

Modeling of data networks by example: NS-2 (III)

Wireless Networks

Holger Füßler

Course overview

1. Introduction

2. Building block: RNG

3. Building block:
Generating random variates I
and modeling examples

4. Building block:
Generating random variates II
and modeling examples

5. Algorithmics:
Management of events

6. NS-2: Introduction

7. NS-2: Fixed networks

8. NS-2: Wireless networks

9. Output analysis: single system

10. Output analysis: comparing
different configurations

11. Omnet++ / OPNET

12. Simulation lifecycle, summary

Lecture overview

Retrace and understand a typical use case for simulation of (wireless) computer networks

- » **Part I: Basics on Mobile Ad-Hoc Network Routing**
- » **Part II: Ns-2 and the cmu wireless extensions**
- » **Part III: Ns-2 scripts for MANET simulation**
- » **Part IV: Output of MANET simulations**
 - the CMU trace file
 - vizualization and analysis

Lecture overview

Retrace and understand a typical use case for simulation of (wireless) computer networks

» **Part I: Basics on Mobile Ad-Hoc Network Routing**

» **Part II: Ns-2 and the cmu wireless extensions**

» **Part III: Ns-2 scripts for MANET simulation**

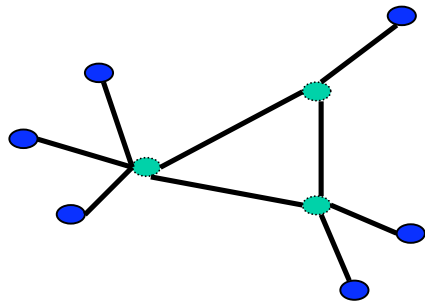
» **Part IV: Output of MANET simulations**

- the CMU trace file
- vizualization and analysis

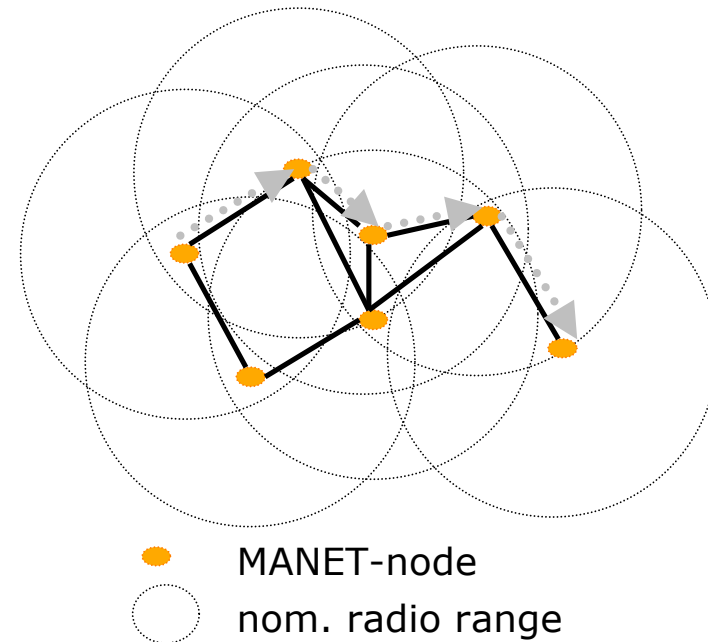
Mobile Ad-Hoc Networking

Characterization:

- All (network) nodes are equipped with radio technology
- nodes can be mobile
- Networks are forming up spontaneously
- Nodes are routers *and* end-systems



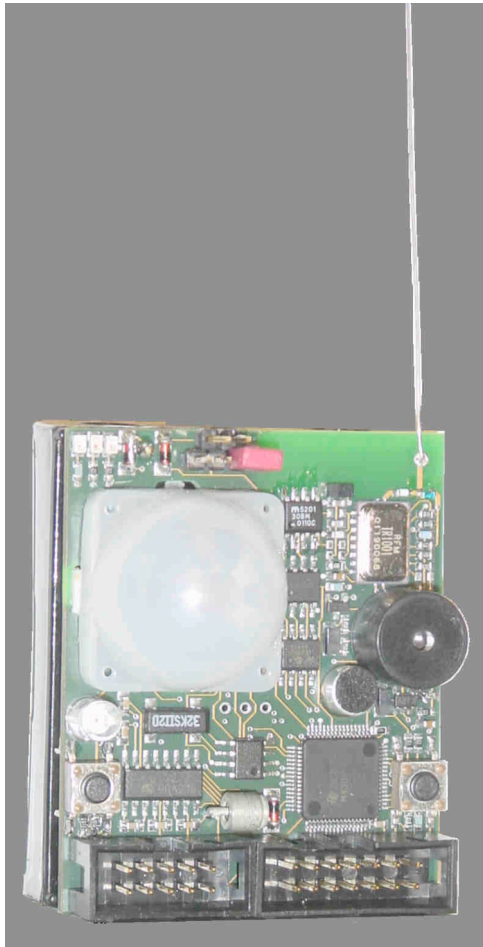
- End-System
- Router



Mobile Ad-Hoc Network Routing

- Routing is the „classical“ MANET problem
 - links often unidirectional
 - radio range unstable
 - strict energy constraints
 - mobility- /radio-induced topology changes
- But also:
 - link / physical layer research
 - cooperativeness
 - security
 - transport layer (TCP ceases to work over wireless links)

MANET example 1: Sensor networks



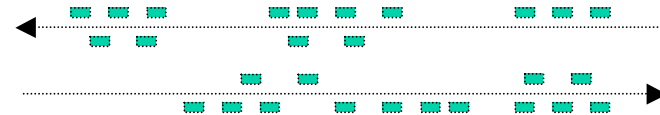
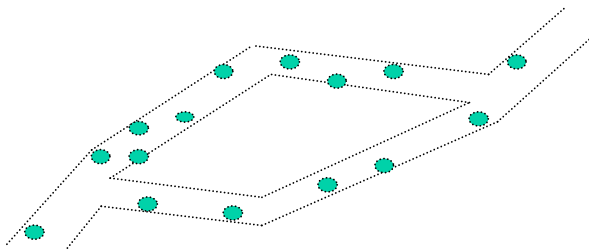
- usually high node density
- but also low mobility
- dominating node property is the availability of sensors (light/heat/movement)
- single nodes are highly restricted on energy and computation power

Embedded Sensor Board, FU Berlin (<http://www.scatterweb.de>)

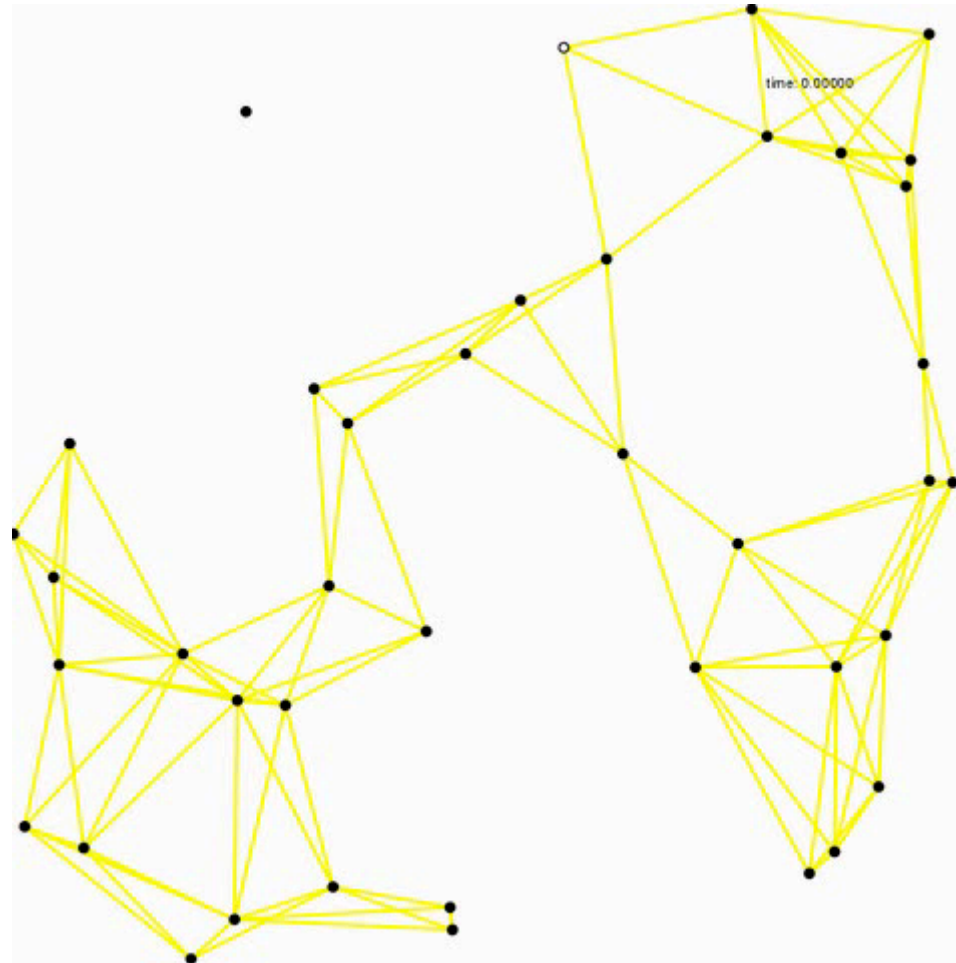
MANET example 2: Vehicular Ad-Hoc Networks



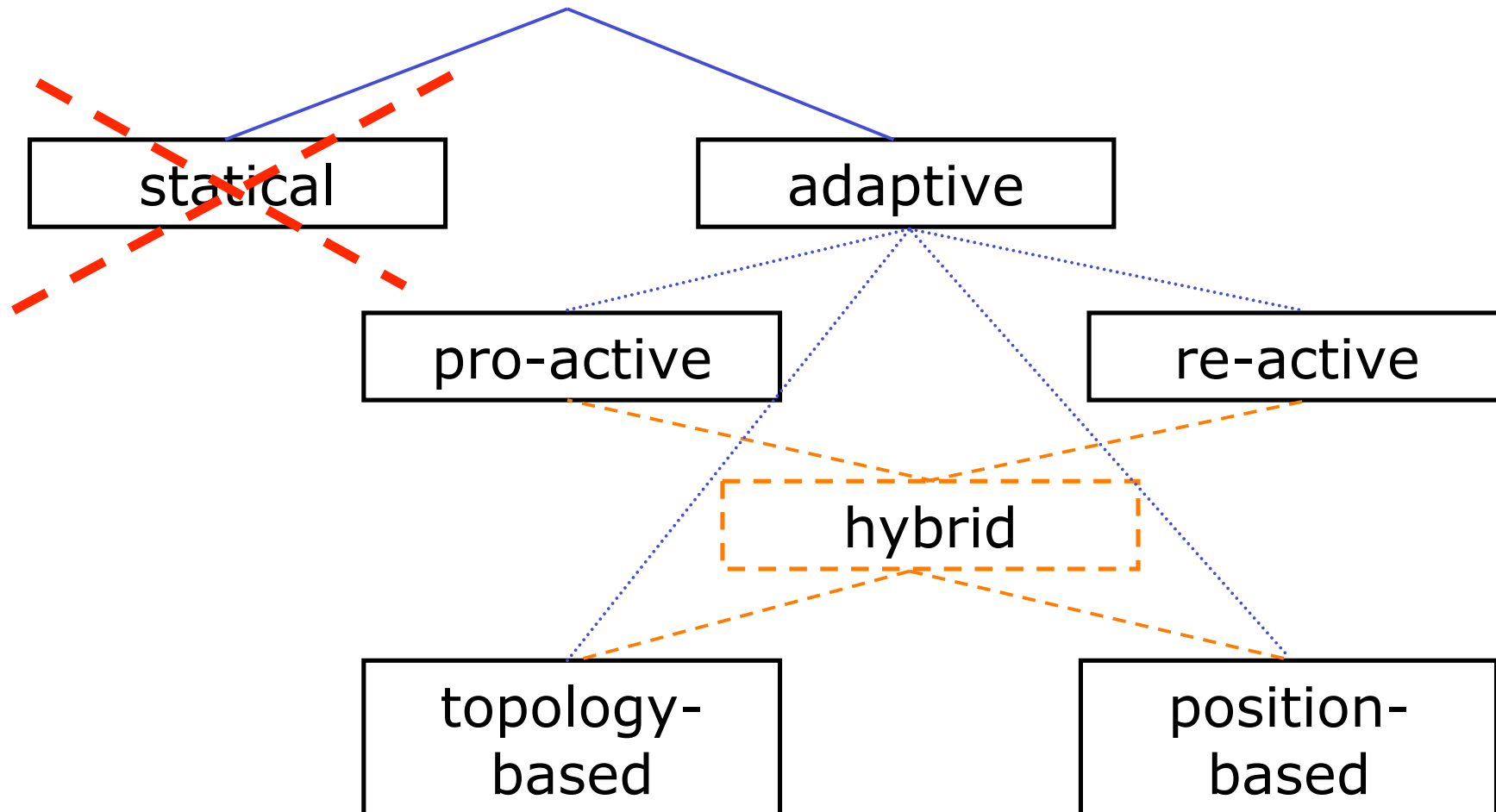
- Ad-Hoc Networks between street-bound cars based on ad-hoc network principles (see <http://www.fleetnet.de>)
- additional sensors (position / movement / possibly radar)
- very high mobility
- almost no energy restrictions



Node Mobility – Why is routing hard?



Classification of Routing Algorithms



Proactive vs. Reactive

- Proactive Algorithms:
 - all nodes permanently keep routes to all other nodes
- Reactive Algorithms:
 - only when communication is desired, the nodes „build up“ a routing information
- Discussion:
 - proactive is more suited for „equally distributed“ communication
 - But: Above a certain mobility rate, pro-active routing will fully load the network

Classification of Routing Algorithms

- **Topology-Based Algorithms:**
 - a (distributed) network topology is built based on the neighborhood relationship between nodes
 - the actual routing is done by
 - Source Routing
 - Distance-Vector Routing
 - Shortest-Path Routing (Link-State)
- **Position-Based Algorithms:**
 - Routing is done mainly in a greedy way minimizing the remaining distance to the destination
 - only possible, when nodes know about their current position
 - if no „greedy route“ is found, a recovery strategy is used

The IETF MANET group

- Topology-Based Algorithms:
 - reactive
 - AODV (Ad-Hoc On-Demand Distance Vector Routing)
 - DSR (Dynamic Source Routing)
 - proactive
 - TBRPF (Truncated Reverse-Path Broadcasting)
 - OLSR (Optimized Link-State Routing)
- Position-Based Algorithms:
 - none (so far ;-)

Classification of Routing Algorithms

- Perkins / Royer 1999 (download available)
- Re-Active Routing Method (i.e. on-demand)
- Topology-Based Distance-Vector Routing
- meanwhile RFC status (experimental RFC 3561)

AODV operation

- » **Route Requests (RREQ) are flooded on-demand**

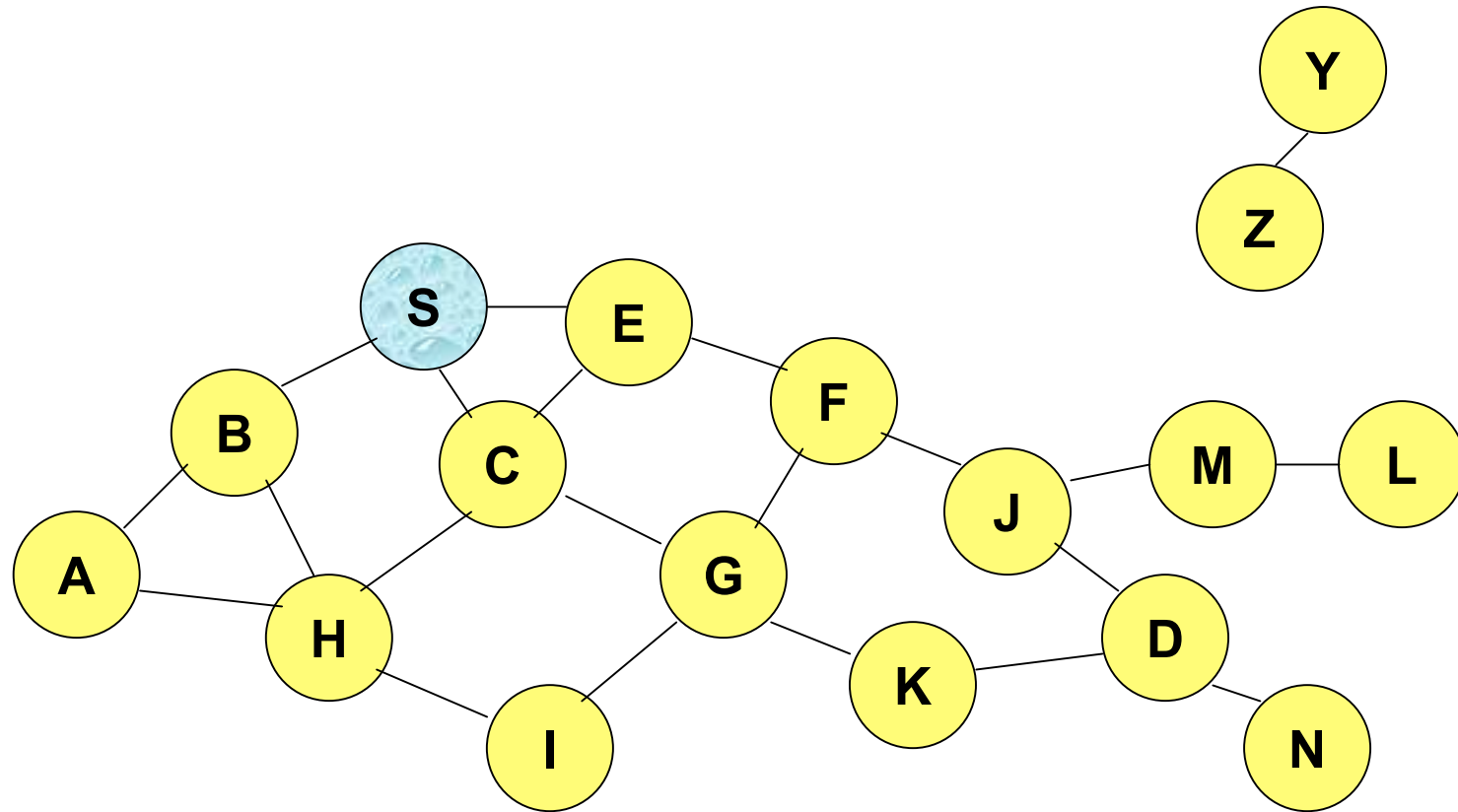
- » **When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source**
 - **AODV assumes symmetric (bi-directional) links**

- » **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**

[Source: Nitin Vaidya]

Route Requests in AODV



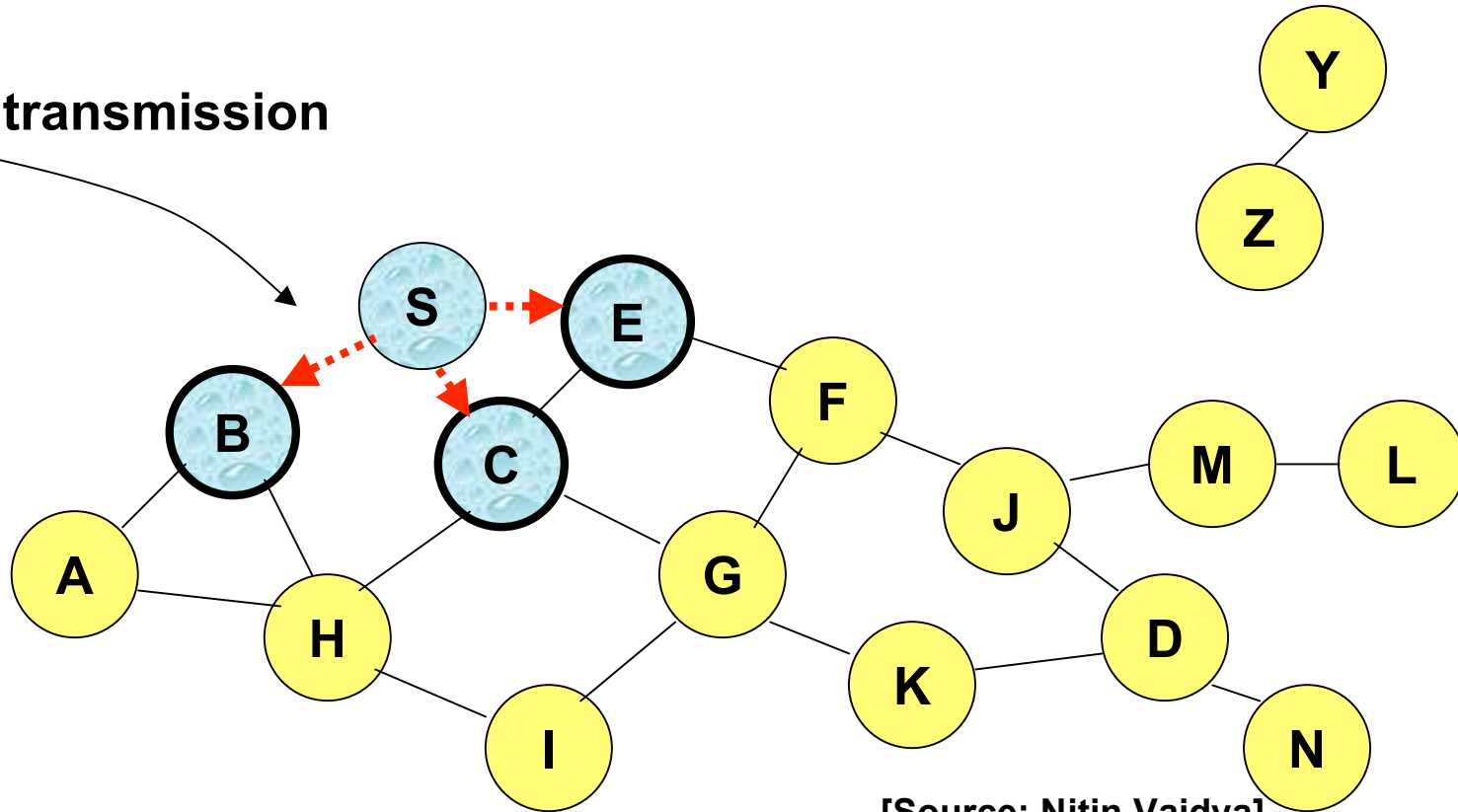
[Source: Nitin Vaidya]



Represents a node that has received RREQ for D from S

Route Requests in AODV

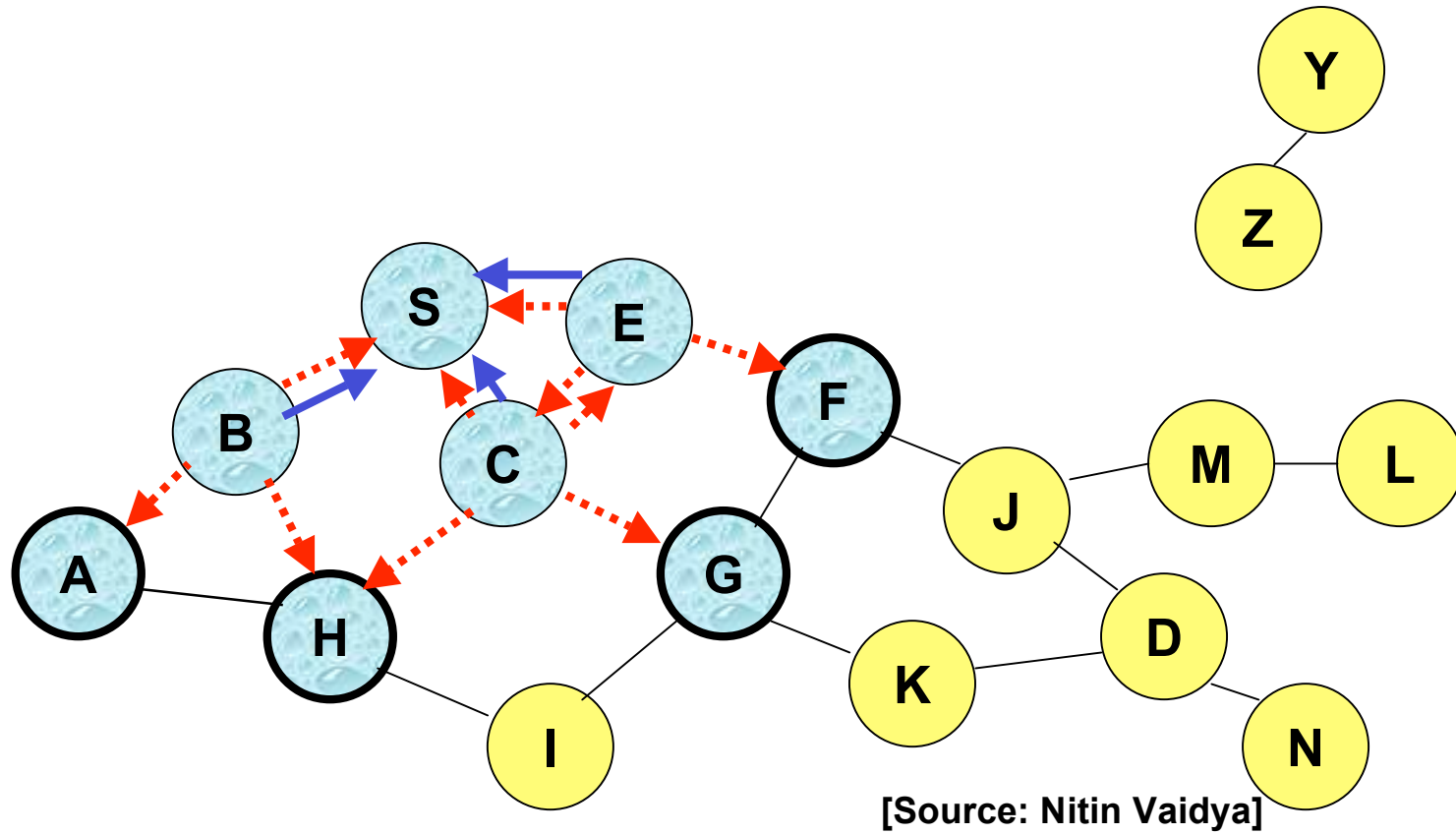
Broadcast transmission



[Source: Nitin Vaidya]

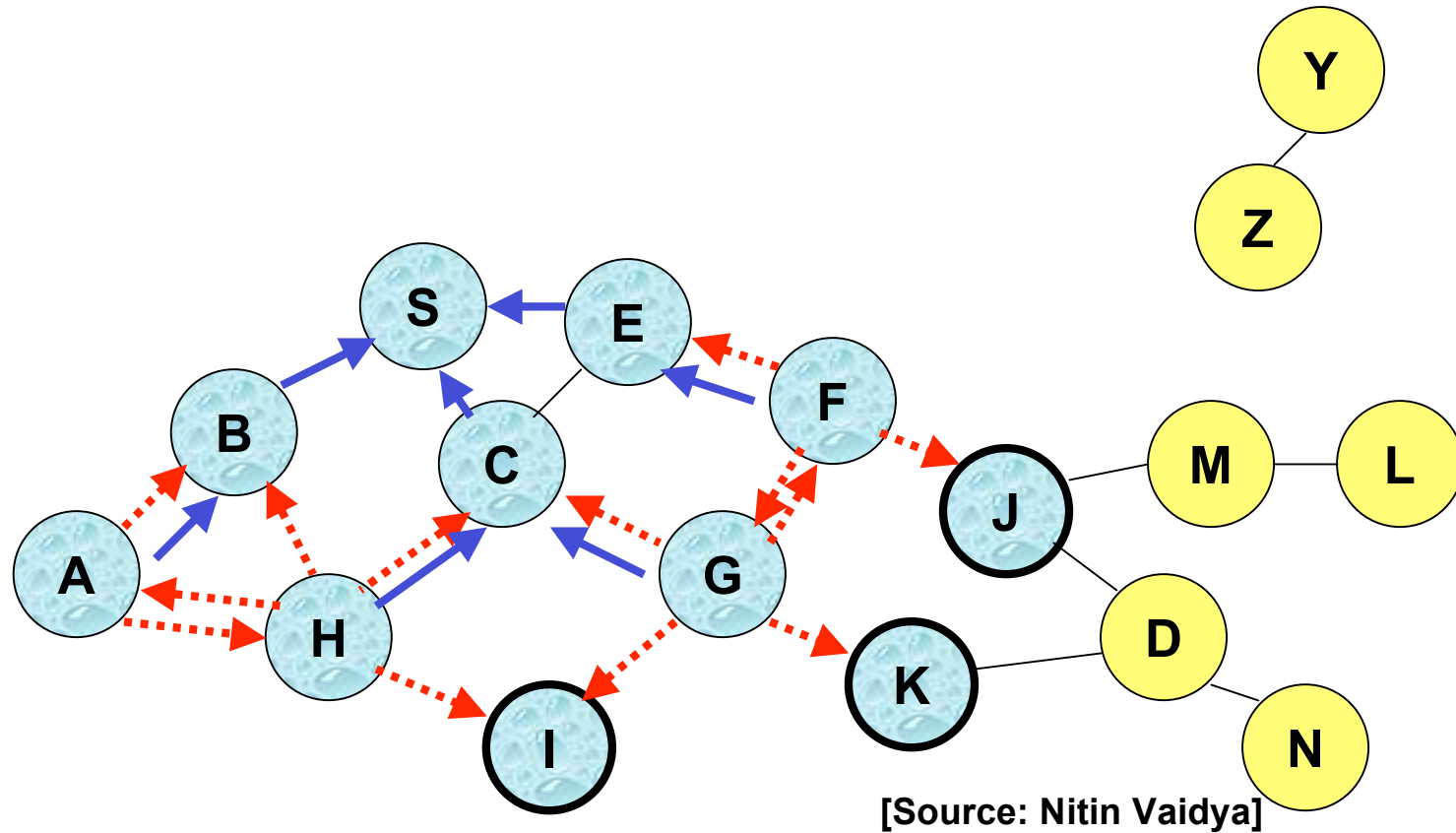
.....➔ Represents transmission of RREQ

Route Requests in AODV



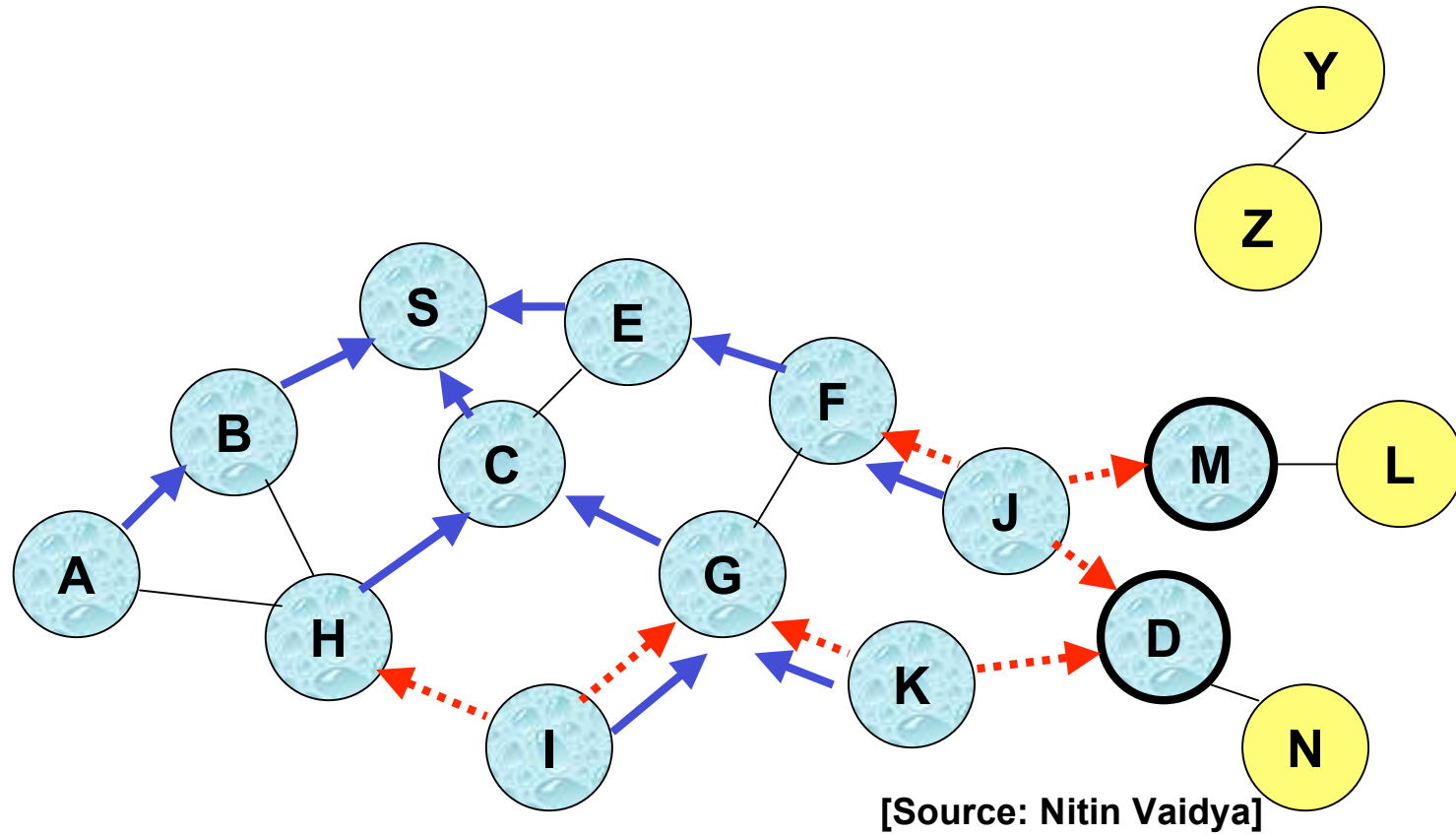
← Represents links on Reverse Path

Reverse Path Setup in AODV

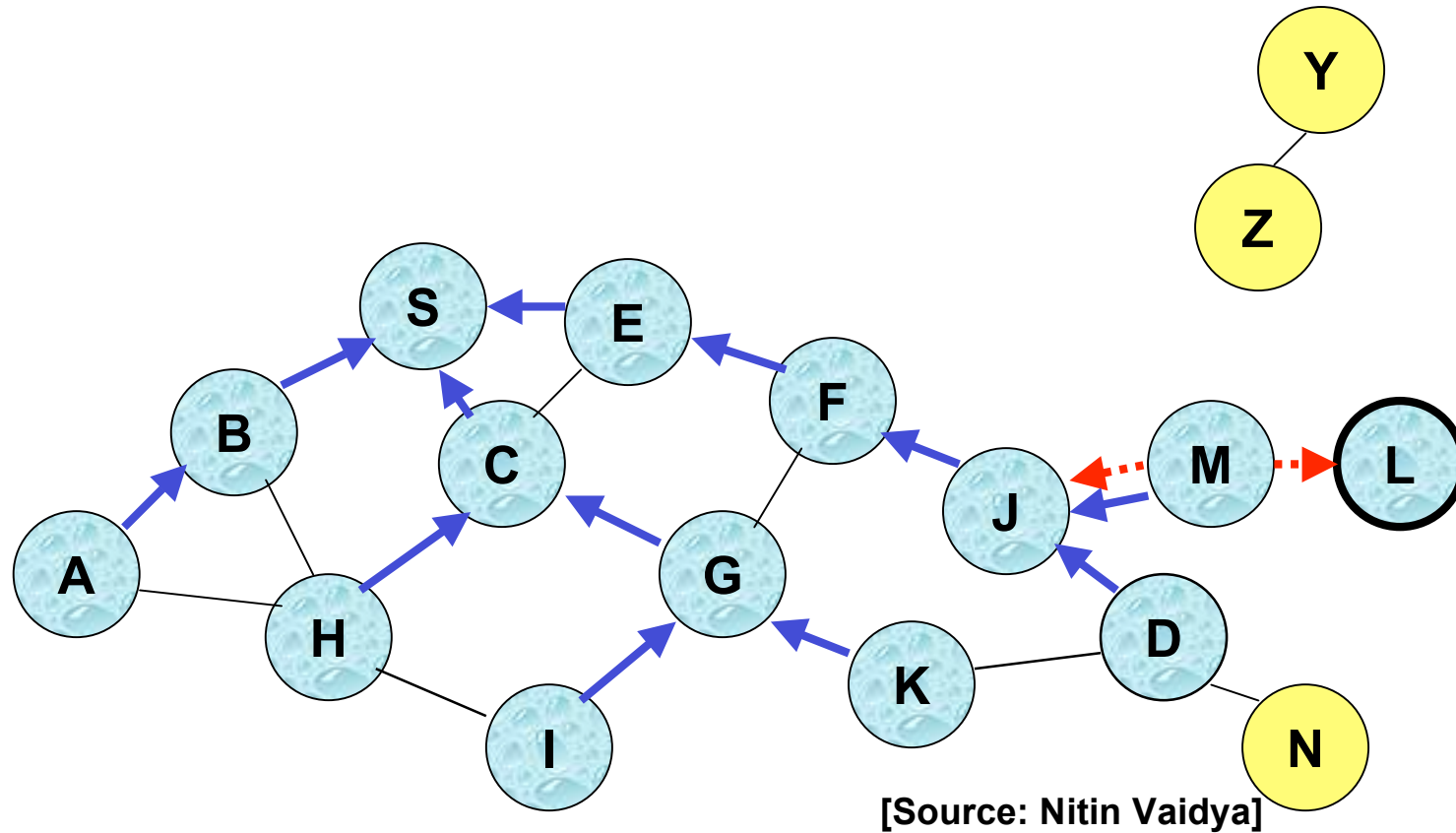


- Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ** once

Reverse Path Setup in AODV

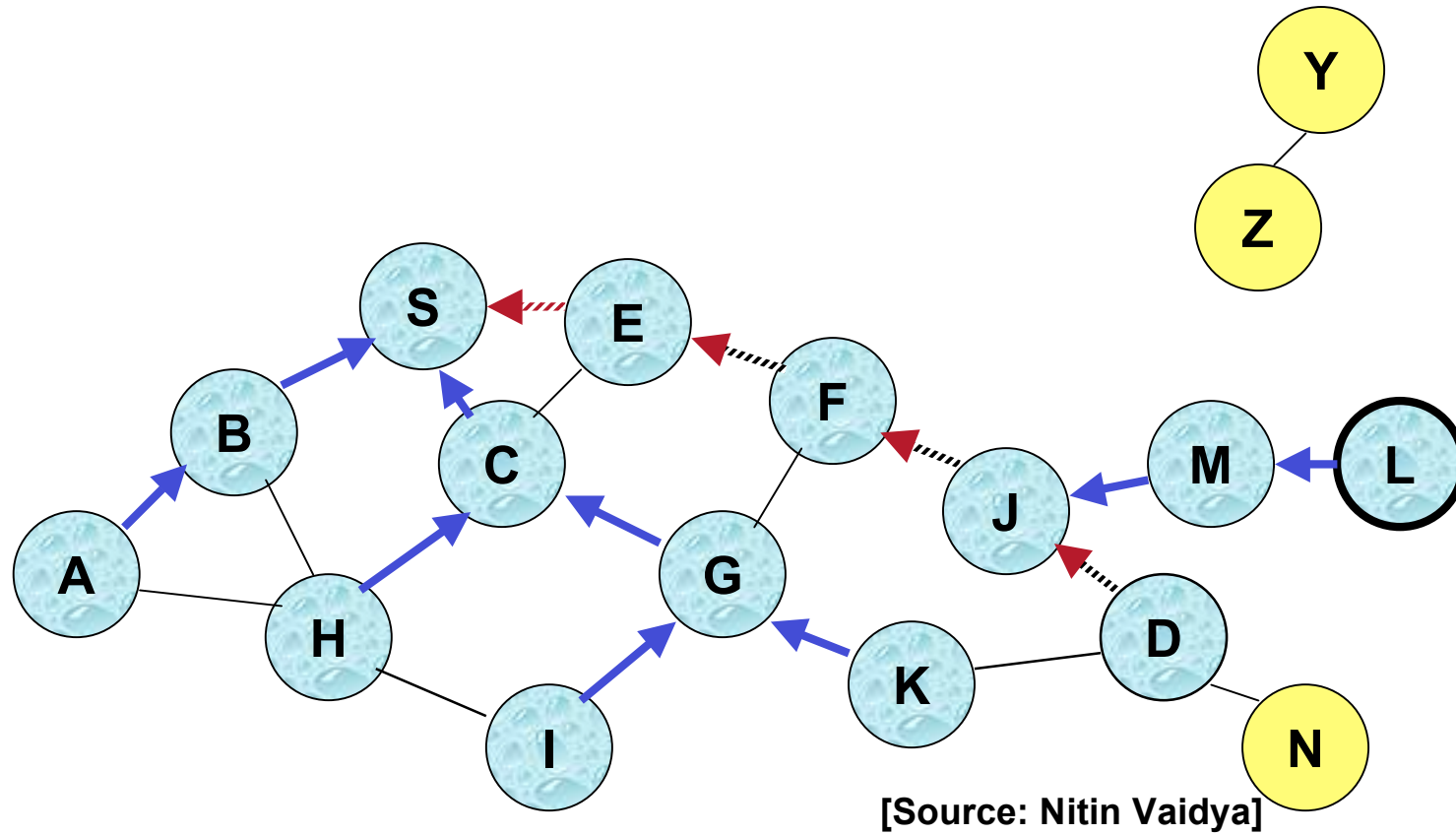


Reverse Path Setup in AODV



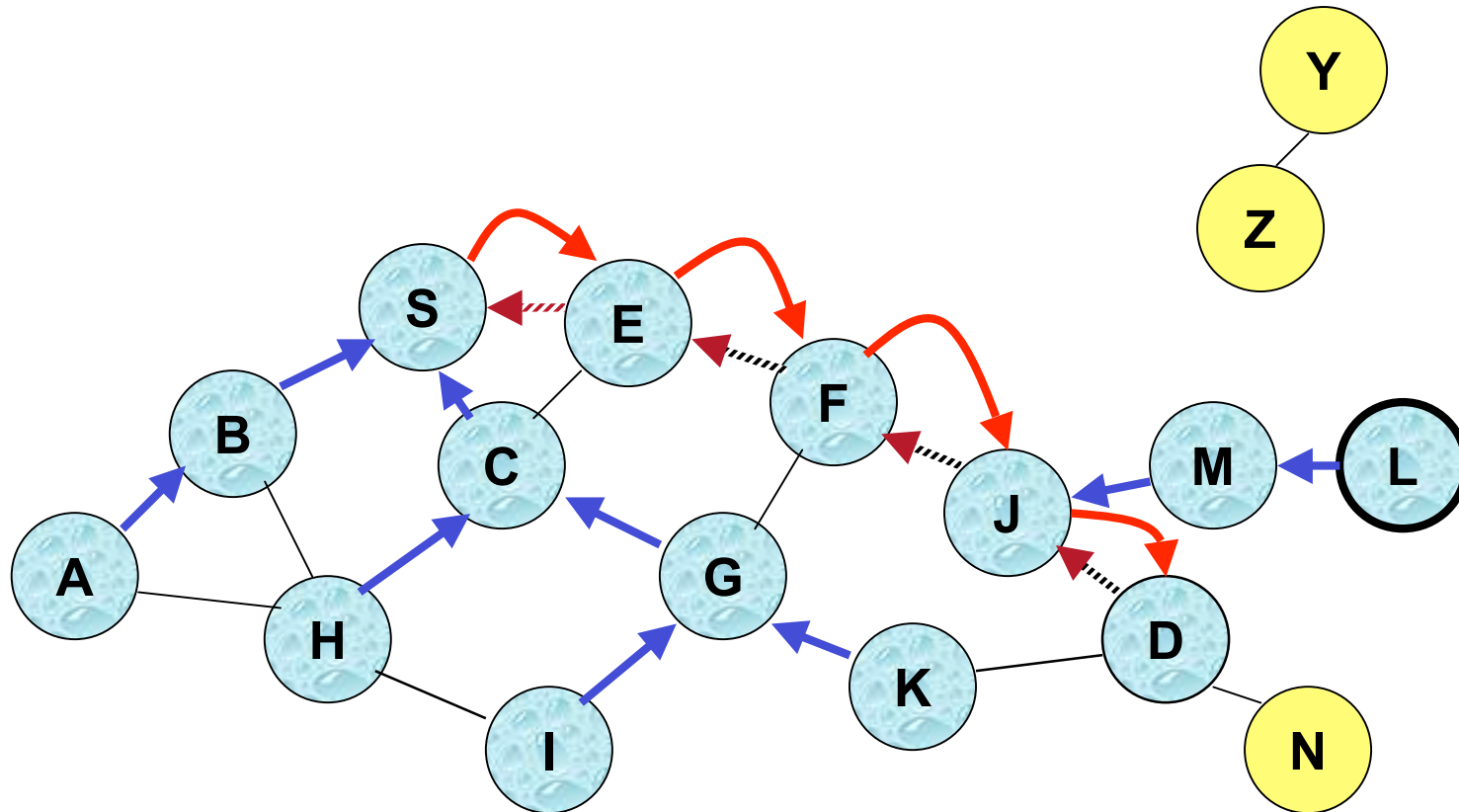
- Node D **does not forward** RREQ, because node D is the **intended target** of the RREQ

Route Reply in AODV



← Represents links on path taken by RREP

Forward Path Setup in AODV

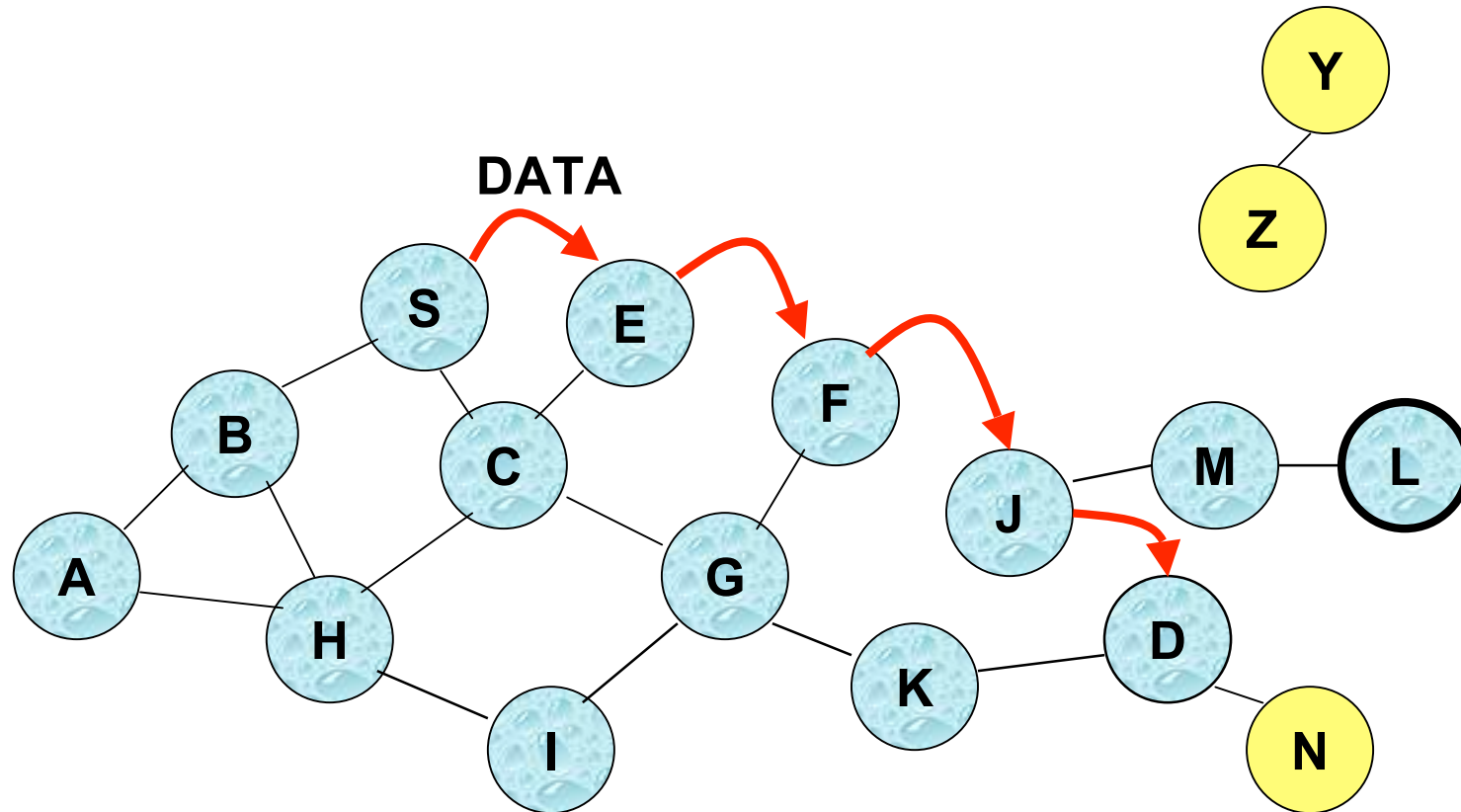


Forward links are setup when RREP travels along the reverse path



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.

Additional Comments

- » **AODV is (in reality) much more complicated**
- » **sequence-number mechanism, among other things for count-to-infinity protection**
- » **lots of protocol parameters (time-out values, link-layer notification, backward path setup via flooding)**

Lecture overview

Retrace and understand a typical use case for simulation of (wireless) computer networks

» **Part I: Basics on Mobile Ad-Hoc Network Routing**

» **Part II: Ns-2 and the CMU wireless extensions**

» **Part III: Ns-2 scripts for MANET simulation**

» **Part IV: Output of MANET simulations**

- the CMU trace file
- vizualization and analysis

NS-2's Wireless Extensions

- » Originally, ns-2 has no support for wireless networks
- » CMU monarch wireless extensions (1998)
(<http://www.monarch.cs.cmu.edu/cmu-ns.html>)
 - mobile nodes with programmable trajectories
 - IEEE 802.11 DCF MAC protocol
 - ARP / DSR / DSDV / TORA
 - wireless networking (Lucent WaveLan DSSS radio)
 - two ray ground radio propagation
 - utility scripts (movement, analysis, vizualization)
 - ...
- » already included in actual ns-2 releases

ns-2 wireless node

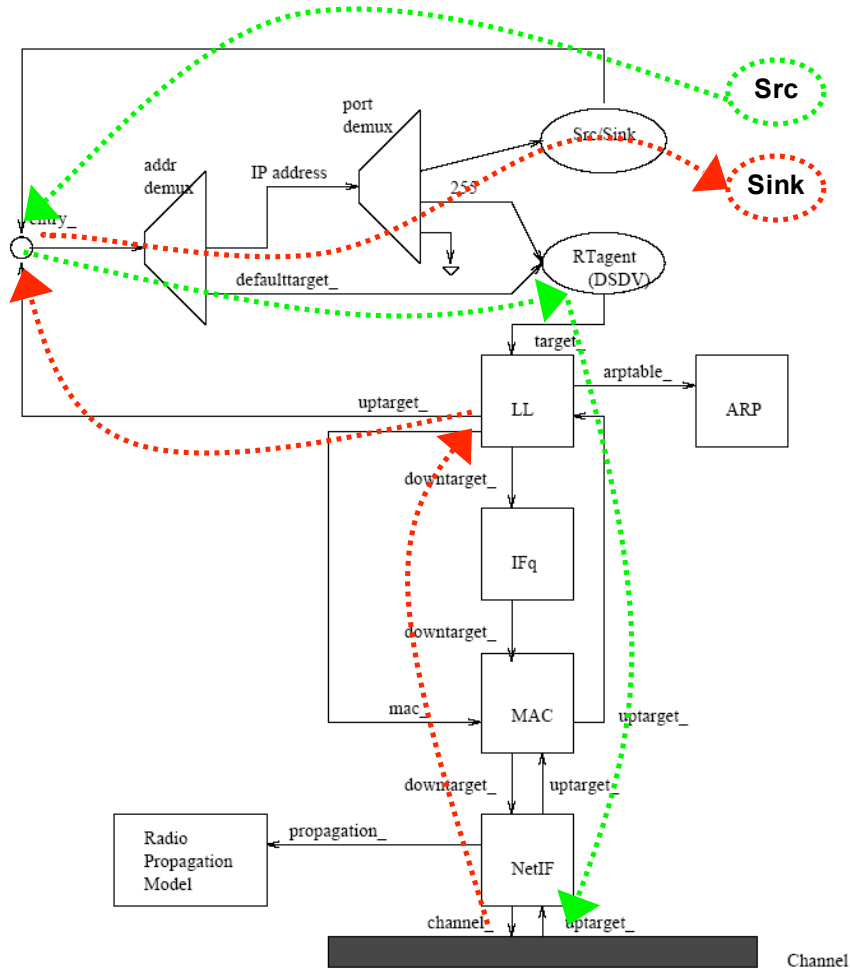


Figure 16.1: Schematic of a mobilenode under the CMU monarch's wireless extensions to ns

1. Packet injected
2. non-local → RTAgent (e.g. AODV)
 - do route request if necessary
 - drop packet or select next hop (add to packet header)
3. hand packet to link layer
 - do ARP if necessary (IP)
4. hand packet to interface queue
5. MAC: get packets one-by-one
 - perform Media Access
6. Radio Propagation Model represents radio characteristics
7. NetIf is interface to channel
 - knows who is sending and if this is jamming my transmission
8. Reception of Packets

Lecture overview

Retrace and understand a typical use case for simulation of (wireless) computer networks

- » **Part I: Basics on Mobile Ad-Hoc Network Routing**
- » **Part II: Ns-2 and the CMU wireless extensions**
- » **Part III: Ns-2 scripts for MANET simulation**
- » **Part IV: Output of MANET simulations**
 - the CMU trace file
 - visualization and analysis

Wireless “scripting basics” – Part 1

Simulation Script

```
# Radio Stuff
set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhy
set val(rr) 250.0
set val(mac) Mac/802_11
set val(bw) 2.0e6
set val(ifq) Queue/DropTail/PriQueue
set val(ll) LL
set val(ant) Antenna/OmniAntenna
set val(ifqlen) 50

# Basic Sim Setup
set val(nn) 11
set val(rp) AODV
set val(x) 2000
set val(y) 300
set val(simtime) 30

God set rrange_ $val(rr)
Mac/802_11 set rrange_ $val(rr)
```

Simulation Script Cont'd

```
# Initialize Global Variables
set ns_ [new Simulator]
set tracefd [open aodv-static_line.tr w]
set god_ [create-god $val(nn)]

$ns_ trace-all $tracefd
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
set channel_ [new $val(chan)]

$ns_ node-config -adhocRouting $val(rp) \
                -llType $val(ll) \
                -macType $val(mac) \
                -ifqType $val(ifq) \
                -ifqLen $val(ifqlen) \
                -antType $val(ant) \
                -phyType $val(netif) \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace ON \
                -movementTrace ON \
                -channel $channel_ \
                -propType $val(prop)
```

Wireless “scripting basics” – node-config options

option	available values	default
general		
addressType	flat, hierarchical	flat
MPLS	ON, OFF	OFF
both satellite- and wireless-oriented		
wiredRouting	ON, OFF	OFF
llType	LL, LL/Sat	""
macType	Mac/802_11, Mac/Csma/Ca, Mac/Sat, Mac/Sat/UnslottedAloha, Mac/Tdma	""
ifqType	Queue/DropTail, Queue/DropTail/PriQueue	""
phyType	Phy/WirelessPhy, Phy/Sat	""
wireless-oriented		
adhocRouting	DIFFUSION/RATE, DIFFUSION/PROB, DSDV, DSR, FLOODING, OMNIMCAST, AODV, TORA	""
propType	Propagation/TwoRayGround, Propagation/Shadowing	""
propInstance	Propagation/TwoRayGround, Propagation/Shadowing	""
antType	Antenna/OmniAntenna	""
channel	Channel/WirelessChannel, Channel/Sat	""
topoInstance	<topology file>	""
mobileIP	ON, OFF	OFF
energyModel	EnergyModel	""
initialEnergy	<value in Joules>	""
rxPower	<value in W>	""
txPower	<value in W>	""
idlePower	<value in W>	""
agentTrace	ON, OFF	OFF
routerTrace	ON, OFF	OFF
macTrace	ON, OFF	OFF
movementTrace	ON, OFF	OFF
errProc	UniformErrorProc	""
FECProc	?	?
toraDebug	ON, OFF	OFF
satellite-oriented		
satNodeType	polar, geo, terminal, geo-repeater	""
downlinkBW	<bandwidth value, e.g. "2Mb">	""

» node-config options from ns-doc

Wireless “scripting basics” – Part 2

Simulation Script Cont'd

```
for {set i 0} {$i < $val(nn)} {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0
}

# source the mvmnt pattern and the comm pattern
source move-static_line.tcl
source comm-static_line.tcl

for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(simtime) "$node_($i) reset";
}

$ns_ at $val(simtime).2 "stop"
$ns_ at $val(simtime).21 "puts \"NS EXITING...\" ; $ns_
halt"

proc stop {} {
    global ns_ tracefd agrfd
    $ns_ flush-trace
    close $tracefd
}

puts "Starting Simulation..."
$ns_ run
```

The Comm Pattern

```
set commsrcno 0
set commdstno 8

set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_($commsrcno) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_($commdstno) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 4.0
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 40
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5 "$cbr_(0) start"
```

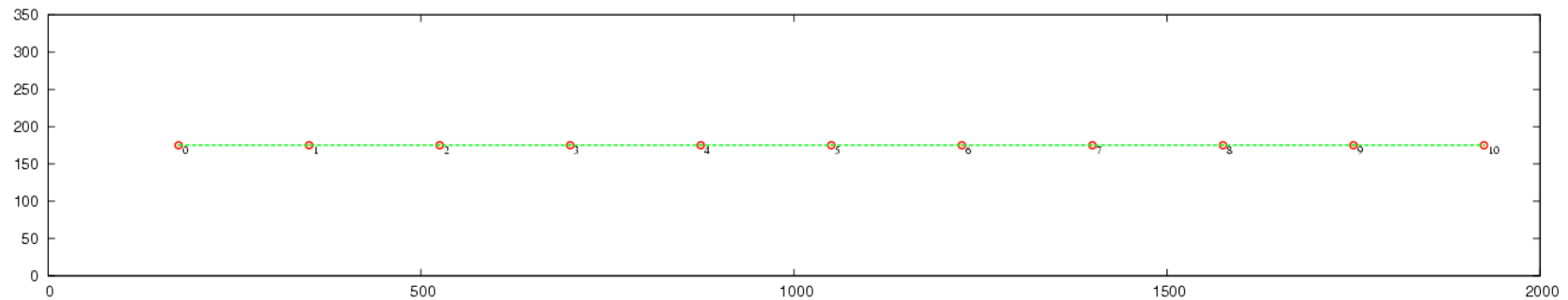

Wireless “scripting basics” – The “Movement Pattern”

move-static_line.tcl

```
$node_(0) set X_      175.00  
$node_(0) set Y_      175.00  
$node_(0) set Z_        0.00  
$node_(1) set X_      350.00  
$node_(1) set Y_      175.00  
$node_(1) set Z_        0.00  
. . .
```

move-static_line.tcl cont'd

```
. . .  
$node_(10) set X_     1925.00  
$node_(10) set Y_      175.00  
$node_(10) set Z_        0.00
```



Excursion: Real Movement

- » **Real Movement, i.e. nodes changing position is supported as follows:**
 - `$ns at $time "$node setdest x2 y2 <speed>"`**
- » **letting the node \$node move from the position it holds at simulation time \$time to x2 / y2 with <speed> m/s beginning at time \$time**
- » **Usually, a utility like “setdest” (in the indep-utils dir) is used to generate random waypoint traffic**
- » **Or: Use real-live movements “converted” to ns-2 input**

Lecture overview

Retrace and understand a typical use case for simulation of (wireless) computer networks

» Part I: Basics on Mobile Ad-Hoc Network Routing

» Part II: Ns-2 and the CMU wireless extensions

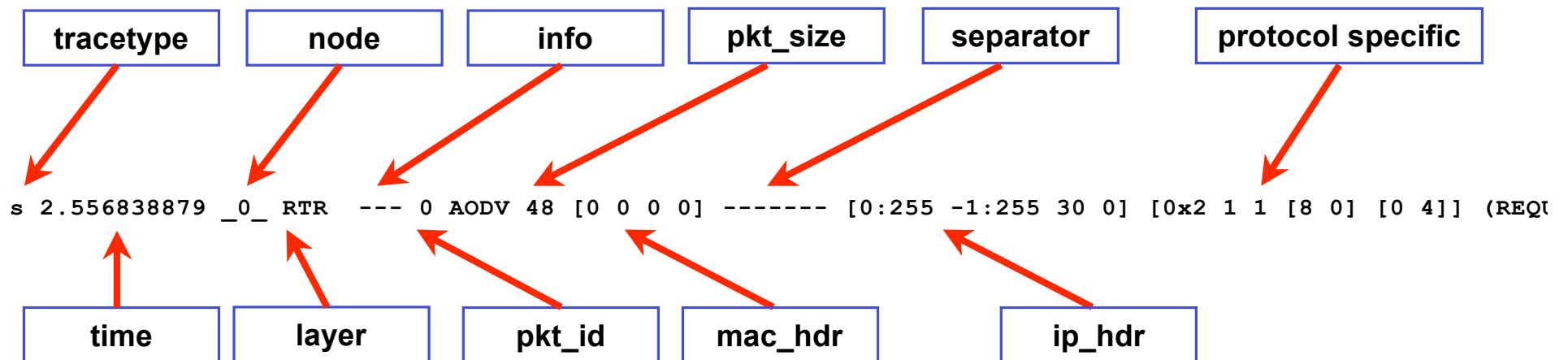
» Part III: Ns-2 scripts for MANET simulation

» Part IV: Output of MANET simulations

- the CMU trace file
- vizualization and analysis

Trace Output

- » The content of the file is controlled by node-config
- » Structure different from wired traces
- » typical line:



Trace Output (Details)

» Mac Header:

[duration dst src pkt_type]

- Duration: Only for RTS/CTS (Network Allocation Vector)
- dst / src: MAC dst/src, for broadcast ffffffff
- pkt_type: Packet Type of enclosed packet (800 for IP)

» IP Header:

[src:port dst:port ttl next_hop]

- src / dst: network address of node (end-2-end), -1 for broadcast
- ports: port numbers (255 for routing)
- ttl: time to live in hops
- next_hop (from IPs point of view)

» further details: “ns-2.27/trace/cmu-trace.(h|cc)”

Tracing: Degrees of Freedom

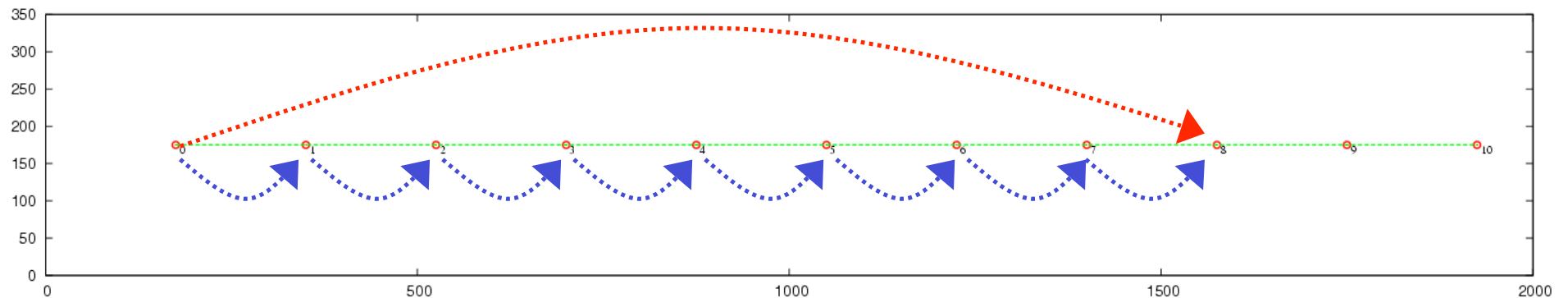
- » **also: new trace format of form `-<valueDescription> value` activated with `$ns new-trace`**
 - Better to parse
 - Harder to read (for me [HMF])
 - Almost twice as Big, but bzip2 handles it pretty well

- » **also: node position logging in every line**

- » **also: mechanism for custom lines in trace file**

Example Scenario

- » 11 nodes in a row
- » Routing Protocol: AODV
- » CBR traffic: node 0 with node 8

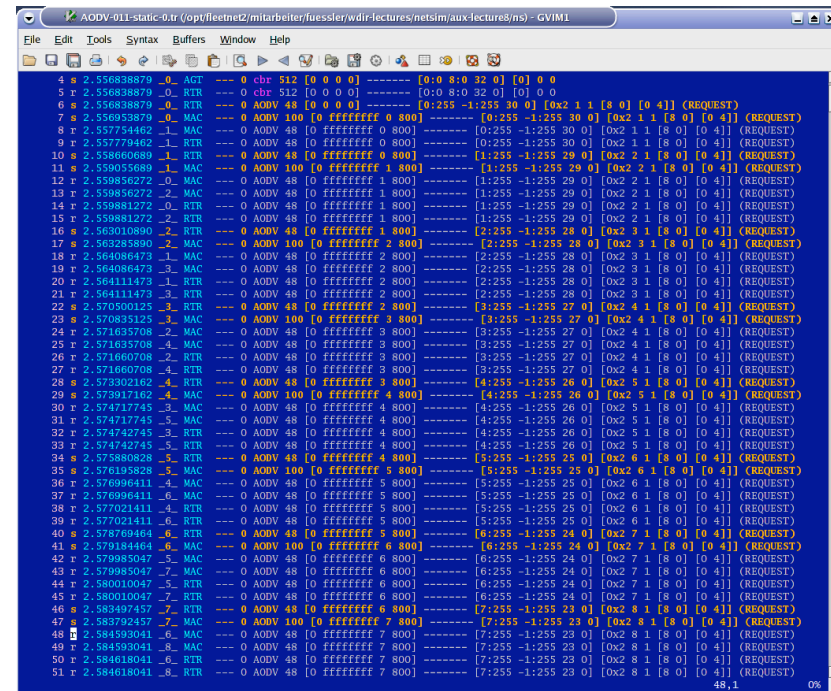


Example Trace

» Open Trace File in Editor

- AODV-011-static-0.tr (old trace format)
- AODV-011-static-0.ntr (new trace format)

» Follow Route Request / Reply in scenario



```
AODV-011-static-0.tr (opt/lectnet2/mitarbeiter/fuessler/wdr/lectures/netsim/aux-lecture8/ns) - GVIM1
File Edit Tools Syntax Buffers Window Help
4 s 2.556838879 _0_AGT --- 0 cbr 512 [0 0 0 0] ----- [0:0 8:0 32 0] [0] 0 0
5 r 2.556838879 _0_RTR --- 0 cbr 512 [0 0 0 0] ----- [0:0 8:0 32 0] [0] 0 0
6 s 2.556838879 _0_RTR --- 0 AODV 48 [0 0 0 0] ----- [0:255 -1:255 30 0] [0x2 1 1 [8 0] [0 4]] (REQUEST)
7 s 2.556953879 _0_MAC --- 0 AODV 100 [0 ffffffff 0 800] ----- [0:255 -1:255 30 0] [0x2 1 1 [8 0] [0 4]] (REQUEST)
8 r 2.55779462 _1_MAC --- 0 AODV 48 [0 ffffffff 0 800] ----- [0:255 -1:255 30 0] [0x2 1 1 [8 0] [0 4]] (REQUEST)
9 r 2.55779462 _1_RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [0:255 -1:255 30 0] [0x2 1 1 [8 0] [0 4]] (REQUEST)
10 s 2.558660689 _1_RTR --- 0 AODV 48 [0 ffffffff 0 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
11 s 2.559055689 _1_MAC --- 0 AODV 100 [0 ffffffff 1 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
12 r 2.559846272 _0_MAC --- 0 AODV 48 [0 ffffffff 1 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
13 r 2.559846272 _2_MAC --- 0 AODV 48 [0 ffffffff 1 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
14 r 2.559881272 _0_RTR --- 0 AODV 48 [0 ffffffff 1 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
15 r 2.559881272 _2_RTR --- 0 AODV 48 [0 ffffffff 1 800] ----- [1:255 -1:255 29 0] [0x2 2 1 [8 0] [0 4]] (REQUEST)
16 s 2.569109890 _2_RTR --- 0 AODV 48 [0 ffffffff 1 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
17 s 2.569263890 _2_MAC --- 0 AODV 100 [0 ffffffff 2 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
18 r 2.564086473 _1_MAC --- 0 AODV 48 [0 ffffffff 2 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
19 r 2.564086473 _3_MAC --- 0 AODV 48 [0 ffffffff 2 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
20 r 2.564111473 _1_RTR --- 0 AODV 48 [0 ffffffff 2 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
21 r 2.564111473 _3_RTR --- 0 AODV 48 [0 ffffffff 2 800] ----- [2:255 -1:255 28 0] [0x2 3 1 [8 0] [0 4]] (REQUEST)
22 s 2.570500125 _3_RTR --- 0 AODV 48 [0 ffffffff 2 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
23 s 2.570835125 _3_MAC --- 0 AODV 100 [0 ffffffff 3 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
24 r 2.571635708 _2_MAC --- 0 AODV 48 [0 ffffffff 3 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
25 r 2.571635708 _4_MAC --- 0 AODV 48 [0 ffffffff 3 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
26 r 2.571660708 _2_RTR --- 0 AODV 48 [0 ffffffff 3 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
27 r 2.571660708 _4_RTR --- 0 AODV 48 [0 ffffffff 3 800] ----- [3:255 -1:255 27 0] [0x2 4 1 [8 0] [0 4]] (REQUEST)
28 s 2.572302162 _4_RTR --- 0 AODV 48 [0 ffffffff 3 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
29 s 2.573917162 _4_MAC --- 0 AODV 100 [0 ffffffff 4 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
30 r 2.57417745 _3_MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
31 r 2.57417745 _5_MAC --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
32 r 2.574742745 _3_RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
33 r 2.574742745 _5_RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [4:255 -1:255 26 0] [0x2 5 1 [8 0] [0 4]] (REQUEST)
34 s 2.575880828 _5_RTR --- 0 AODV 48 [0 ffffffff 4 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
35 s 2.576195828 _5_MAC --- 0 AODV 100 [0 ffffffff 5 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
36 r 2.576996411 _4_MAC --- 0 AODV 48 [0 ffffffff 5 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
37 r 2.576996411 _6_MAC --- 0 AODV 48 [0 ffffffff 5 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
38 r 2.577021411 _4_RTR --- 0 AODV 48 [0 ffffffff 5 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
39 r 2.577021411 _6_RTR --- 0 AODV 48 [0 ffffffff 5 800] ----- [5:255 -1:255 25 0] [0x2 6 1 [8 0] [0 4]] (REQUEST)
40 s 2.578769464 _6_RTR --- 0 AODV 48 [0 ffffffff 5 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
41 s 2.57818464 _6_MAC --- 0 AODV 100 [0 ffffffff 6 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
42 r 2.579985047 _5_MAC --- 0 AODV 48 [0 ffffffff 6 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
43 r 2.579985047 _7_MAC --- 0 AODV 48 [0 ffffffff 6 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
44 r 2.580010047 _5_RTR --- 0 AODV 48 [0 ffffffff 6 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
45 r 2.580010047 _7_RTR --- 0 AODV 48 [0 ffffffff 6 800] ----- [6:255 -1:255 24 0] [0x2 7 1 [8 0] [0 4]] (REQUEST)
46 s 2.582497457 _7_RTR --- 0 AODV 48 [0 ffffffff 6 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
47 s 2.583792457 _7_MAC --- 0 AODV 100 [0 ffffffff 7 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
48 s 2.584593041 _6_MAC --- 0 AODV 48 [0 ffffffff 7 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
49 r 2.584593041 _8_MAC --- 0 AODV 48 [0 ffffffff 7 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
50 r 2.584618041 _6_RTR --- 0 AODV 48 [0 ffffffff 7 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
51 r 2.584618041 _8_RTR --- 0 AODV 48 [0 ffffffff 7 800] ----- [7:255 -1:255 23 0] [0x2 8 1 [8 0] [0 4]] (REQUEST)
```


Evaluation

» **Typical metrics for evaluation of “Routing Protocol Performance” (see also RFC 2501):**

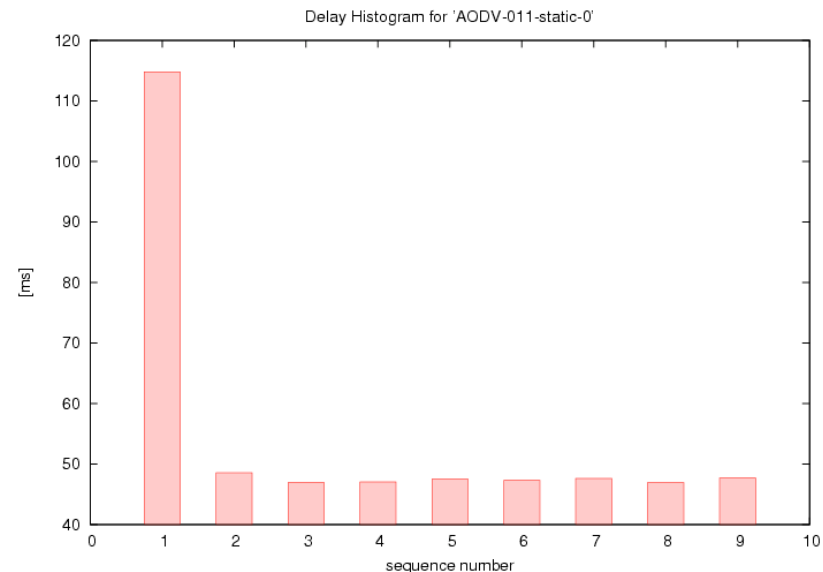
- **Packet Delivery Ratio (PDR)**
- **(avg) End-2-End Delay / Route Acquisition Time**
- **Overhead / Cost**
 - on Routing / MAC Layer
 - overall / per packet or payload bit

» **Warning: Do not generate a wrong feeling of linearity!**

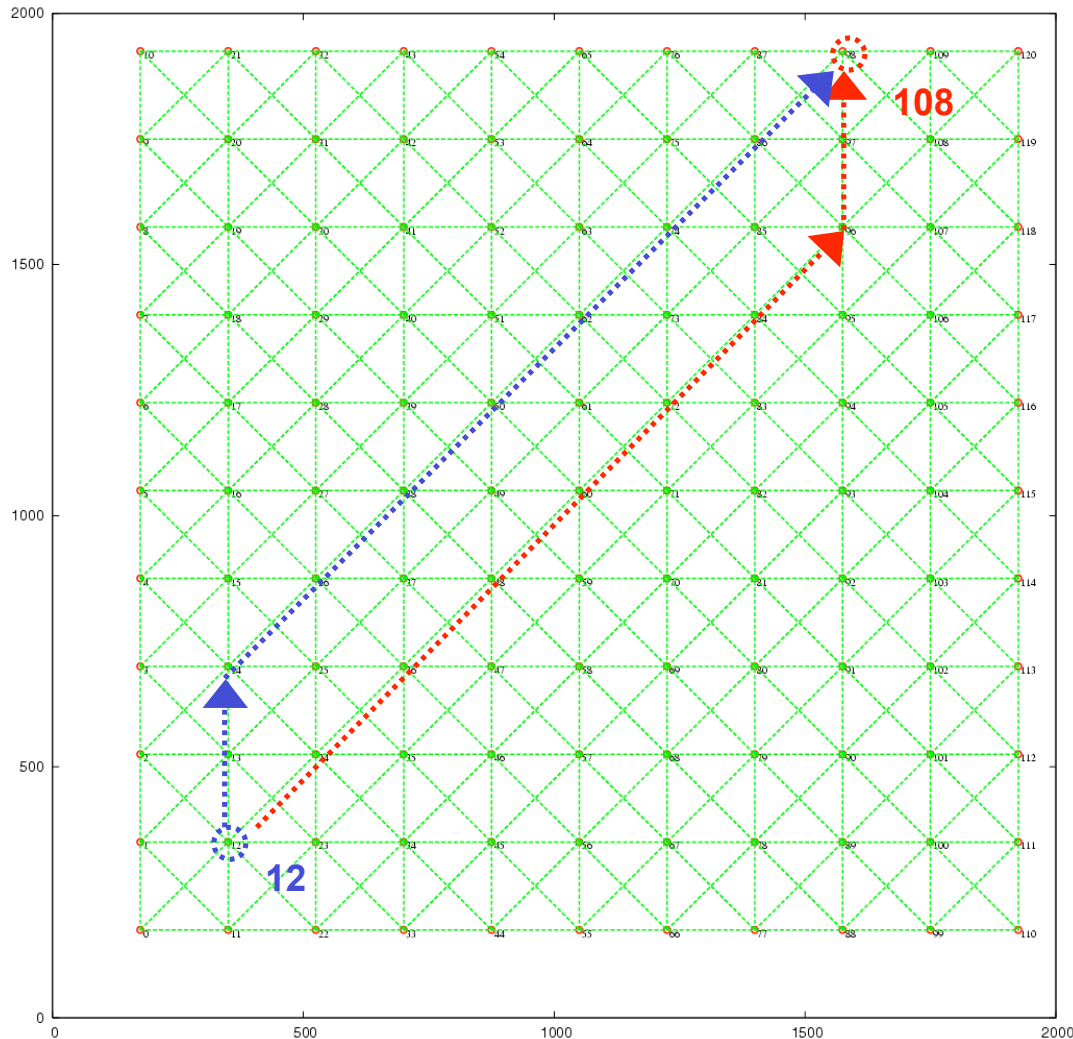
- **Histogram or Function Plot (Interpolation)?**
- **Absolute Values per Simulation Run / per Packet?**

Evaluation (End-2-End Delay Line Scenario)

- » **Histogram shows the end-2-end-packet delay for the nine packets sent**
- » **„First Packet Delay“ higher because of „Route Acquisition Time“**
- » **Remaining Packets fairly stable**



The Grid Scenario (Setup)



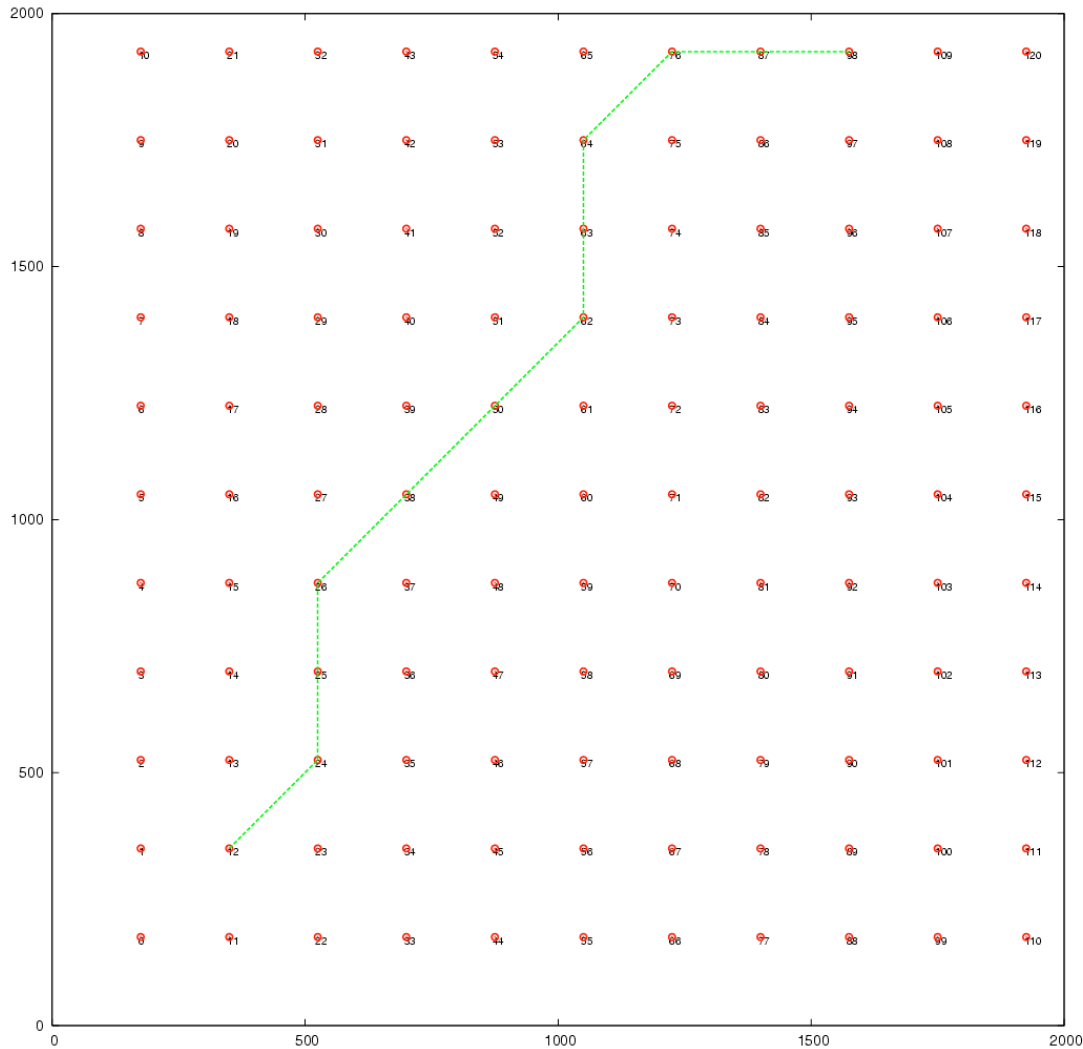
» $11 \times 11 = 121$ nodes (0..120)

» CBR: 12 → 108

» Which are the shortest routes to be expected?

» Hop Length: 9

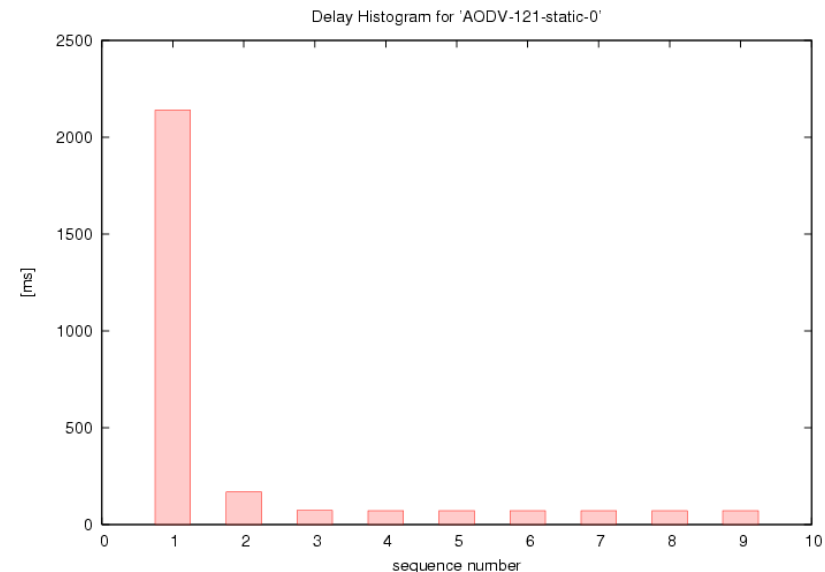
The Grid Scenario (Route Taken)



- » Why is that so?
- » AODV immediately answers the RREQ, even if it's suboptimal
- » Protocol Variant: Answer Each RREQ and improve route

Evaluation (End-2-End Delay Grid Scenario)

- » Histogram shows the end-2-end-packet delay for the nine packets sent
- » „First Packet Delay“ higher because of „Route Acquisition Time“
- » Remaining Packets fairly stable
- » But: „First Packet Delay“ one order of magnitude higher than in line scenario



There is still more...

- » **play with downloadable examples**
- » **parser scripts with different purposes**
- » **have a look at the trace files**

Wrap-Up

- » **Basics on Mobile Ad-Hoc Network Routing**
 - **Challenges**
 - **Classification of Algorithms**
- » **Learned about the wireless extensions of ns-2**
- » **Created simple AODV simulations**
- » **Learned to read the wireless traces**

References

- » The ns-2 user manual <http://www.isi.edu/nsnam/ns>
- » S. Corson, J. Macker: RFC 2501: Mobile Ad-Hoc Networking (MANET) Routing Protocol Performance Issues and Evaluation Considerations
- » C. Perkins, E. Royer “Ad-Hoc On-Demand Distance-Vector Routing”

In " Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA) ", pp. 90-100, New Orleans, LA, February 1999
- » C.Perkins, E. Belding-Royer, Samir Das: RFC 3561: Ad-Hoc On-Demand Distance-Vector (AODV) Routing, IETF
- » Samir R. Das, Charles E. Perkins, Elizabeth M. Royer, “Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks”, IEEE Infocom 2000, Tel Aviv, Israel, March 2000