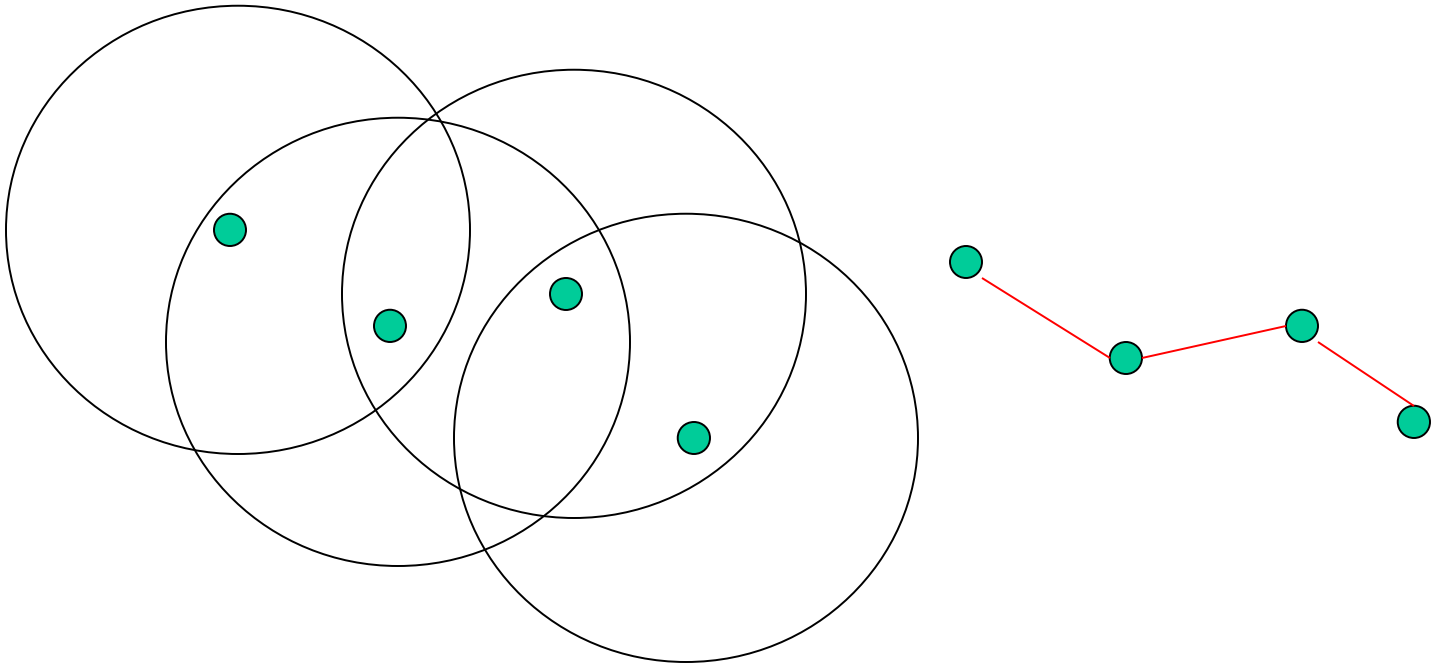
May need to traverse multiple links to reach a destination





Mobile Ad Hoc Networks

Mobility causes route changes

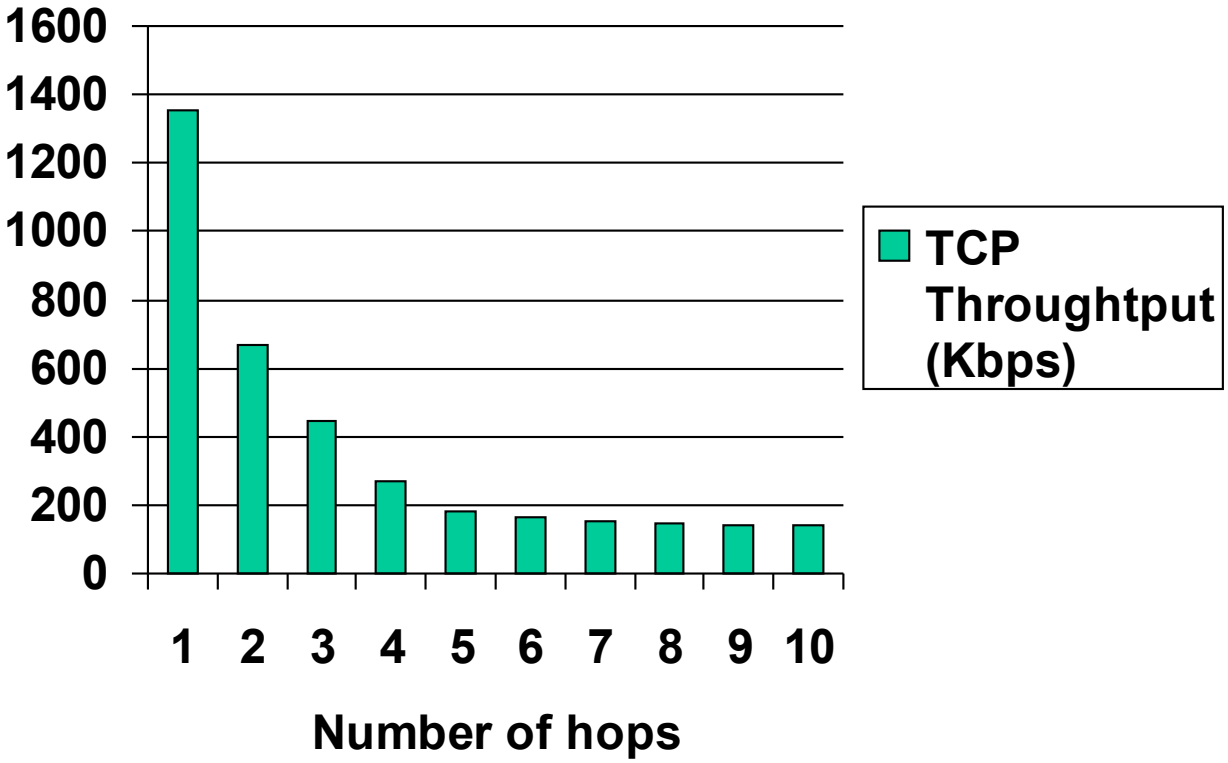


Throughput over Multi-Hop Wireless Paths
[Gerla99]

Connections over multiple hops are at a disadvantage compared to shorter connections, because they have to contend for wireless access at each hop

Impact of Multi-Hop Wireless Paths

[Holland99]



TCP Throughput using 2 Mbps 802.11 MAC

Throughput Degradations with Increasing Number of Hops

Packet transmission can occur on at most one hop among three consecutive hops

- Increasing the number of hops from 1 to 2, 3 results in increased delay, and decreased throughput

Increasing number of hops beyond 3 allows simultaneous transmissions on more than one link, however, degradation continues due to contention between TCP Data and Acks traveling in opposite directions

When number of hops is large enough, the throughput stabilizes due to *effective pipelining*

Ideal Throughput

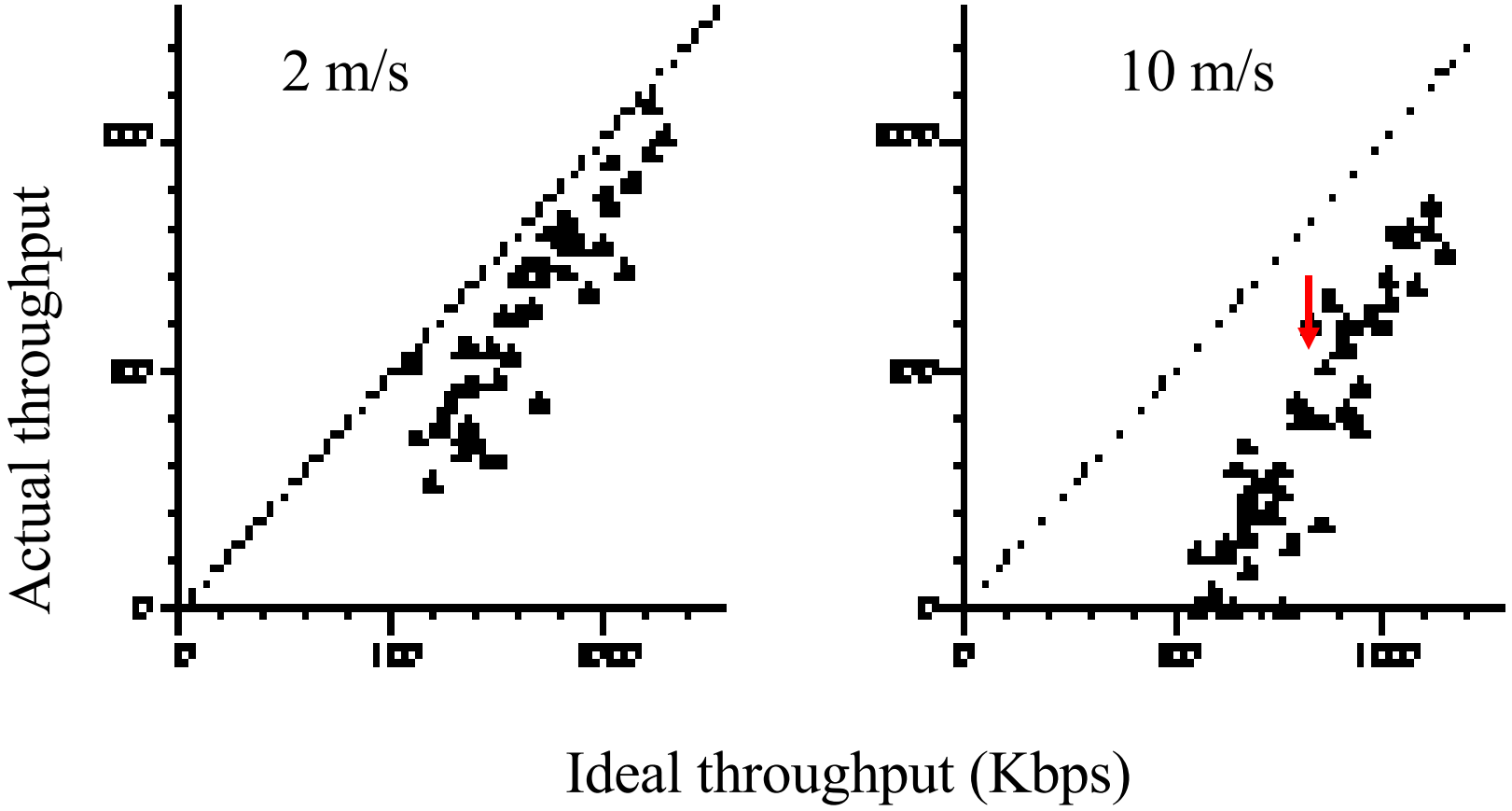
f(i) = fraction of time for which shortest path length between sender and destination is *I*

T(i) = Throughput when path length is *I*

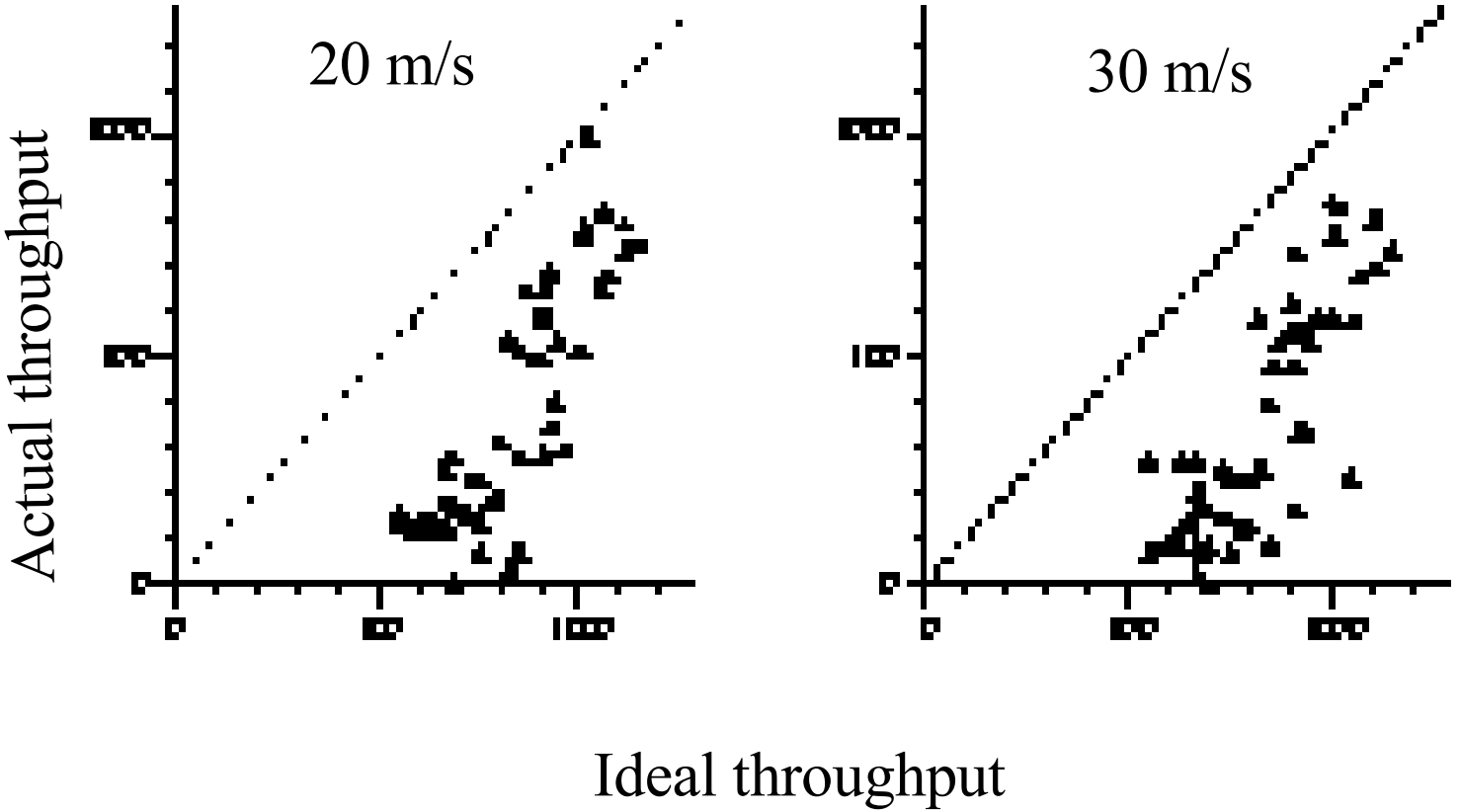
From previous figure

Ideal throughput = Σ *f(i)* * *T(i)*

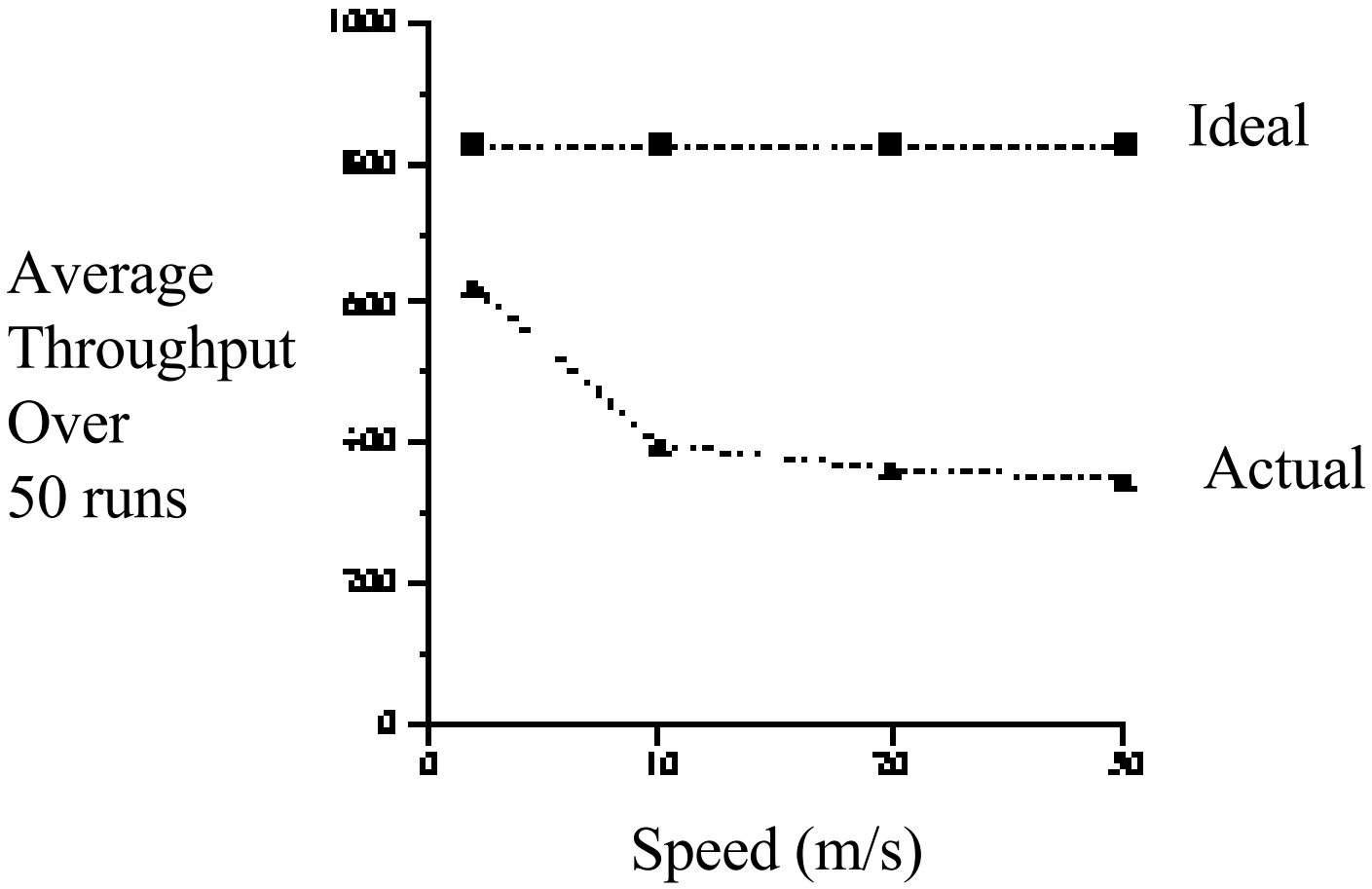
Impact of Mobility
TCP Throughput



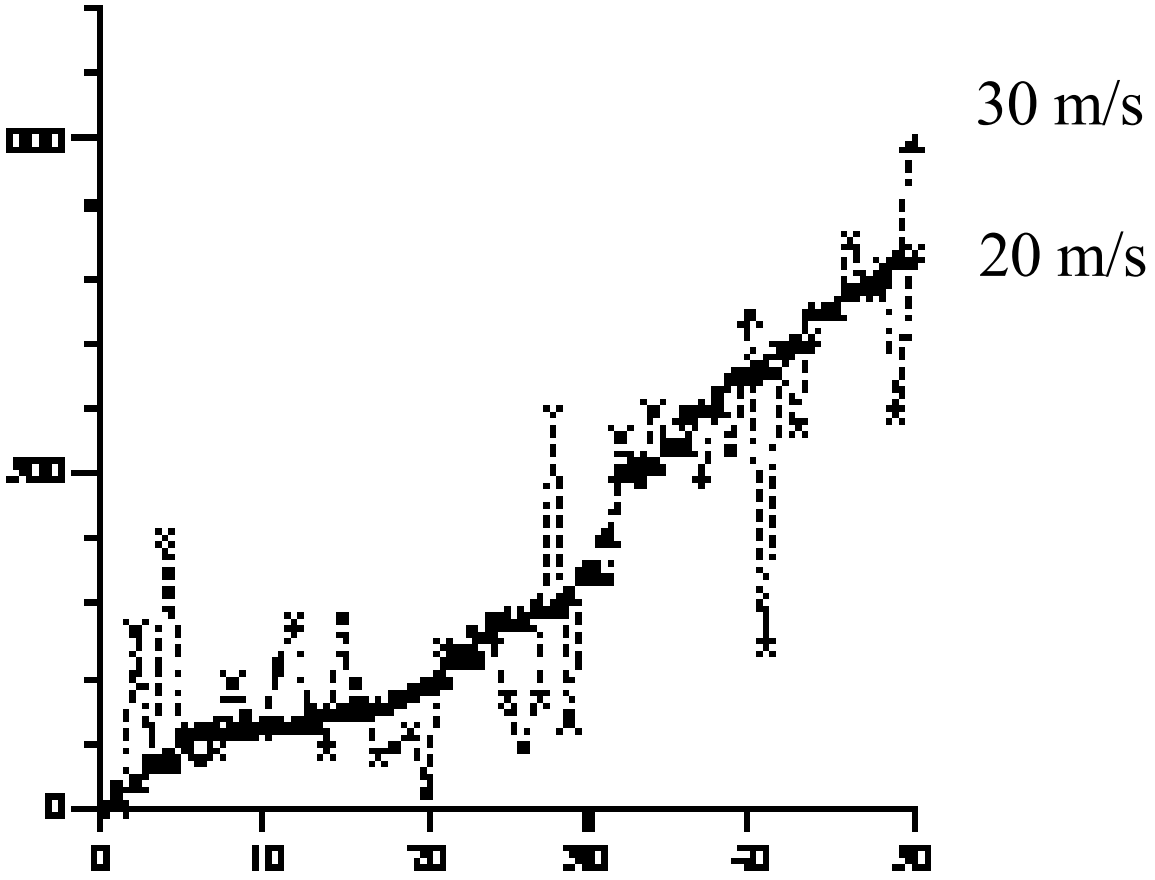
Impact of Mobility



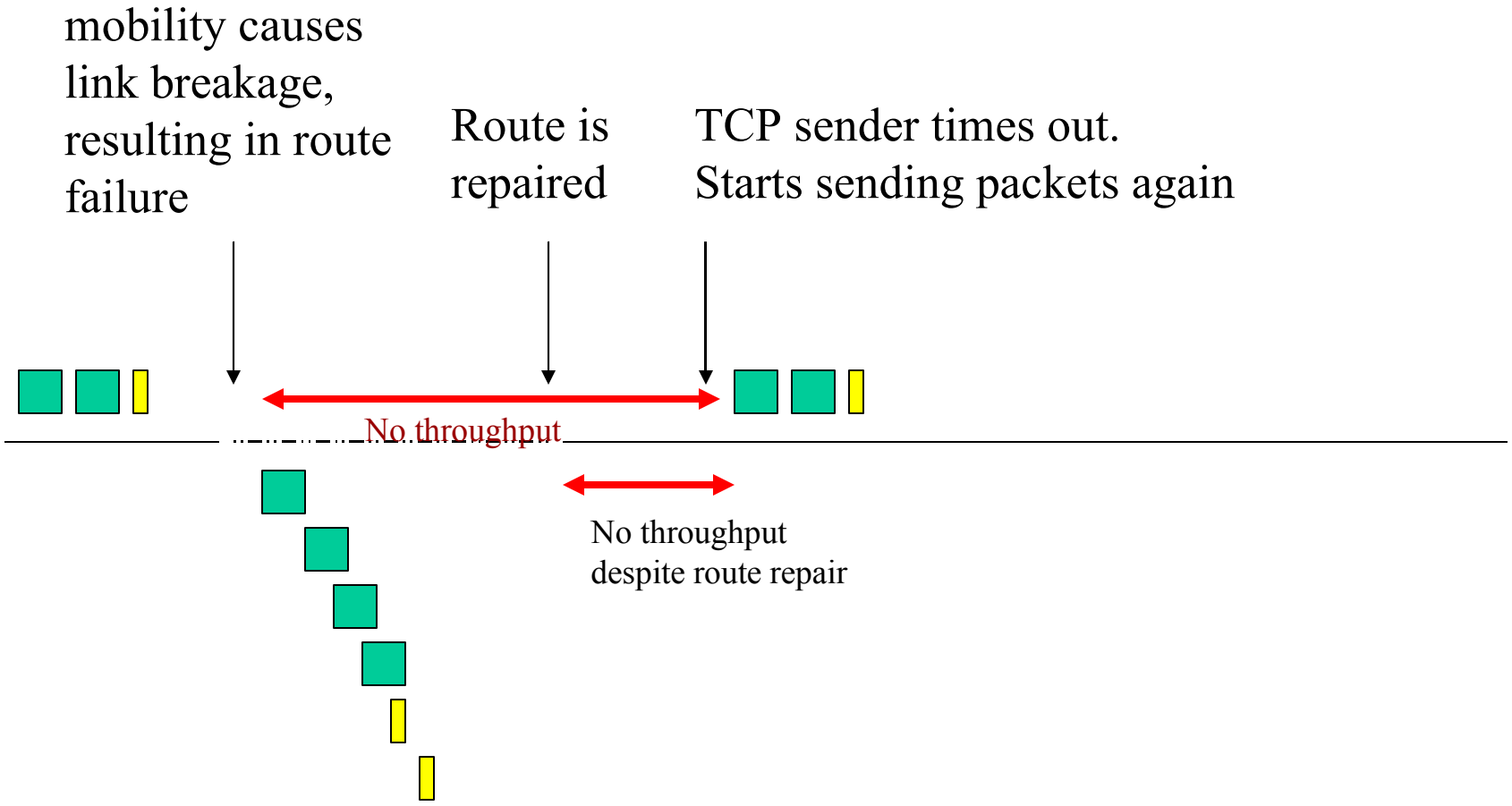
Throughput generally degrades with increasing speed …



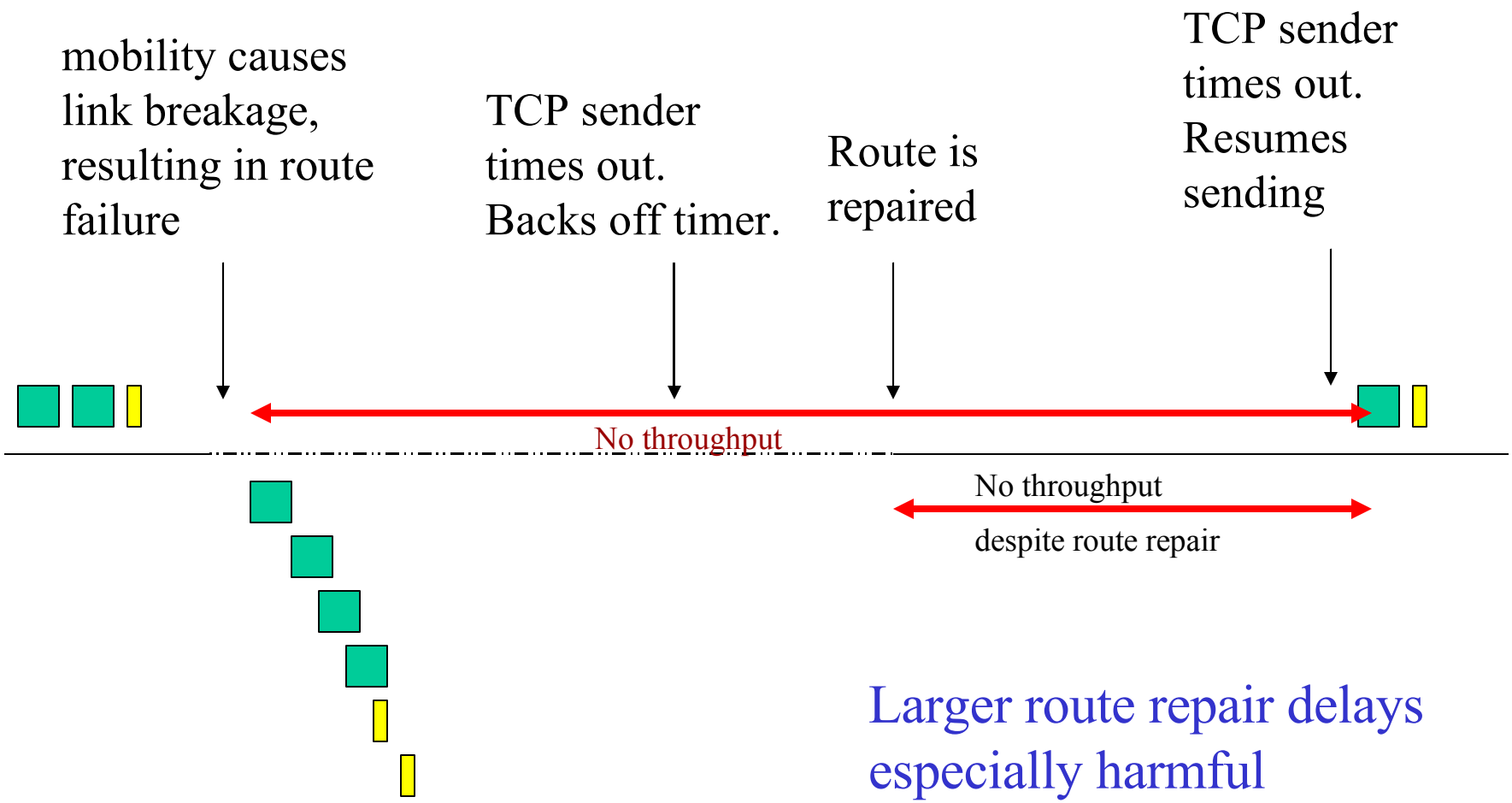
But not always …



Why Does Throughput Degrade?

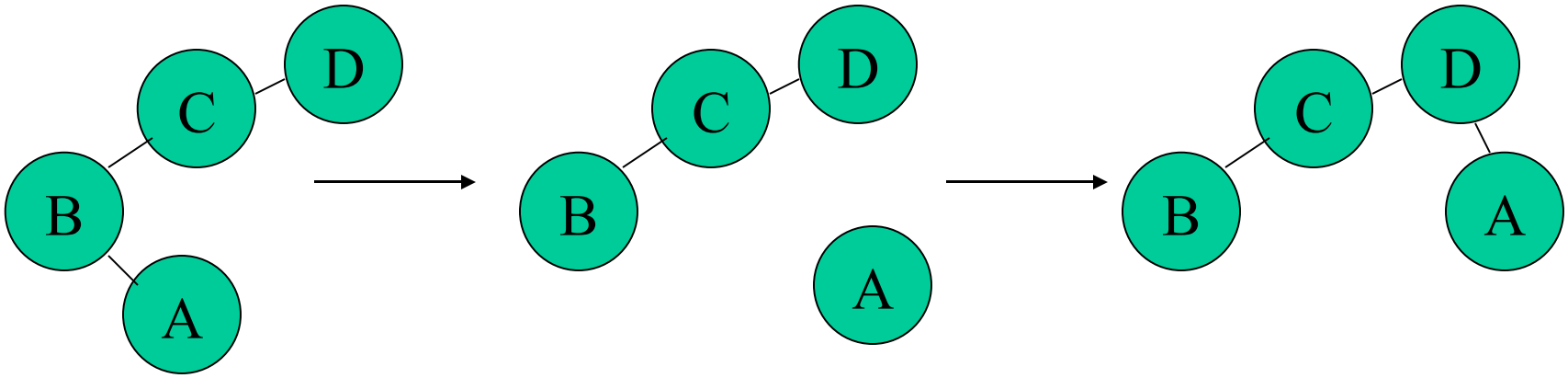


Why Does Throughput Degrade?



Larger route repair delays especially harmful

Why Does Throughput Improve?
Low Speed Scenario



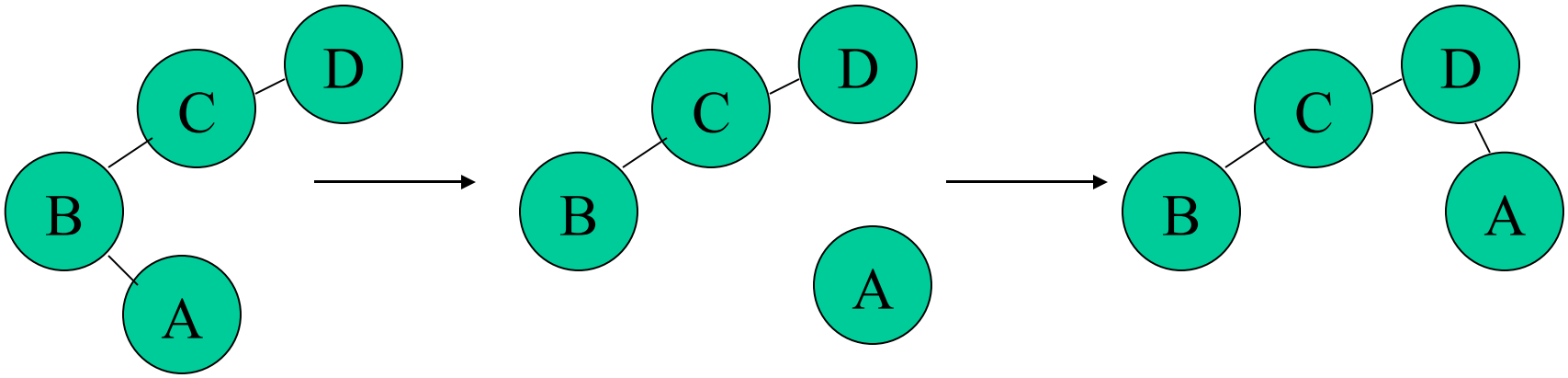
1.5 second route failure

Route from A to D is broken for ~1.5 second.

When TCP sender times after 1 second, route still broken.

TCP times out after another 2 seconds, and only then resumes.

Why Does Throughput Improve?
Higher (double) Speed Scenario



0.75 second route failure

Route from A to D is broken for ~ 0.75 second.

When TCP sender times after 1 second, route is repaired.

Why Does Throughput Improve?
General Principle

The previous two slides show a plausible cause for improved throughput

TCP timeout interval somewhat (not entirely) independent of speed

Network state at higher speed, when timeout occurs, may be more favorable than at lower speed

Network state

- Link/route status
- Route caches
- Congestion

How to Improve Throughput (Bring Closer to Ideal)

Network feedback

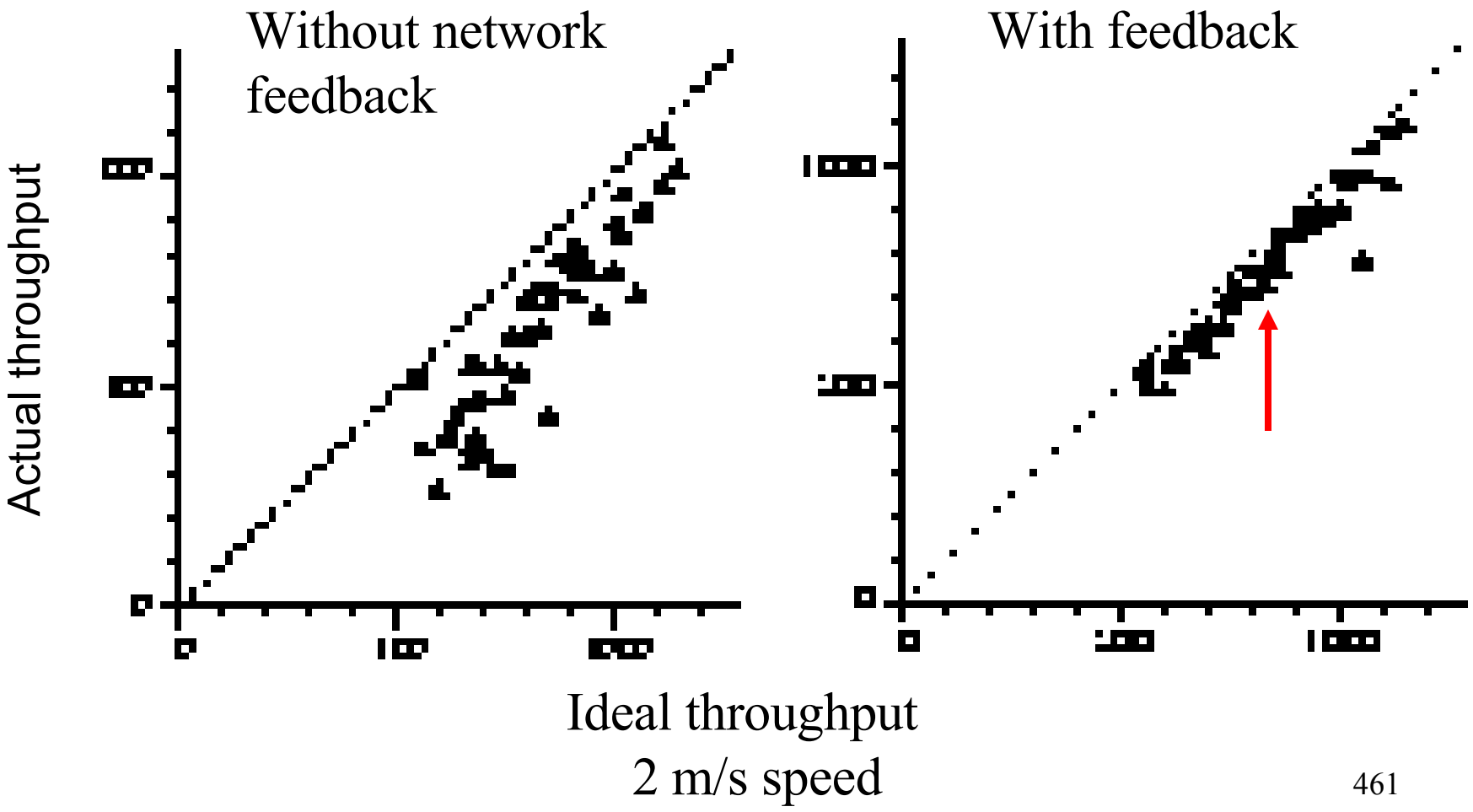
Inform TCP of route failure by explicit message

Let TCP know when route is repaired

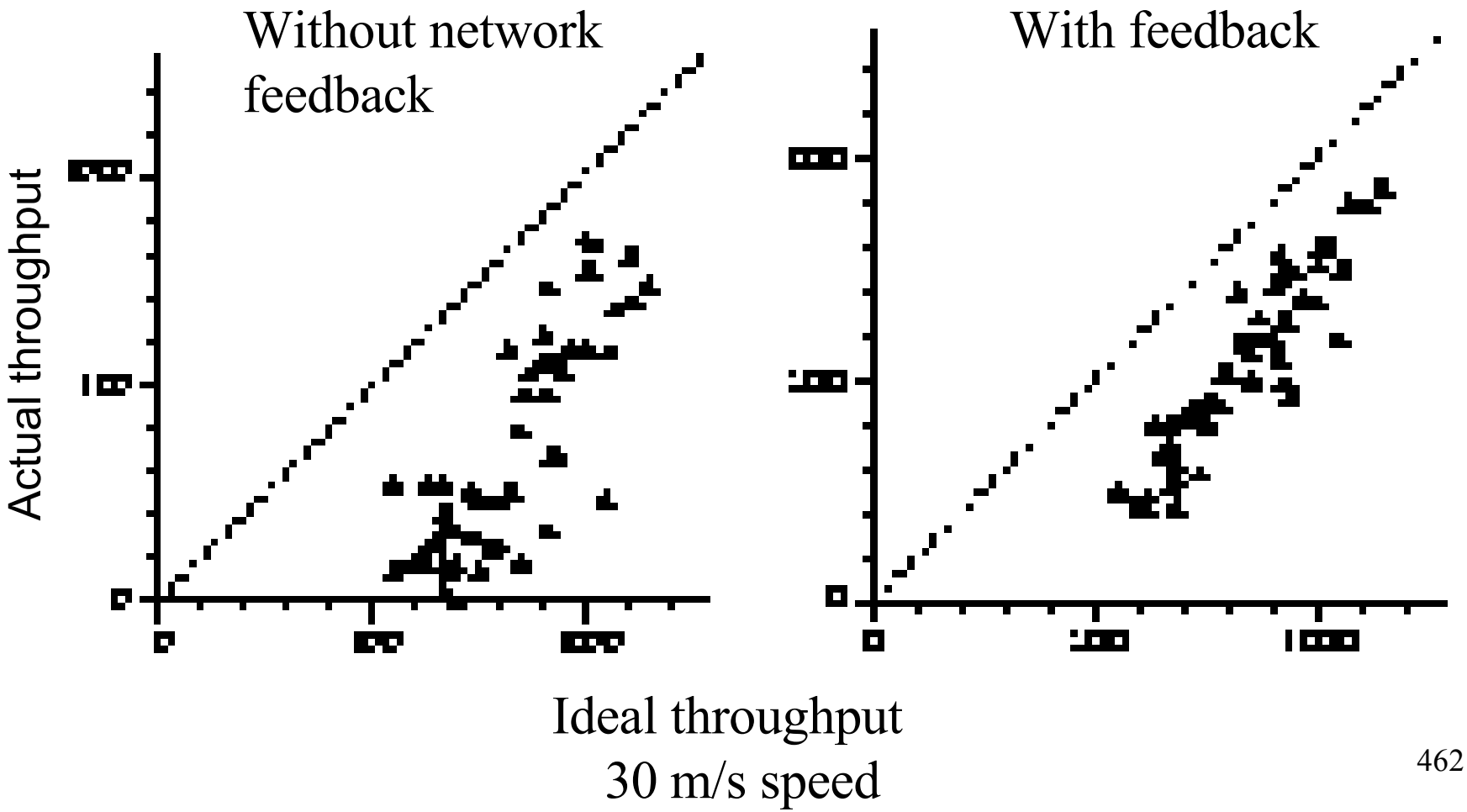
- Probing
- Explicit notification

Reduces repeated TCP timeouts and backoff

Performance Improvement

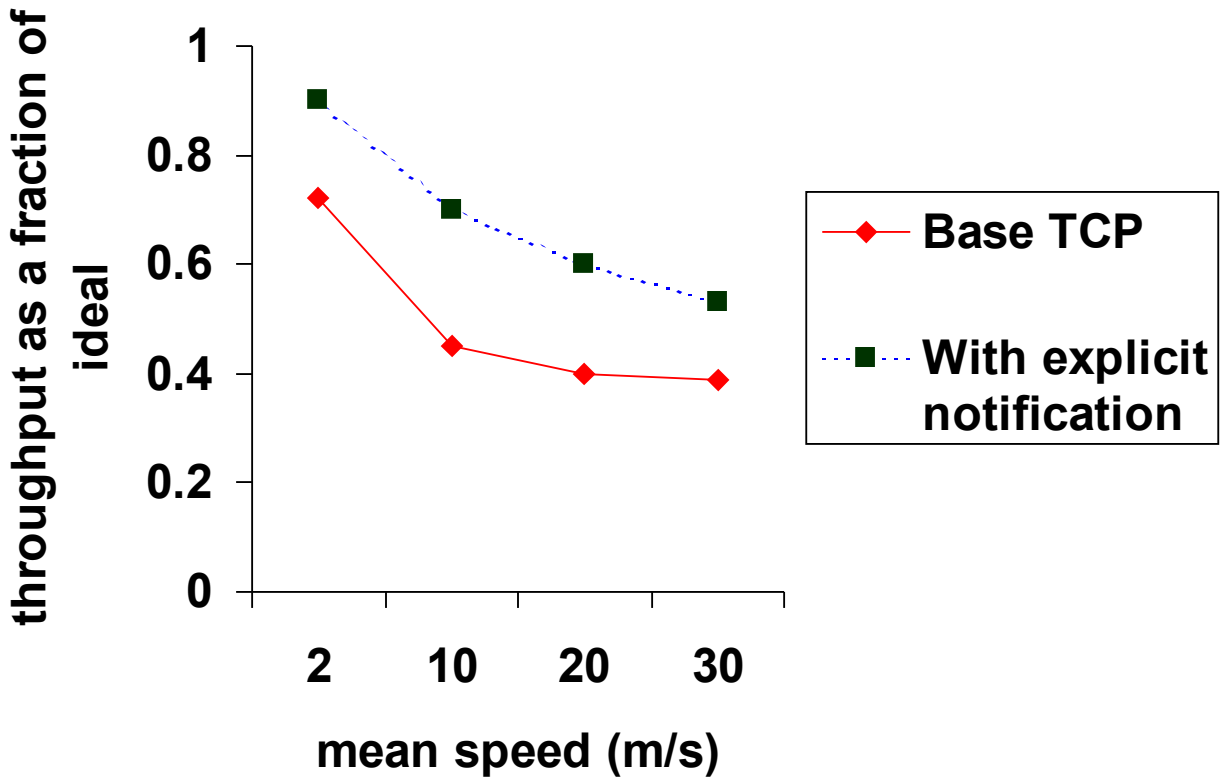


Performance Improvement



Performance with Explicit Notification

[Holland99]



Issues
Network Feedback

Network knows best (why packets are lost)

+ Network feedback beneficial
- Need to modify transport & network layer to receive/send feedback

Need mechanisms for information exchange between layers

[Holland99] discusses alternatives for providing feedback (when routes break and repair)

[Chandran98] also presents a feedback scheme

Impact of Caching

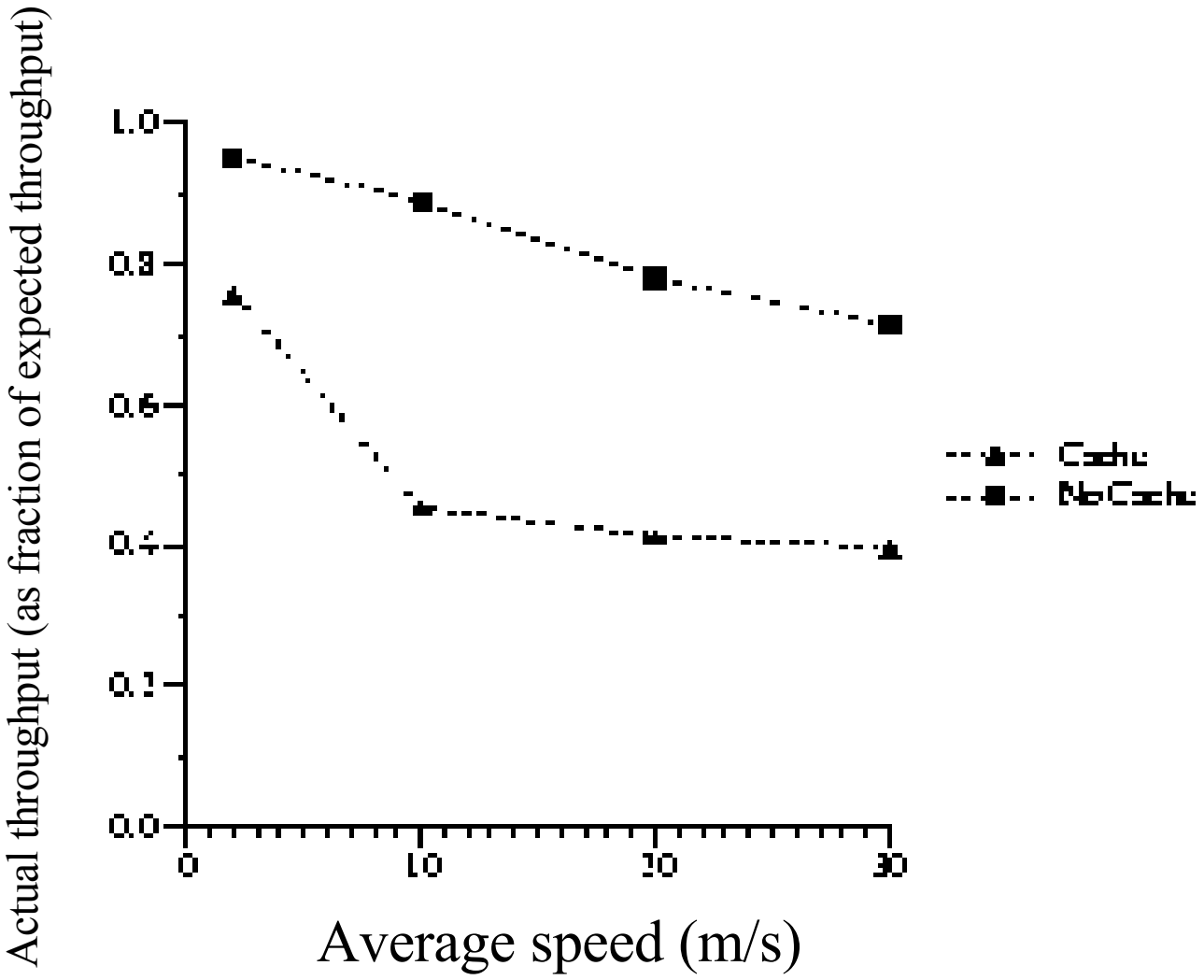
Route caching has been suggested as a mechanism to reduce route discovery overhead [Broch98]

Each node may cache one or more routes to a given destination

When a route from S to D is detected as broken, node S may:

- Use another cached route from local cache, or
- Obtain a new route using cached route at another node

To Cache or Not to Cache



Why Performance Degrades With Caching

When a route is broken, route discovery returns a cached route from local cache or from a nearby node

After a time-out, TCP sender transmits a packet on the new route.

However, the cached route has also broken after it was cached

Another route discovery, and TCP time-out interval

Process repeats until a good route is found

Issues
To Cache or Not to Cache

Caching can result in faster route "repair"

Faster does not necessarily mean correct

If incorrect repairs occur often enough, caching performs poorly

Need mechanisms for determining when cached routes are stale

Caching and TCP performance

Caching can reduce overhead of route discovery even if cache accuracy is not very high

But if cache accuracy is not high enough, gains in routing overhead may be offset by loss of TCP performance due to multiple time-outs

TCP Performance

Two factors result in degraded throughput in presence of mobility:

- Loss of throughput that occurs while waiting for TCP sender to timeout (as seen earlier)
  - This factor can be mitigated by using explicit notifications and better route caching mechanisms
- Poor choice of congestion window and RTO values after a new route has been found
  - How to choose *cwnd* and *RTO* after a route change?

Issues
Window Size After Route Repair

Same as before route break: may be too optimistic

Same as startup: may be too conservative

Better be conservative than overly optimistic

- Reset window to small value after route repair
- Let TCP figure out the suitable window size
- Impact low on paths with small delay-bw product

Issues
RTO After Route Repair

Same as before route break

- If new route long, this RTO may be too small, leading to timeouts

Same as TCP start-up (6 second)

- May be too large
- May result in slow response to next packet loss

Another plausible approach: new RTO = function of old RTO, old route length, and new route length

- Example: new RTO = old RTO * new route length / old route length
- Not evaluated yet
- Pitfall: RTT is not just a function of route length

Out-of-Order Packet Delivery

Out-of-order (OOO) delivery may occur due to:

- Route changes
- Link layer retransmissions schemes that deliver OOO

Significantly OOO delivery confuses TCP, triggering fast retransmit

Potential solutions:

- Deterministically prefer one route over others, even if multiple routes are known
- Reduce OOO delivery by re-ordering received packets
  - can result in unnecessary delay in presence of packet loss
- Turn off fast retransmit
  - can result in poor performance in presence of congestion

Impact of Acknowledgements

TCP Acks (and link layer acks) share the wireless bandwidth with TCP data packets

Data and Acks travel in opposite directions

- In addition to bandwidth usage, acks require additional receive-send turnarounds, which also incur time penalty

- To reduce frequency of send-receive turnaround and contention between acks and data

Impact of Acks: Mitigation [Balakrishnan97]

Piggybacking link layer acks with data

Sending fewer TCP acks - ack every d-th packet (d may be chosen dynamically)

- but need to use rate control at sender to reduce burstiness (for large d)

Ack filtering - Gateway may drop an older ack in the queue, if a new ack arrives

reduces number of acks that need to be delivered to the sender

Security Issues

Caveat

Much of security-related stuff is mostly beyond my expertise

So coverage of this topic is **very** limited

Security Issues in Mobile Ad Hoc Networks

Not much work in this area as yet

Many of the security issues are same as those in traditional wired networks and cellular wireless

What's new ?

What's New ?

Wireless medium is easy to snoop on

Due to ad hoc connectivity and mobility, it is hard to guarantee access to any particular node (for instance, to obtain a secret key)

Easier for trouble-makers to insert themselves into a mobile ad hoc network (as compared to a wired network)

Resurrecting Duckling [Stajano99]

Battery exhaustion threat: A malicious node may interact with a mobile node often with the goal of draining the mobile node's battery

Authenticity: Who can a node talk to safely?

- Resurrecting duckling: Analogy based on a duckling and its mother. Apparently, a duckling assumes that the first object it hears is the mother
- A mobile device will trust first device which sends a secret key

Secure Routing [Zhou99]

Attackers may inject erroneous routing information

By doing so, an attacker may be able to divert network traffic, or make routing inefficient

[Zhou] suggests use of digital signatures to protect routing information and data both

Such schemes need a Certification Authority to manage the private-public keys

Secure Routing

Establishing a Certification Authority (CA) difficult in a mobile ad hoc network, since the authority may not be reachable from all nodes at all times

[Zhou] suggests distributing the CA function over multiple nodes

MANET Authentication Architecture
[Jacobs99ietf-id]

Digital signatures to authenticate a message

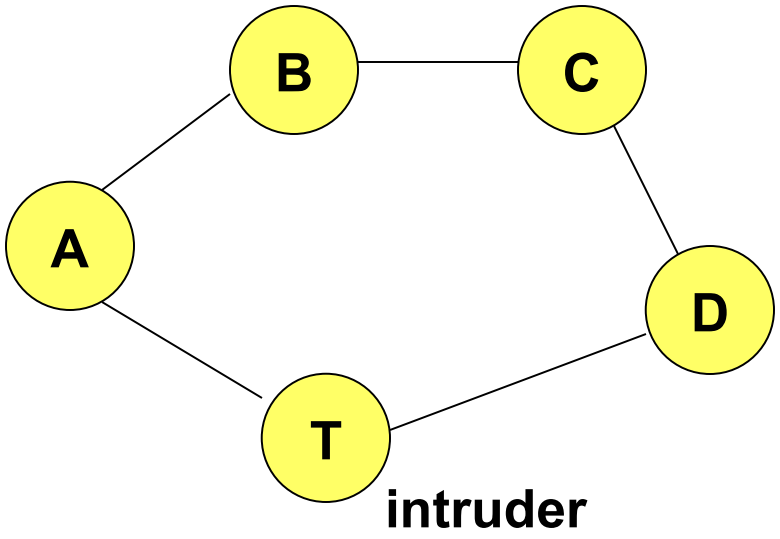
Key distribution via certificates

Need access to a certification authority

[Jacobs99ietf-id] specifies message formats to be used to carry signature, etc.

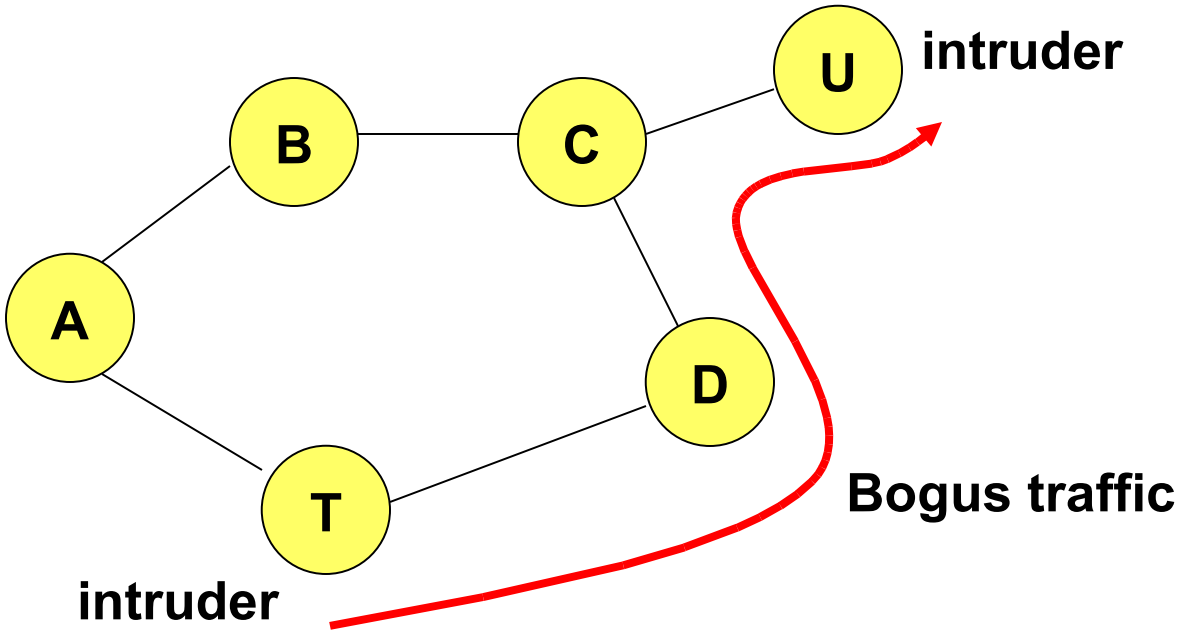
Techniques for Intrusion-Resistant Ad Hoc Routing Algorithms (TIARA) [Ramanujan00Milcom]

Flow disruption attack: Intruder (or compromised) node T may delay/drop/corrupt all data passing through, but leave all routing traffic unmodified



Techniques for Intrusion-Resistant Ad Hoc Routing Algorithms (TIARA) [Ramanujan00Milcom]

Resource Depletion Attack: Intruders may send data with the objective of congesting a network or depleting batteries



Intrusion Detection [Zhang00Mobicom]

Detection of abnormal routing table updates

- Uses “training” data to determine characteristics of normal routing table updates (such as rate of change of routing info)
- Efficacy of this approach is not evaluated, and is debatable

Similar abnormal behavior may be detected at other protocol layers

- For instance, at the MAC layer, *normal* behavior may be characterized for access patterns by various hosts
- Abnormal behavior may indicate intrusion

Solutions proposed in [Zhang00Mobicom] are preliminary, not enough detail provided

Preventing Traffic Analysis
[Jiang00iaas,Jiang00tech]

Even with encryption, an eavesdropper may be able to identify the traffic pattern in the network

Traffic patterns can give away information about the *mode* of operation

- Attack versus retreat

Traffic analysis can be prevented by presenting "constant" traffic pattern independent of the underlying operational mode

- May need insertion of dummy traffic to achieve this

Packet Purse Model [Byttayn00MobiHoc]

Cost-based approach for motivating collaboration between mobile nodes

The packet purse model assigns a cost to each packet transfer

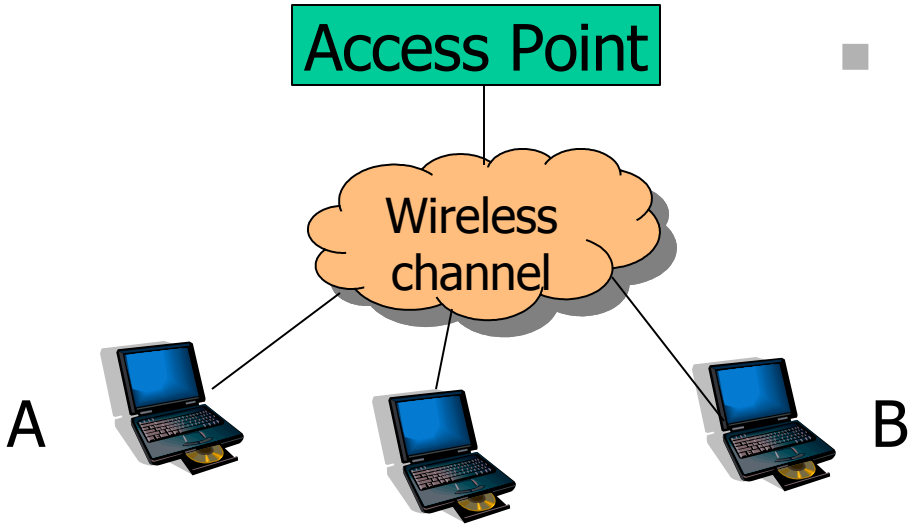
- Link-level recipient of a packet pays the link-level sender for the service
- Virtual money (“beans”) used for this purpose

Security issues:

- How to ensure that some node does not sale the same packet to too many people to make money ?
- How to ensure that each receiver indeed has money to pay for service?

MAC Layer Misbehavior

Selfish Misbehavior to Improve Performance



- Nodes are required to follow Medium Access Control (MAC) rules

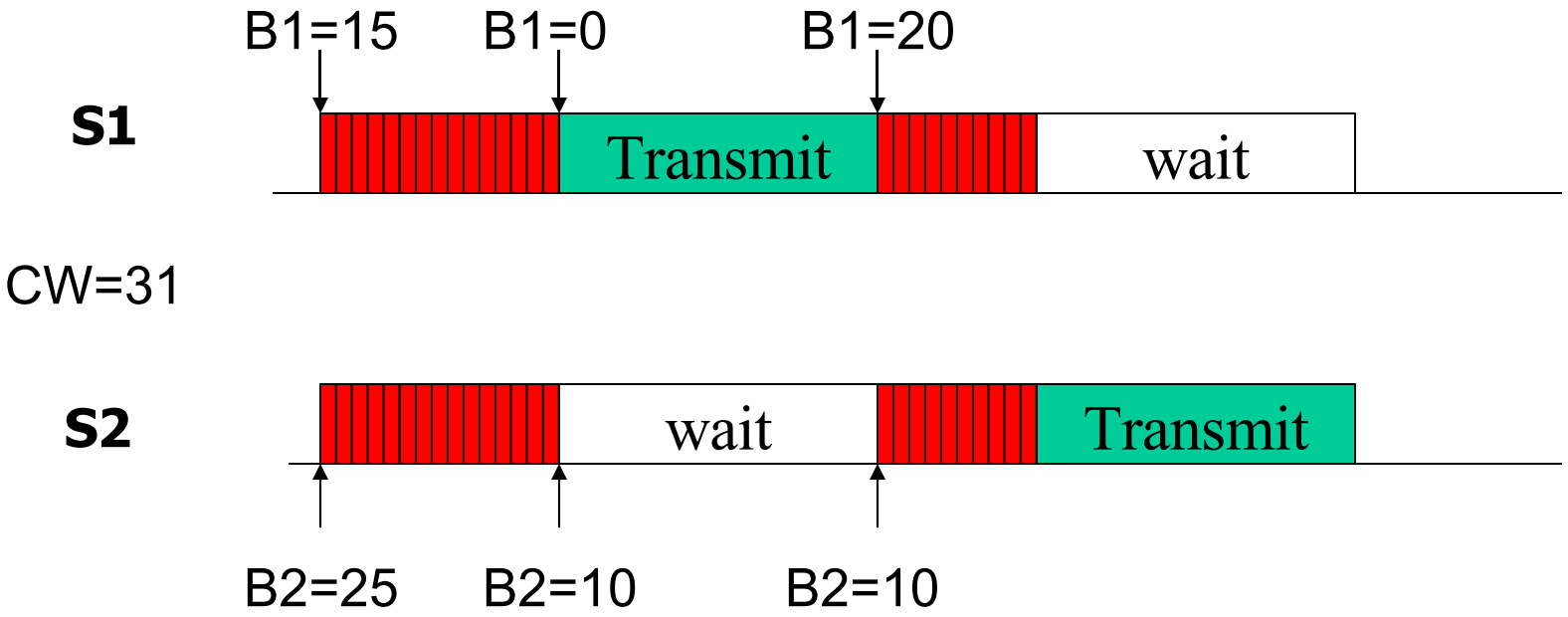
Misbehaving nodes may violate MAC rules

Backoff Example

Choose backoff value B in range [0,CW]

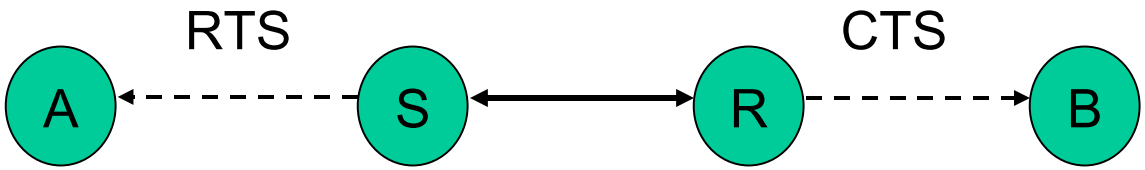
CW is the Contention Window

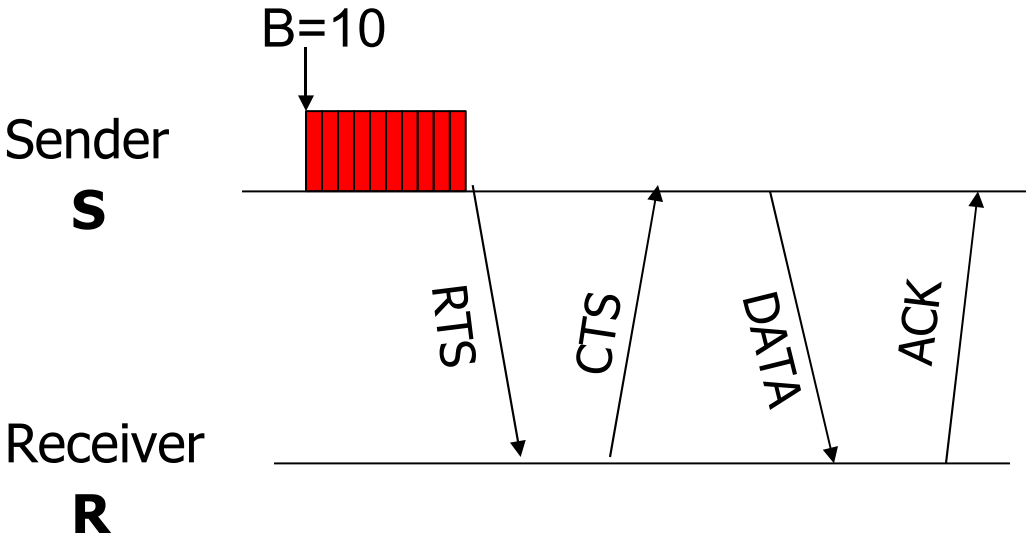
Count down backoff by 1 every idle slot



Data Transmission

Reserve channel with RTS/CTS exchange

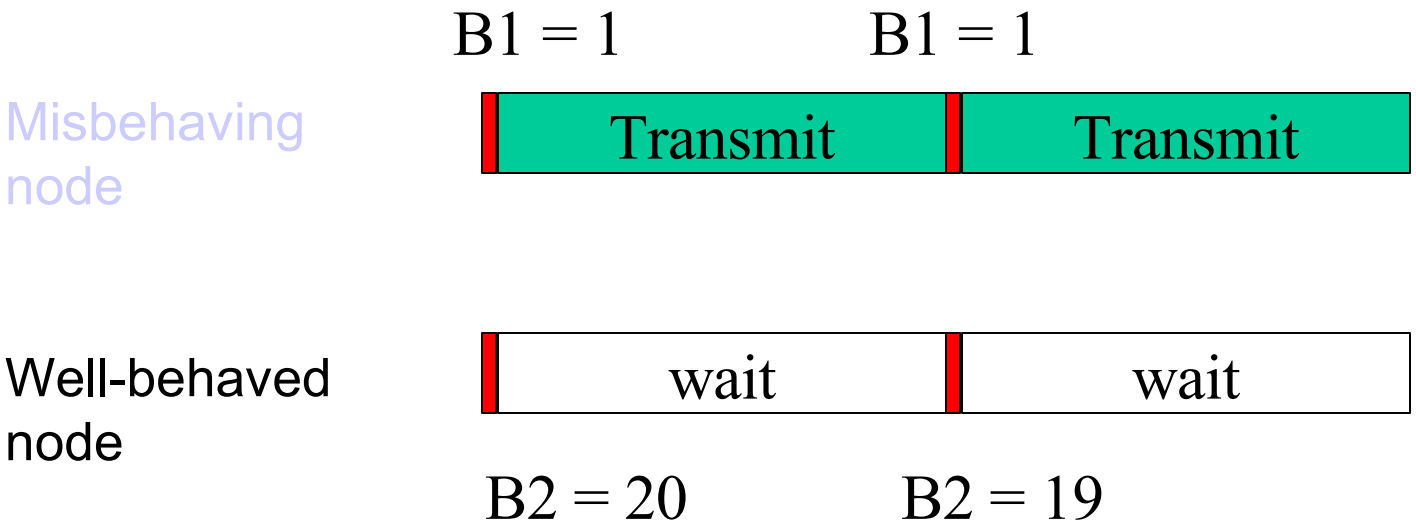




Possible Misbehavior

Backoff from biased distribution

Example: Always select a small backoff value



Goals [Kyasanur03dsn]

Diagnose node misbehavior

- Catch misbehaving nodes

Discourage misbehavior

- Punish misbehaving nodes

MAC Selfishness: Game-Theoretic Approach

MacKenzie addresses selfish misbehavior in Aloha networks

- Nodes may use higher access probabilities

Solution uses game theoretic approach

- Assumes there is some cost for transmitting
- Nodes independently adjust access prob.
- Under some assumptions network reaches a *fair equilibrium*

MAC: Selfishness

[Konorski01, Konorski02] discuss selfish misbehavior in 802.11 networks

Game theory used to analyze solution

- Nodes use a black-burst to resolve contention
- Winner is not the largest burst, but node with burst within Δ slots of largest burst

Game theory - Discussion

Protocols resilient to misbehavior can be developed

- Do not need explicit misbehavior detection

Solutions assume perfect knowledge

- No guarantees with imperfect information

Performance at equilibrium may be poor

Alternative Approach

Use payment schemes, charging per packet

Misbehaving node can achieve lower delay (e.g., by sending packet bursts)

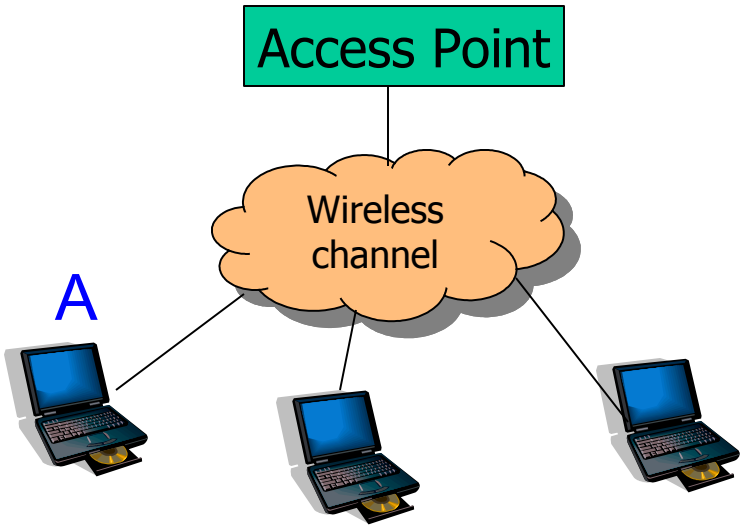
→ Average delay is less with *same* cost

Per-packet payment schemes not sufficient (need to factor delay – harder)

Another Approach

Receivers detect sender misbehavior

Assume receivers are well-behaved (can be relaxed)



- Access Point is trusted. When AP transmits, it is well-behaved
- When AP receives, it can monitor sender behavior

Issues

Receiver does not know exact backoff value chosen by sender

- Sender chooses random backoff
- Hard to distinguish between maliciously chosen small values and a legitimate random sequence

Wireless channel introduces uncertainties

- Channel status seen by sender and receiver may be different

Potential Solution:
Use long-term statistics

Observe backoffs chosen by sender over multiple packets

Backoff values not from expected distribution ➔
Misbehavior

Selecting right observation interval difficult

A Simpler Approach

Remove the non-determinism

A Simpler Approach

Receiver provides backoff values to sender

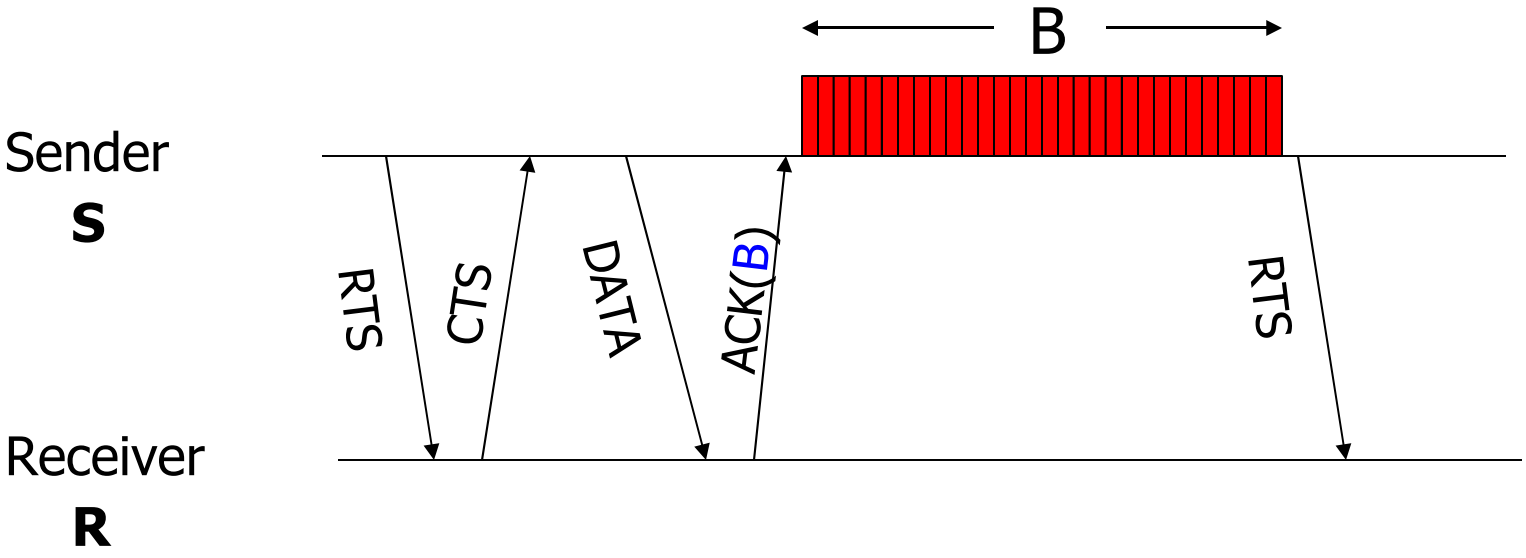
- Receiver specified backoff for next packet in ACK for current packet

Modification does not significantly change 802.11 behavior

- Backoffs of different nodes still *independent*

Uncertainty of sender's backoff eliminated

Modifications to 802.11



- R provides backoff B to S in ACK

  B selected from [0,CW_{min}]

- S uses B for backoff

Protocol steps

Step 1: For each transmission:

Detect deviations: Decide if sender backed off for less than required number of slots

Penalize deviations: Penalty is added, if the sender appears to have deviated

Goal: Identify and penalize suspected misbehavior

Reacting to individual transmission makes it harder to adapt to the protocol

Protocol steps

Step 2: Based on last W transmissions:

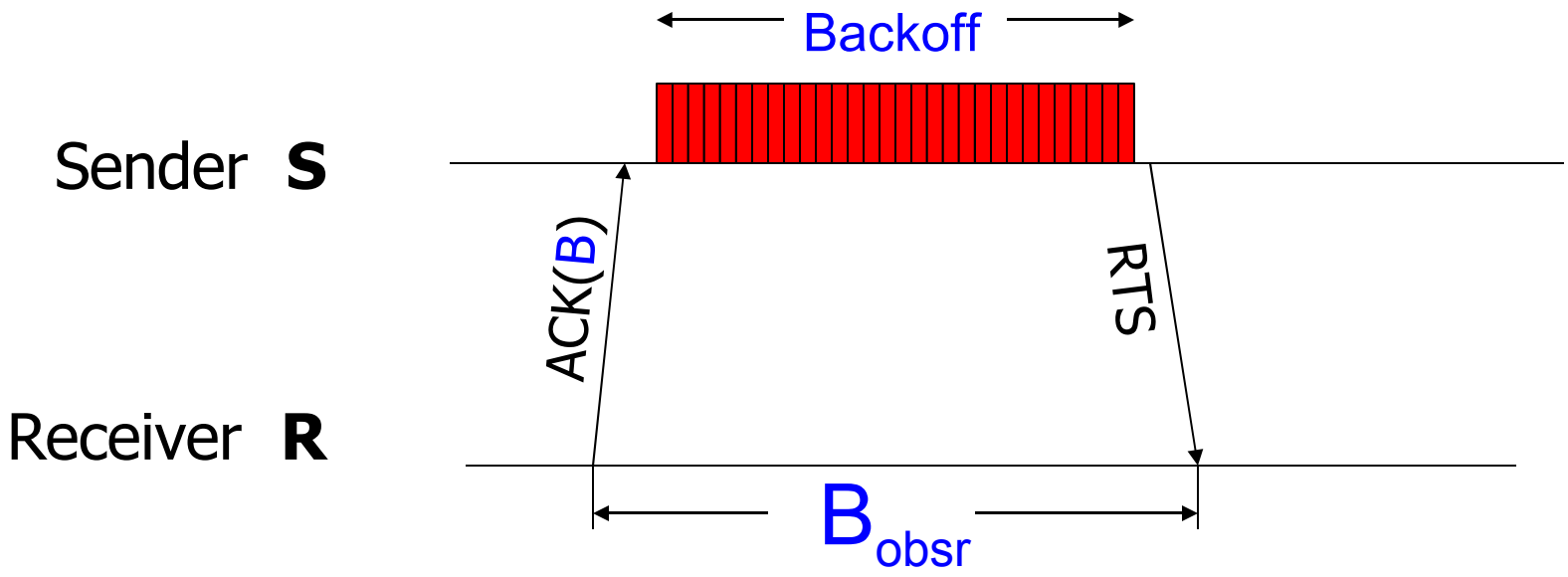
Diagnose misbehavior: Identify misbehaving nodes

Goal: Identify misbehaving nodes with high probability

Reduce impact of channel uncertainties

Filter out misbehaving nodes from well-behaved nodes

Detecting deviations

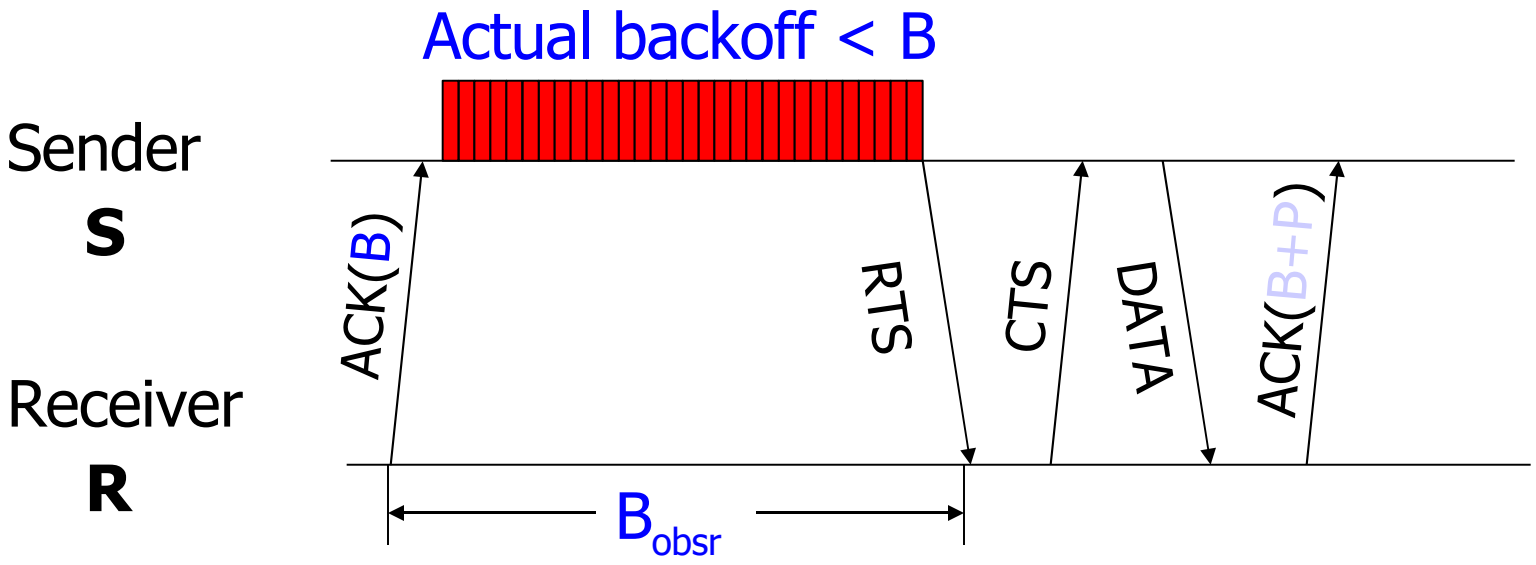


Receiver counts number of idle slots B_{obsr}

Condition for detecting deviations: $B_{obsr} < \alpha B$

$(0 < \alpha <= 1)$

Penalizing Misbehavior



- When $B_{obsr} < \alpha B$, penalty P added
  - P proportional to $\alpha B - B_{obsr}$
- Total backoff assigned = B + P

Penalty Scheme issues

Misbehaving sender has two options

- Ignore assigned penalty ➔ Easier to detect
- Follow assigned penalty ➔ No throughput gain

With penalty, sender has to misbehave more for same throughput gain

Diagnosing Misbehavior

Total deviation for last W packets used

Deviation per packet is $B - B_{obsr}$

If total deviation > THRESH then sender is designated as misbehaving

Higher layers / administrator can be informed of misbehavior

MANET
Implementation Issues

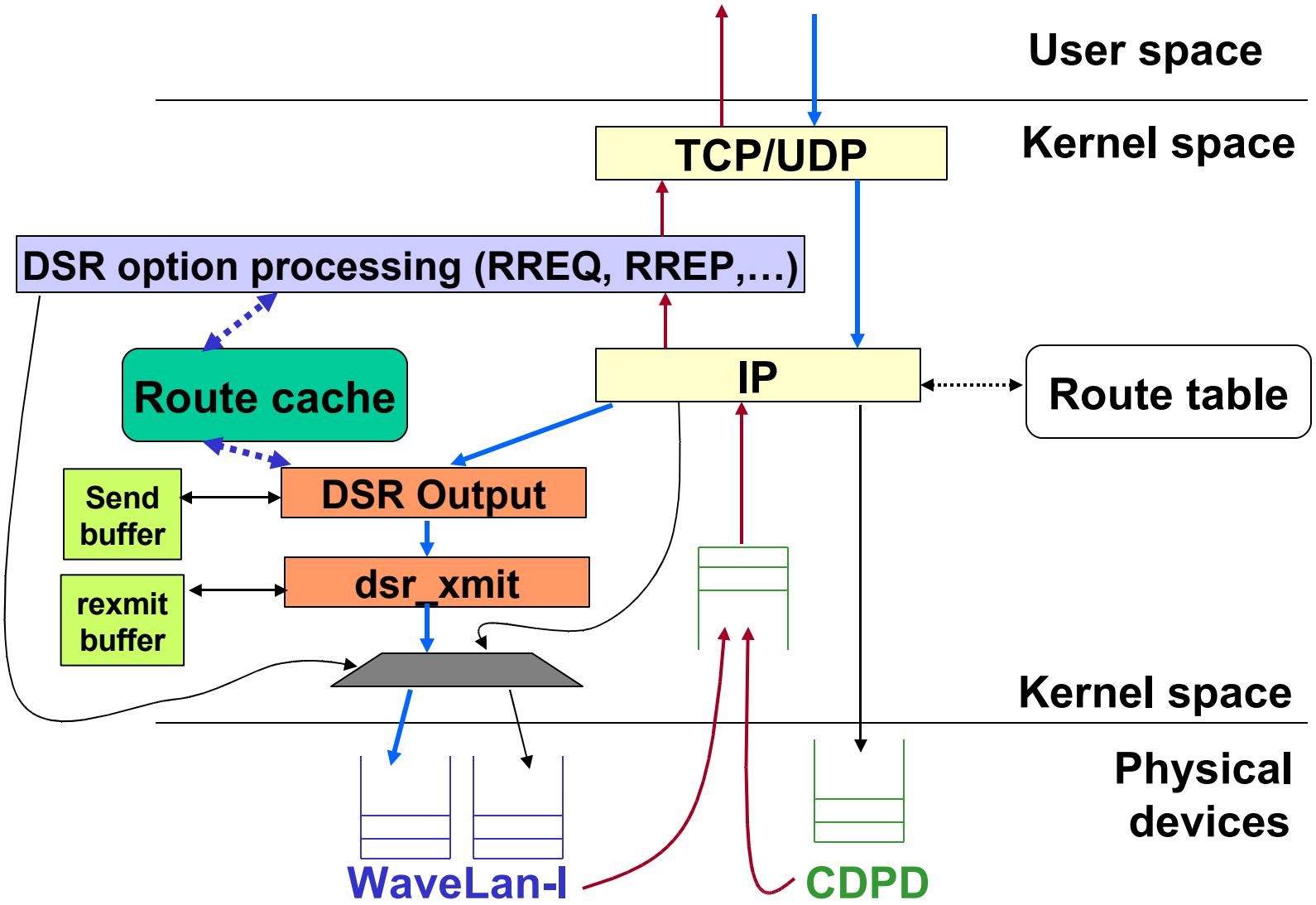
Existing Implementations

Several implementations apparently exist (see IETF MANET web site)

Only a few available publicly

Most implementations focus on unicast routing

CMU Implementation [Maltz99]



CMU Implementation: Lessons Learned

Multi-level priority queues helpful: Give higher priority to routing control packets, and lower for data

If retransmission is implemented above the link layer, it must be adaptive to accommodate congestion

- Since Wavelan-I MAC does not provide retransmissions, DSR performs retransmits itself
- DSR per-hop ack needs to contend for wireless medium
- Time to get the ack (RTT) is dependent on congestion
- TCP-like RTT estimation and RTO used for triggering retransmits by DSR on each hop
- This is not very relevant when using IEEE 802.11 where the ack is sent immediately after data reception

CMU Implementation: Lessons Learned

"Wireless propagation is not what you would expect" [Maltz99]

Straight flat areas with line-of-sight connectivity had worst error rates

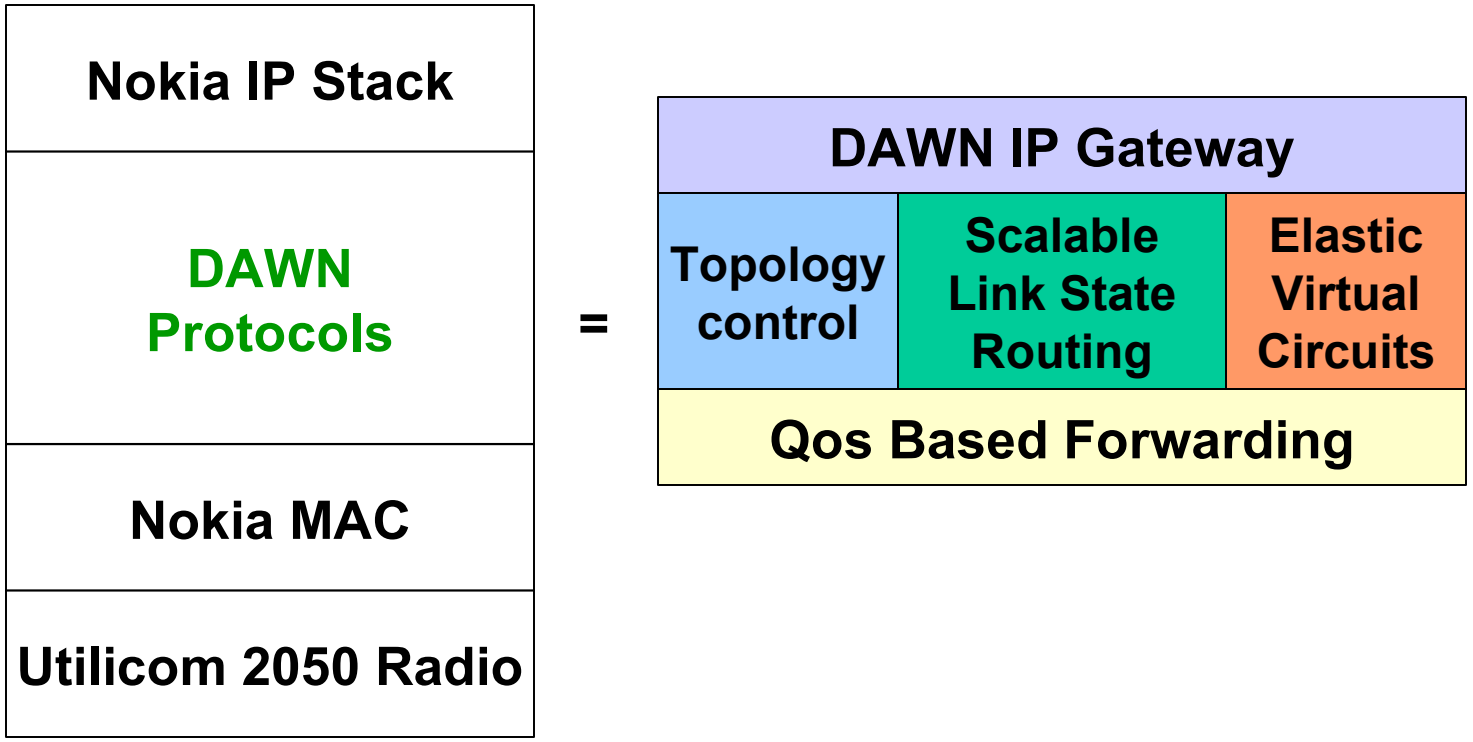
"Bystanders will think you are nuts" [Maltz99]

If you are planning experimental studies in the streets, it may be useful to let police and security guards know in advance what you are up to

BBN Implementation [Ramanathan00Wcnc]

Density and Asymmetric-Adaptive Wireless Network (DAWN)

Quote from [Ramanathan00Wcnc]: *DAWN is a "subnet" or "link" level system from IP's viewpoint and runs "below" IP*



DAWN Features

Topology control by transmit power control

- To avoid topologies that are too sparse or too dense
- To extend battery life

Scalable link state routing: Link state updates with small TTL (time-to-live) sent more often, than those with greater TTL

- As a packet gets closer to the destination, more accurate info is used for next hop determination

Elastic Virtual Circuits (VC):

- Label switching through the DAWN nodes (label = VC id)
- Path repaired transparent to the endpoints when hosts along the path move away

Implementation Issues:
Where to Implement Ad Hoc Routing

Link layer

Network layer

Application layer

Implementation Issues: Security

How can I trust you to forward my packets without tampering?

- Need to be able to detect tampering

How do I know you are what you claim to be ?

- Authentication issues
- Hard to guarantee access to a certification authority

Implementation Issues

Can we make any guarantees on performance?

- When using a non-licensed band, difficult to provide hard guarantees, since others may be using the same band

Must use an licensed channel to attempt to make any guarantees

Implementation Issues

Only some issues have been addresses in existing implementations

Security issues often ignored

Address assignment issue also has not received sufficient attention

Integrating MANET with the Internet [Broch99]

Mobile IP + MANET routing

At least one node in a MANET should act as a gateway to the rest of the world

Such nodes may be used as foreign agents for Mobile IP

IP packets would be delivered to the foreign agent of a MANET node using Mobile IP. Then, MANET routing will route the packet from the foreign agent to the mobile host.

Related Standards Activities

Internet Engineering Task Force (IETF) Activities

IETF manet (**Mobile Ad-hoc Networks)** working group

http://www.ietf.org/html.charters/manet-charter.html

IETF mobileip (**IP Routing for Wireless/Mobile Hosts)** working group

http://www.ietf.org/html.charters/mobileip-charter.html

Internet Engineering Task Force (IETF) Activities

IETF pilc (**Performance Implications of Link Characteristics)** working group

http://www.ietf.org/html.charters/pilc-charter.html

http://pilc.grc.nasa.gov

Refer [RFC2757] for an overview of related work

Related Standards Activities

BlueTooth

http://www.bluetooth.com

HomeRF [Lansford00ieee]

http://www.homerf.org

IEEE 802.11

http://grouper.ieee.org/groups/802/11/

Hiperlan/2

http://www.etsi.org/technicalactiv/hiperlan2.htm

Bluetooth
[Haartsen98,Bhagawat00Tutorial]

Features: Cheaper, smaller, low power, ubiquitous, unlicensed frequency band

Spec version 1.0B released December 1999

(1000+ pages)

Promoter group consisting of 9

Ericsson, IBM, Intel, Nokia, Toshiba, 3Com, Lucent, Microsoft, Motorola

1800+ adopters

Bluetooth: Link Types

Designed to support multimedia applications that mix voice and data

Synchronous Connection-Oriented (SCO) link

- Symmetrical, circuit-switched, point-to-point connections
- Suitable for voice
- Two consecutive slots (forward and return slots) reserved at fixed intervals

Asynchronous Connectionless (ACL) link

- Symmetrical or asymmetric, packet-switched, point-to-multipoint
- Suitable for bursty data
- Master units use a polling scheme to control ACL connections

Bluetooth: Piconet

A *channel* is characterized by a frequency-hopping pattern

Two or more terminals sharing a channel form a *piconet*

1 Mbps per Piconet

One terminal in a piconet acts as a *master* and up to 7 *slaves*

Other terminals are *slaves*

Polling scheme: A slave may send in a slave-to-master slot when it has been addressed by its MAC address in the previous master-to-slave slot

Inter-Piconet Communication

A slave can belong to two different piconets, but not at the same time

A slave can leave its current piconet (after informing its current master the duration of the leave) and join another piconet

A maser of one piconet can also join another piconet temporarily as a slave

Bluetooth: Scatternet

Several piconets may exist in the same area (such that units in different piconets are in each other's range)

Each piconet uses a different channel and gets 1 Mbps for the piconet

- Since two independently chosen hopping patterns may select same hop simultaneously with non-zero probability, some collisions between piconets are possible, reducing effective throughput

A group of piconets is called a *scatternet*

Routing

Ad hoc routing protocols needed to route between multiple piconets

Existing protocols may need to be adapted for Bluetooth [Bhagwat99Momuc]

- For instance, not all nodes within transmission range of node X will hear node X
  - Only nodes which belong to node X's current piconet can hear the transmission from X
- Flooding-based schemes need to take this limitation into account

Open Issues
in
Mobile Ad Hoc Networking

Open Problems

Issues other than MAC and routing have received much less attention so far

Other interesting problems:

Improving interaction between protocol layers

Distributed algorithms for MANET

Applications for MANET

Related Research Areas

Algorithms for dynamic networks (e.g., [Afek89])

Sensor networks [DARPA-SensIT]

- Ad hoc network of sensors
- Addressing based on data (or function) instead of name
  - “send this packet to a temperature sensor”

Thank you !!

For more information, send e-mail to
Nitin Vaidya at
nhv@uiuc.edu