A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols

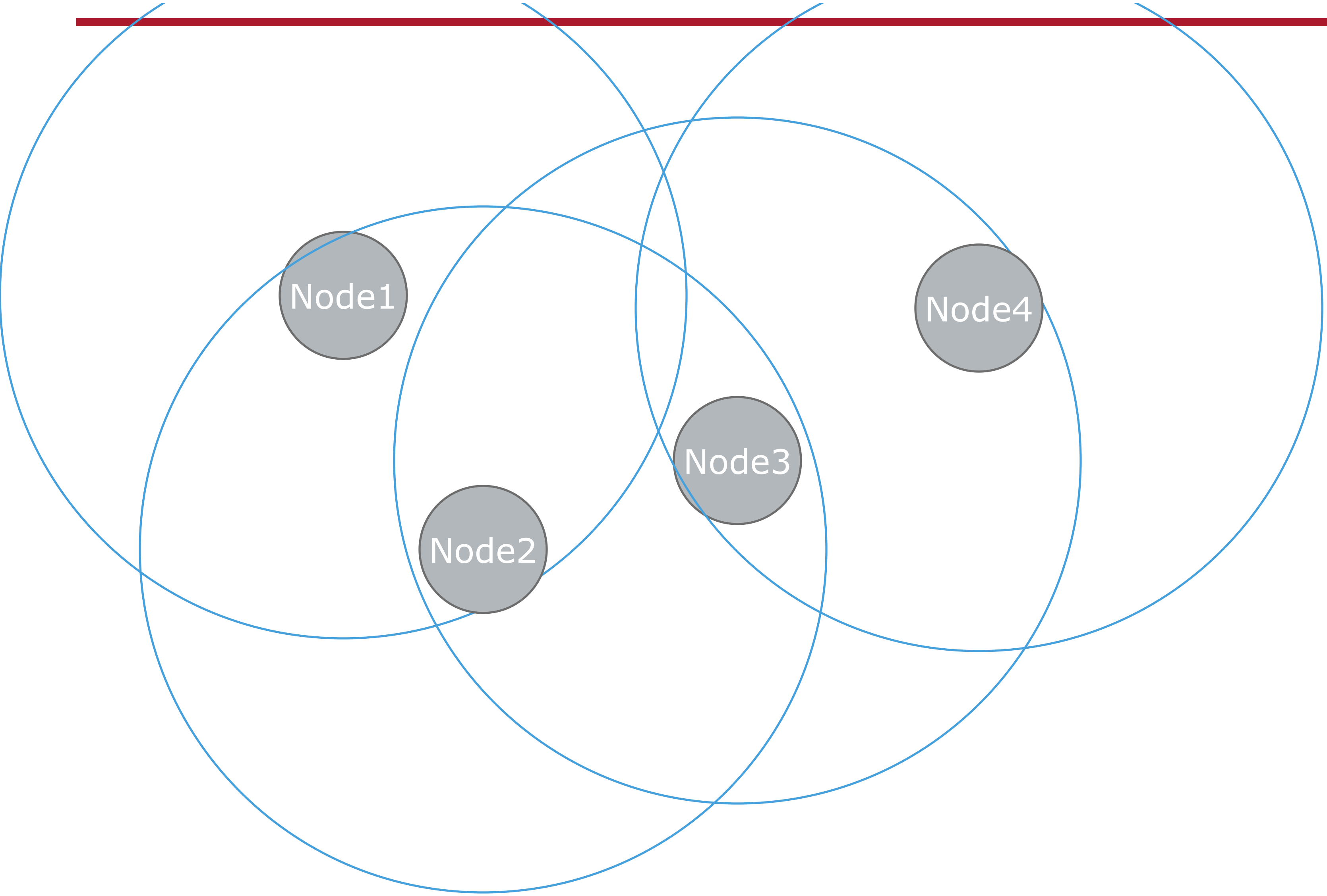
Broch et al

Presented by Brian Card

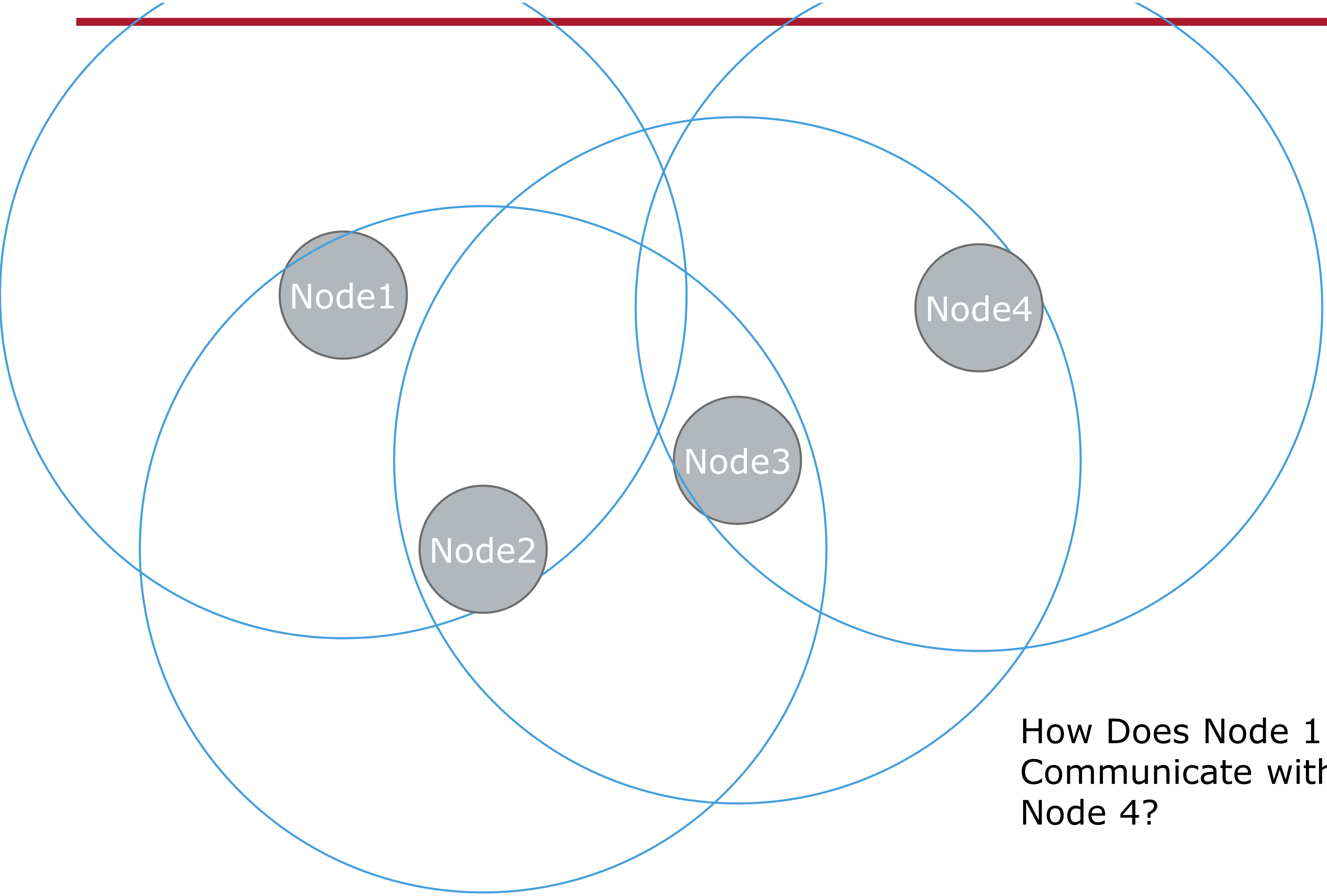
Outline

- Introduction
- NS enhancements
- Protocols:
  - DSDV
  - TORA
  - DRS
  - AODV
- Evaluation
- Conclusions

Introduction

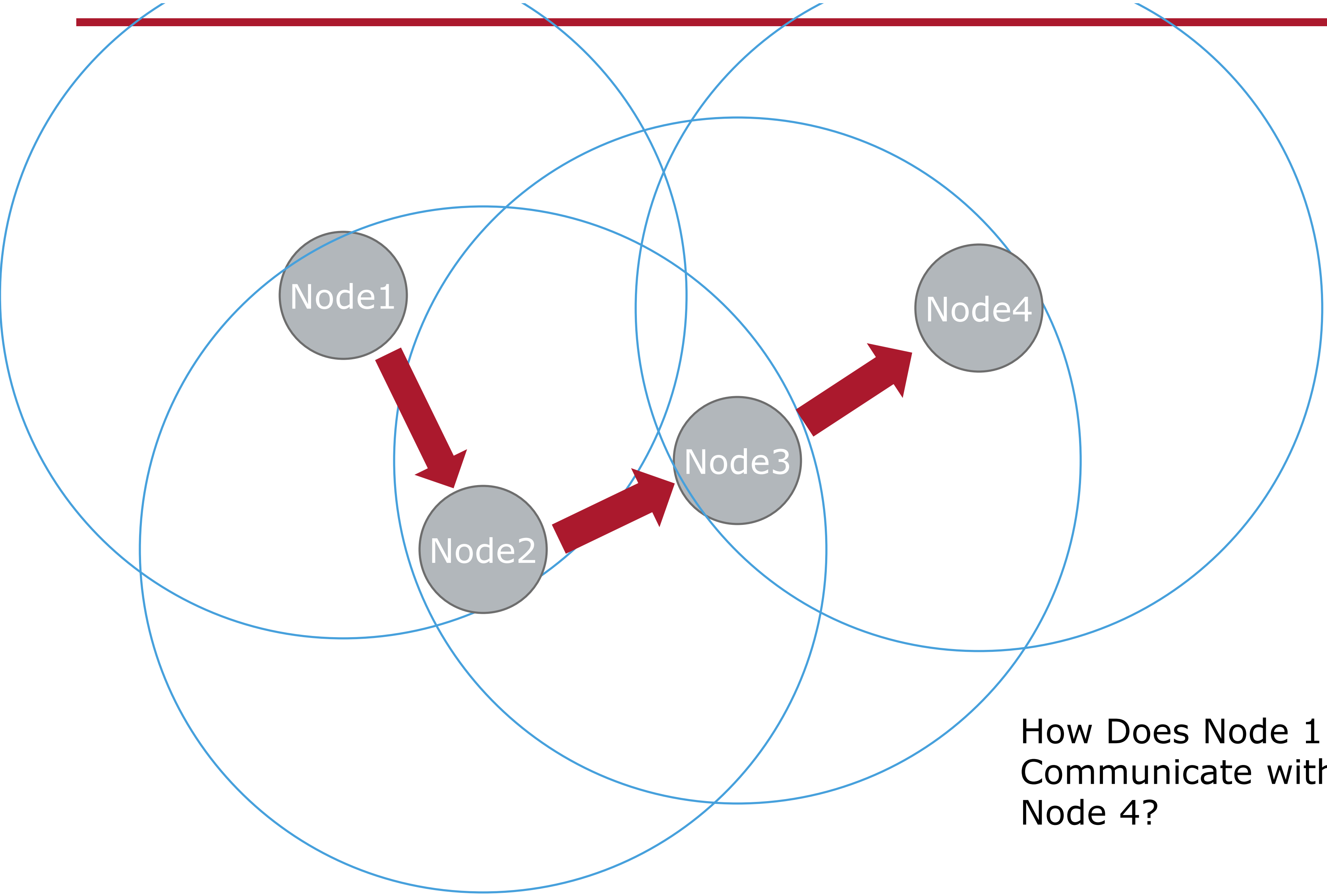


Introduction



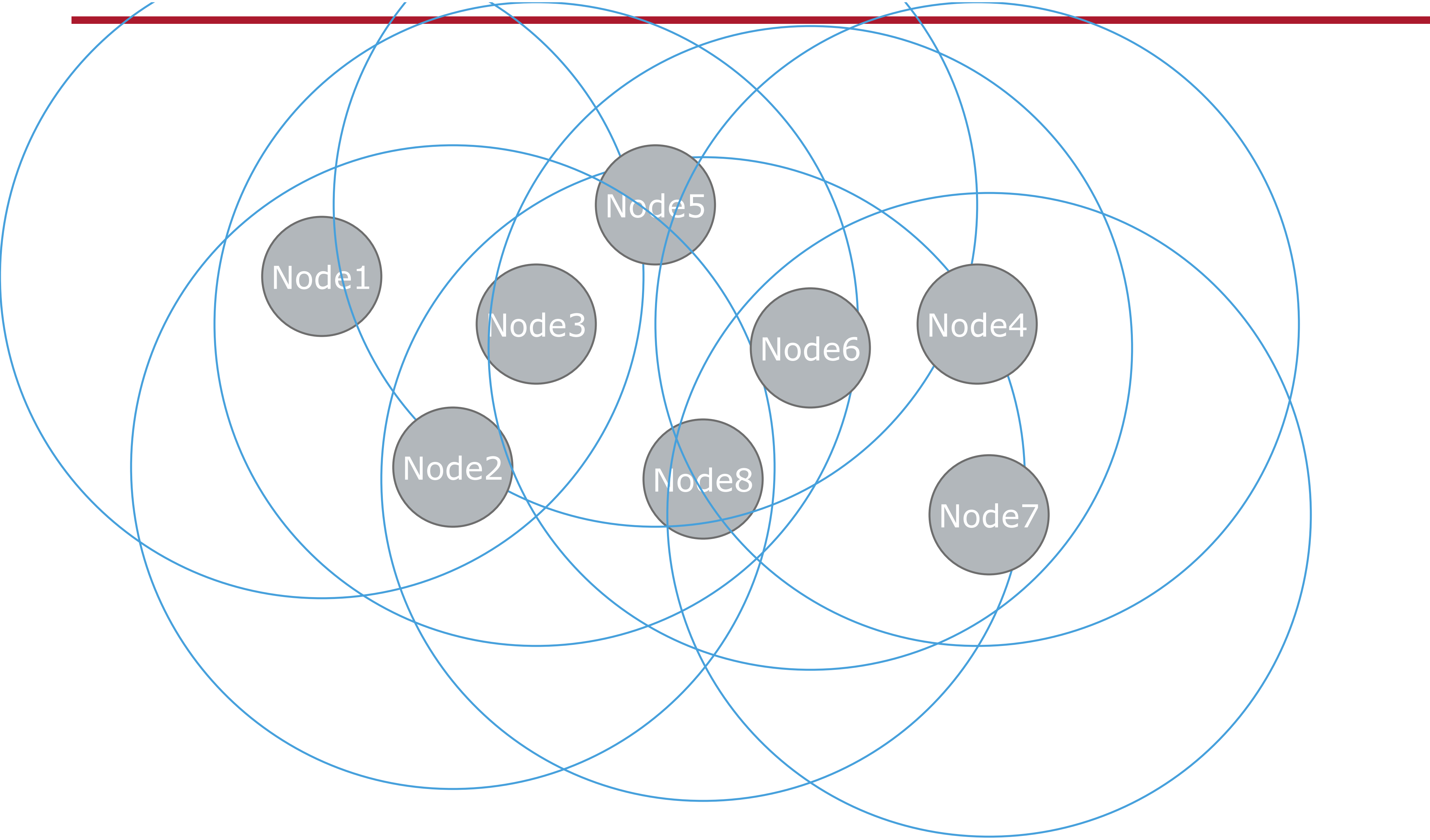
How Does Node 1 Communicate with Node 4?

Introduction

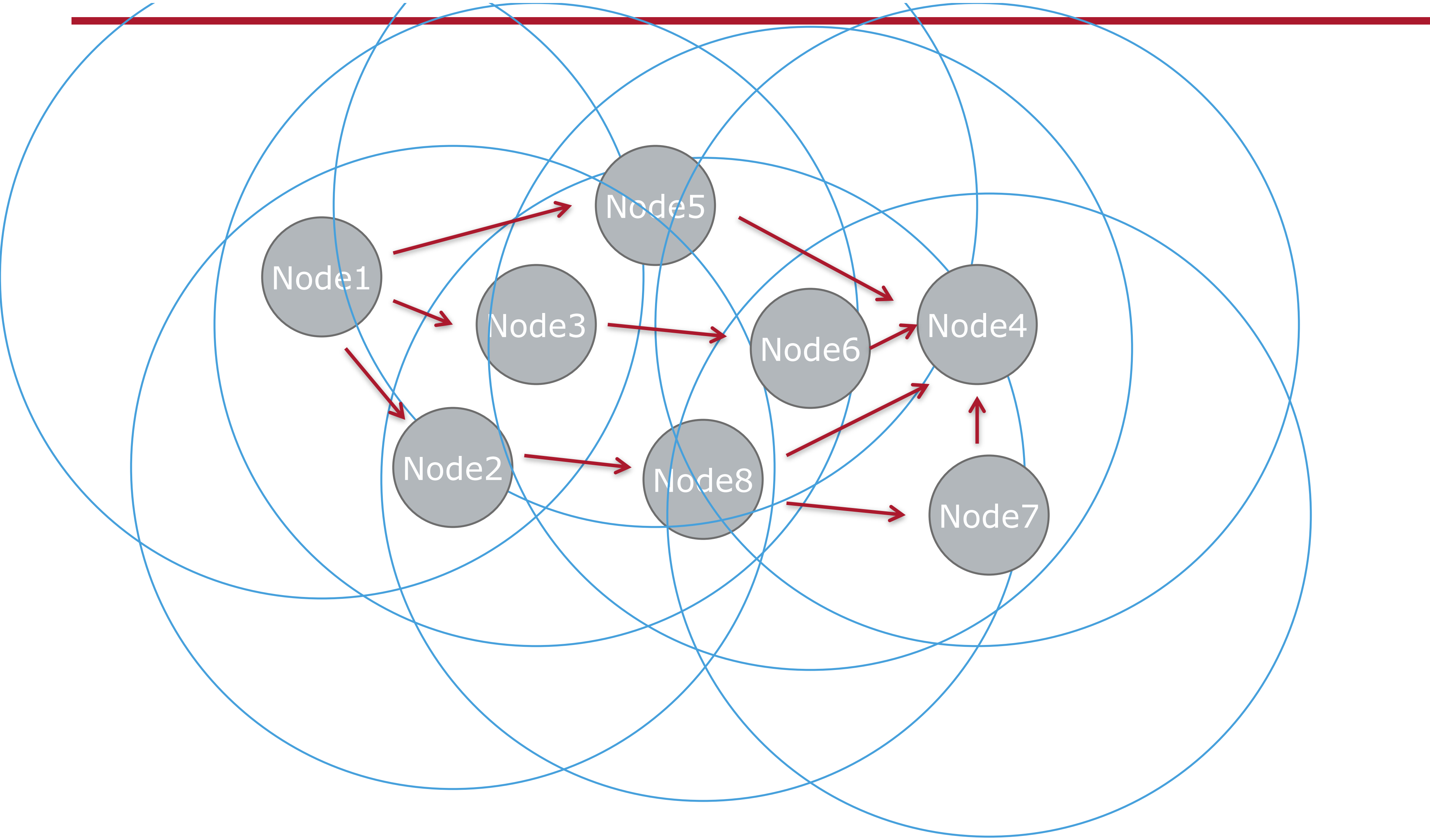


How Does Node 1 Communicate with Node 4?

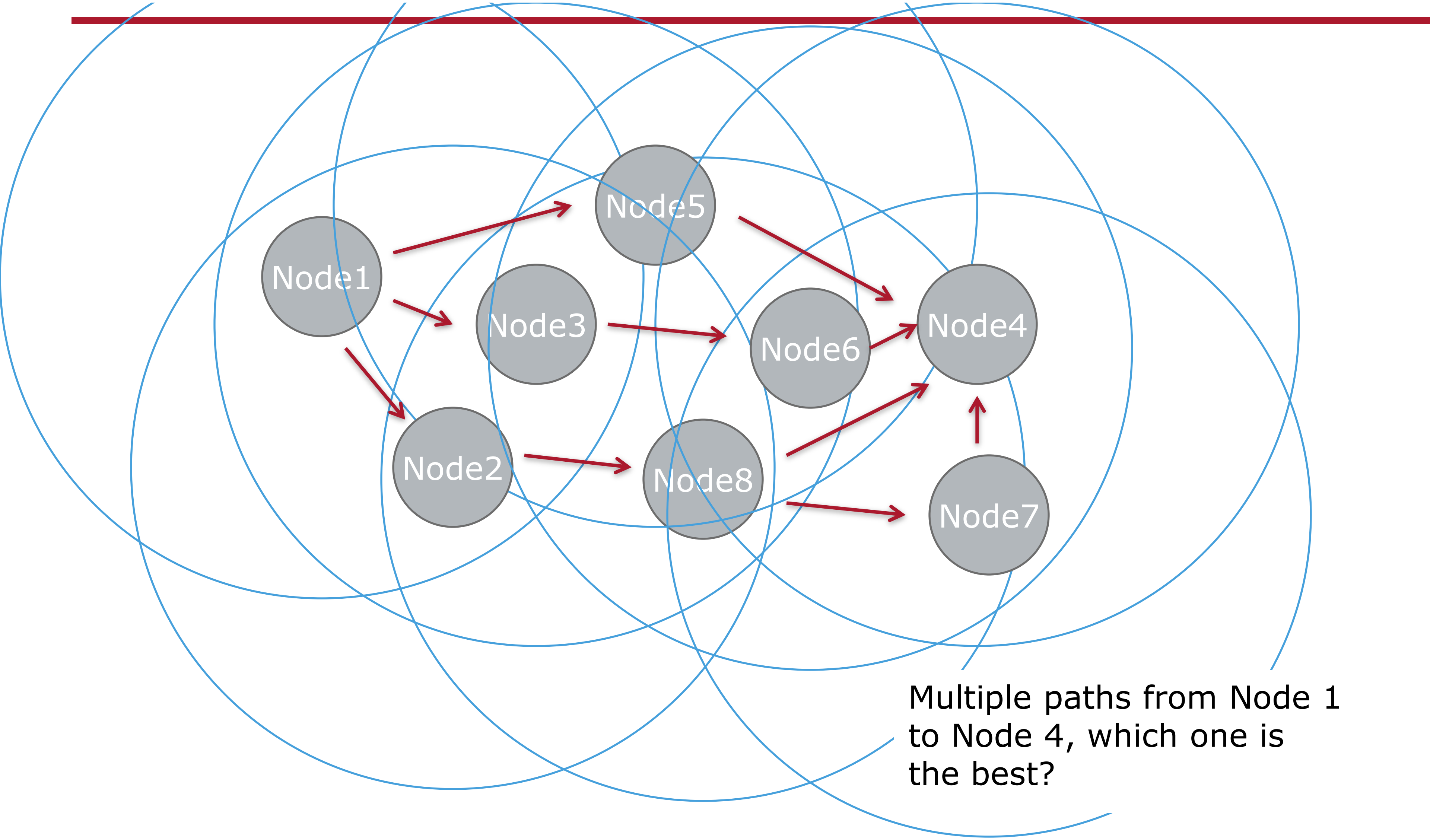
What if the network looks like this?



What if the network looks like this?

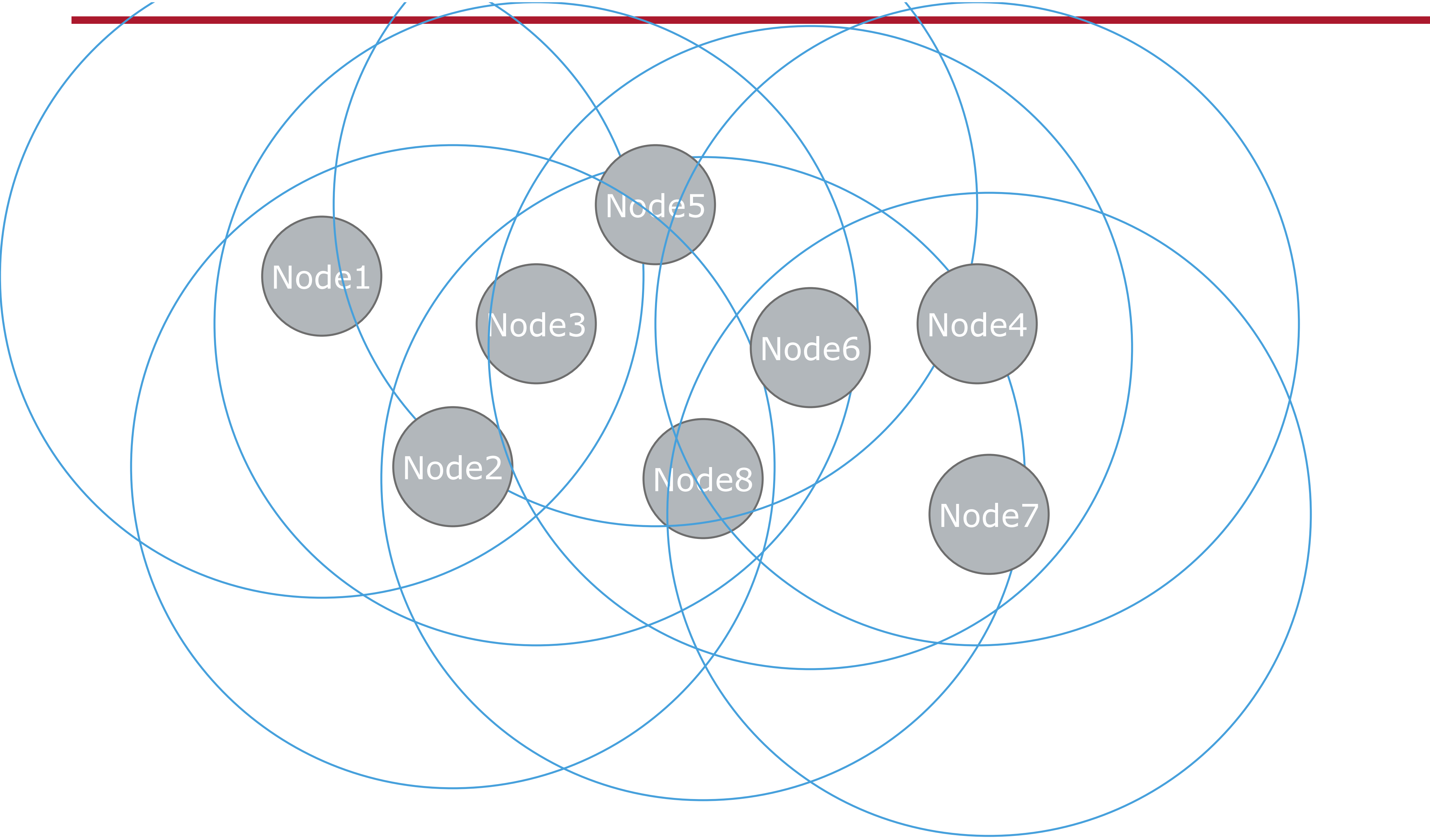


What if the network looks like this?



Multiple paths from Node 1 to Node 4, which one is the best?

What if the network is mobile?



What if the network is mobile?

- Need intelligent routing between nodes

Mobile Ah-Hoc Networks

- Hop between nodes when point to point communication is not possible
- Nodes can leave and join the network at any time
- Link characteristics between nodes unpredictable
- Nodes may move!
  - In and out of range
  - Can cause variations in link characteristics

Protocols for Ad-Hoc Mobile Networks

- Need to quickly and accurately find routes to different nodes
- Need to be able to recalculate based on changing node positions or changes in link characteristics
- Need to be efficient

Issues with Protocols for Ad-Hoc Mobile Networks

- Several protocols already exist, how do we know which one to choose?
  - No performance evaluation comparing protocols
- Simulation tools don’t accurately model mobile networks
  - No support for physical layer characteristics
  - No support for MAC layer
  - No support for node positions
- This paper attempts to address these issues

Outline

- Introduction
- **NS enhancements**
- Protocols:
  - DSDV
  - TORA
  - DRS
  - AODV
- Evaluation
- Conclusions

NS Enhancements

- NS (Network Simulator) is a discrete event simulator widely used for network performance evaluation
- Extensive support for simulating TCP
- No support for Wi-Fi MAC layer or physical layer
- No position information

Physical Layer Additions to NS

- $1/r^2$ attenuation model within reference distance (100m), $1/r^4$ attenuation model afterwards
- Movement is modeled using position as a function of time using flat surface or topographical map
- Power is tracked for each interface, when model predicts power is lower than receive threshold the packet is marked as dropped in error
- Carrier sensing threshold is used to treat low power transmissions as noise
- Propagation delay is also accounted for

MAC Layer additions to NS

- Physical lay feed packets to MAC Layer
- *virtual carrier sensing* is used at the MAC layer (RTS/CTS)
- ACK packets are transmitted for unicast packets, retransmits occur from sender until ACKs are received

Other NS Updates

- ARP (Address Resolution Protocol) is used for determining link-layer IP addresses
  - This is important because ARP REQUEST is broadcast and can interact with protocols
- Each node has a 50 packet send queue. Drop-tail is used for queue management

Outline

- Introduction
- NS enhancements
- **Protocols:**
  - DSDV
  - TORA
  - DRS
  - AODV
- Evaluation
- Conclusions

Protocols

- Authors implemented 4 different routing protocols
- Some changes were made to the protocols to improve performance
- The following changes were made to all of them:
  – Broadcasts and broadcast responses were jittered using a random delay between 0 and 10 ms to prevent synchronization
  – Routing packets were transmitted before data or ARP packets
    ▪ This was to ensure that routing information propagated quickly
  – Link breakage was detected at the MAC layer except for DSDV

DSDV: Destination-Sequenced Distance Vector

- Hop by hop distance vector routing protocol
- Each node keeps a routing table with three fields for each destination:
  - Next hop
  - Sequence number
  - Metric
- Routers are chosen based on sequence number and metric
- *Higher* sequence number (newer route) wins first
- Afterwards *lower* metric wins

DSDV: Destination-Sequenced Distance Vector

- Nodes are periodically sending out sequence numbers which represent the ‘freshness’ of a link
- When a link is broken, the nodes marks the metric as infinite
- This causes routes to avoid that node
- When the node comes back up, a new sequence number is generated and packets flow over the new link

DSDV Implementation

- MAC protocol link breakages are not used
  - Authors noted when using MAC level breakages if a single link is broken the node becomes unreachable
  - Sequence number from the breakage becomes higher than other sequence numbers and becomes the preferred route
  - This causes the node to be completely unreachable (packet drops) until it can advertise and create a new sequence number

DSDV and DSDV-SQ

- Original protocol description is ambiguous about when to send updates
- Authors use an additional scheme they call DSDV-SQ (SQ for sequence number) which also sends out updates when a sequence number changes
- This increases overhead, but provides better performance since broken links are detected sooner
- Authors use this for all experiments and provide a comparison to DSDV at the end of the paper

DSDV Constants

Table I Constants used in the DSDV-SQ simulation.

Periodic route update interval	15 s
Periodic updates missed before link declared broken	3
Initial triggered update weighted settling time	6 s
Weighted settling time weighting factor	7/8
Route advertisement aggregation time	1 s
Maximum packets buffered per node per destination	5

TORA: Temporarily-Ordered Routing Algorithm

- Routes are discovered on-demand
- Network is modeled like a system of pipes with the packets being water in the pipes
- Protocol is layered on top of IMEP to provide guaranteed in-order packet delivery
  - Other protocols do not require this
- IMEP can be used for address resolution but the authors did not use this and used ARP for all protocols
- IMEP also groups TORA and IMEP control messages into blocks called ‘object blocks’

TORA Basic Usage

- QUERY packet broadcasted when a packet needs to be delivered to some address.
- Packet moves through the network until it reaches the destination or a node that can route to the destination
- When a QUERY packet is received an UPDATE packet is then sent with the node's *height* with respect to that destination
  - Height is used to calculate the flow parameters
  - Greater height indicates more resistance
- Each node that receives an UPDATE packet then adjusts it's own height for that destination to be larger than the value in the UPDATE packet
- When a link is broken, the height it updated to a local maximum and an UPDATE packet is sent out

Implementation

- TORA sensitive to intervals used for IMEP ‘object blocks’, no guidance given by specification with respect to these parameters
  - authors chose 150-250ms
- TORA nodes must have an accurate picture of the network
  - In order guaranteed delivery very important
  - If A can’t reach B then B must also think that it can’t reach A

TORA Constants

Table II Constants used in the TORA simulation.

BEACON period	1 s
Time after which a link is declared down if no BEACON or HELLO packets were exchanged	3 s
Time after which an object block is retransmitted if no acknowledgment is received	500 ms
Time after which an object block is not retransmitted and the link to the destination is declared down	1500 ms
Min HELLO and ACK aggregation delay	150 ms
Max HELLO and ACK aggregation delay	250 ms

DSR: Dynamic Source Routing

- Each packed contains the entire route needed to deliver the packet
- Each node does not maintain up to date routing information
  - No route advertisements that are used in other protocols

DSR Basic Usage

- When a packet needs to be sent a ROUTE REQUEST is broadcasted
  - Either the destination node or another node that knows how to get to the destination respond with a ROUTE REPLY
  - Nodes cache messages and use them to aggressively limit the spread of ROUTE REQUEST messages
  - This process is called Route Discovery
- When network topology changes, a ROUTE ERROR is used to indicate a broken link
  - Used to invalidate caches
  - This process is called Route Maintenance

DSR Implementation

- Only support bi-directional links
  - ROUTE REPLY packets traverse same links the ROUTE REQUESTS were sent over
- The first time a ROUTE REQUEST is made, send it to only the neighbor nodes
  - This reduces network usage and allows a sender to query the caches of it's neighbors and optimize for the use case where the destination is in range
  - If nothing comes back, re-broadcast and allow propagation.
- All nodes scan for ROUTE ERRORs in promiscuous mode
  - Also if a node hears a packet and it can route to the destination, it sends a pre-emptive ROUTE REPLY
- Finally, routers will change the route if it knows the next hop is not available and it has another path in it's cache

DRS Constants

Table III Constants used in the DSR simulation.

Time between retransmitted ROUTE REQUESTS (exponentially backed off)	500 ms
Size of source route header carrying n addresses	$4n + 4$ bytes
Timeout for nonpropagating search	30 ms
Time to hold packets awaiting routes	30 s
Max rate for sending gratuitous REPLYs for a route	1/s

AODV: Ad Hoc On-Demand Distance Vector

- Combination of DSR and DSDV
  - Combines Route Discovery and Route Maintenance from DSR
  - With hop-by-hop routing, sequence numbers and beacons from DSDV
- Creates both forward and reverse routes from nodes when ROUTE REQUESTs are sent out
- Nodes only remember the next hop and not the entire route
- Periodic HELLO messages are broadcasted by nodes, if a node misses 3 HELLOs from a neighbor the node is marked down, and this state is broadcasted

AODV Implementation

- Authors created variation called AODV-LL which uses the link layer to detect broken links
  - Removes overhead from periodic HELLO messages, but broken links can only be detected on demand!
- AODV-LL performs slightly better than AODV
- Changed ROUTE REPLY timeout from 120 seconds to 6 seconds
  - Protocol reacts to dropped packets much faster with this lower timeout

AODV-LL Constants

Table IV Constants used in the AODV-LL simulation.

Time for which a route is considered active	300 s
Lifetime on a ROUTE REPLY sent by destination node	600 s
Number of times a ROUTE REQUEST is retried	3
Time before a ROUTE REQUEST is retried	6 s
Time for which the broadcast id for a forwarded ROUTE REQUEST is kept	3 s
Time for which reverse route information for a ROUTE REPLY is kept	3 s
Time before broken link is deleted from routing table	3 s
MAC layer link breakage detection	yes

Outline

- Introduction
- NS enhancements
- Protocols:
  - DSDV
  - TORA
  - DRS
  - AODV
- **Evaluation**
- Conclusions

Experimental Setup

- Major component of the paper is to test how protocols react with moving nodes and physical layer / MAC simulations
- 50 nodes for a 900 second simulation
- Rectangular area to test longer routes

- Generate 210 different *scenarios*, run each algorithm against each scenario and compare results

Experimental Scenarios

- Each scenario was a pre-recorded sequence of events
- Nodes switched between being stationary and moving, stationary time was called *pause time*
  - 7 different pause times: 0, 30, 60, 120, 600, 900
  - 0 means constantly moving, 900 is no movement
- 10 randomly generated movement patterns for each pause time
- 20 meters/sec max speed, 10 meter/sec avg speed

Data Sources

- Varied the number of sources from 10, 20, 30
- Packet sizes of 64 bytes or 1024 bytes
- 4 packets per second
- All sources use UDP traffic transmitted at constant bit rates
- 3 sets of sources X 70 movement patterns = 210 scenarios
- No TCP sources

Measured Shorted Path Lengths

- Simulation software measures the number of hops for each path for each scenario
- Changing speed has little effect on number of hops
- 2.6 hops on average

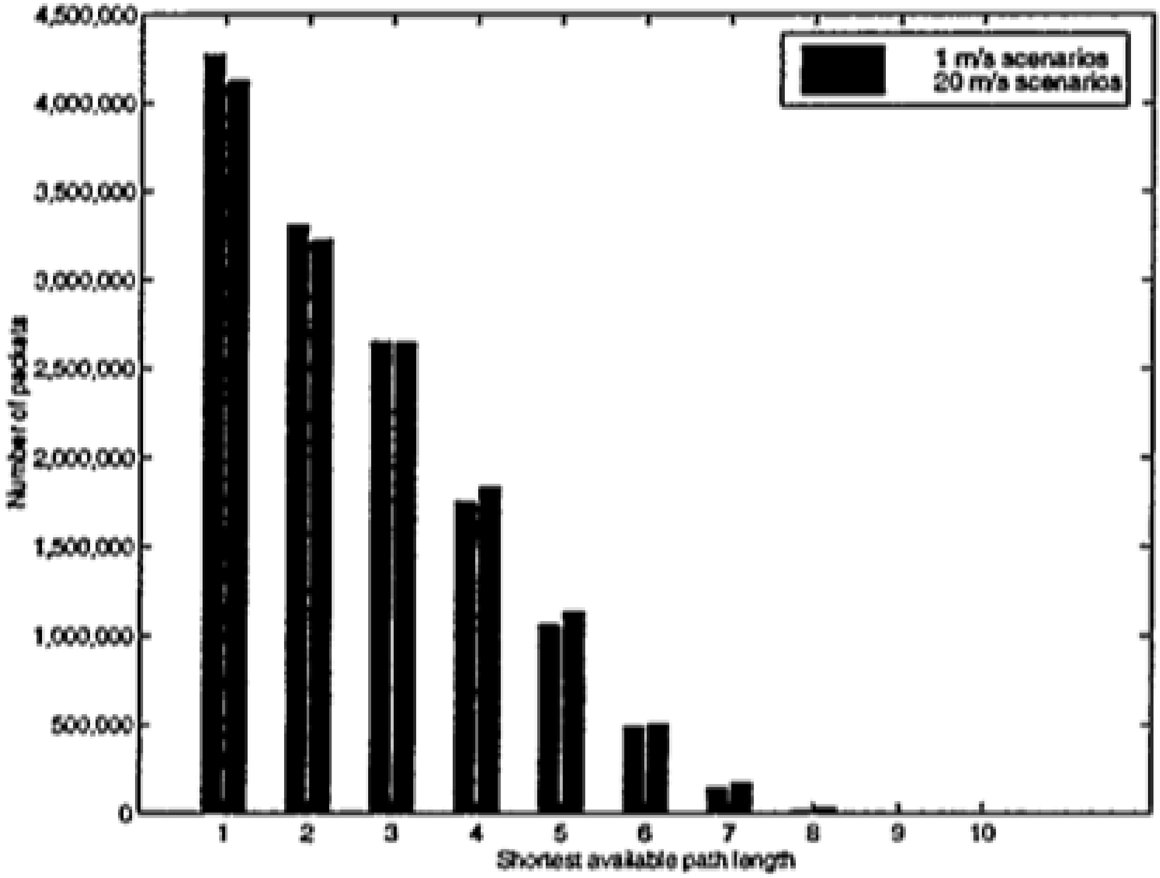


Figure 1 Distribution of the shortest path available to each application packet originated over all scenarios.

Measured Shorted Path Lengths

- Simulation software measures the number of hops for each path for each scenario
- Changing speed has little effect on number of hops
- 2.6 hops on average

Number of hops for 20 m/s vs 1 m/s is about the same

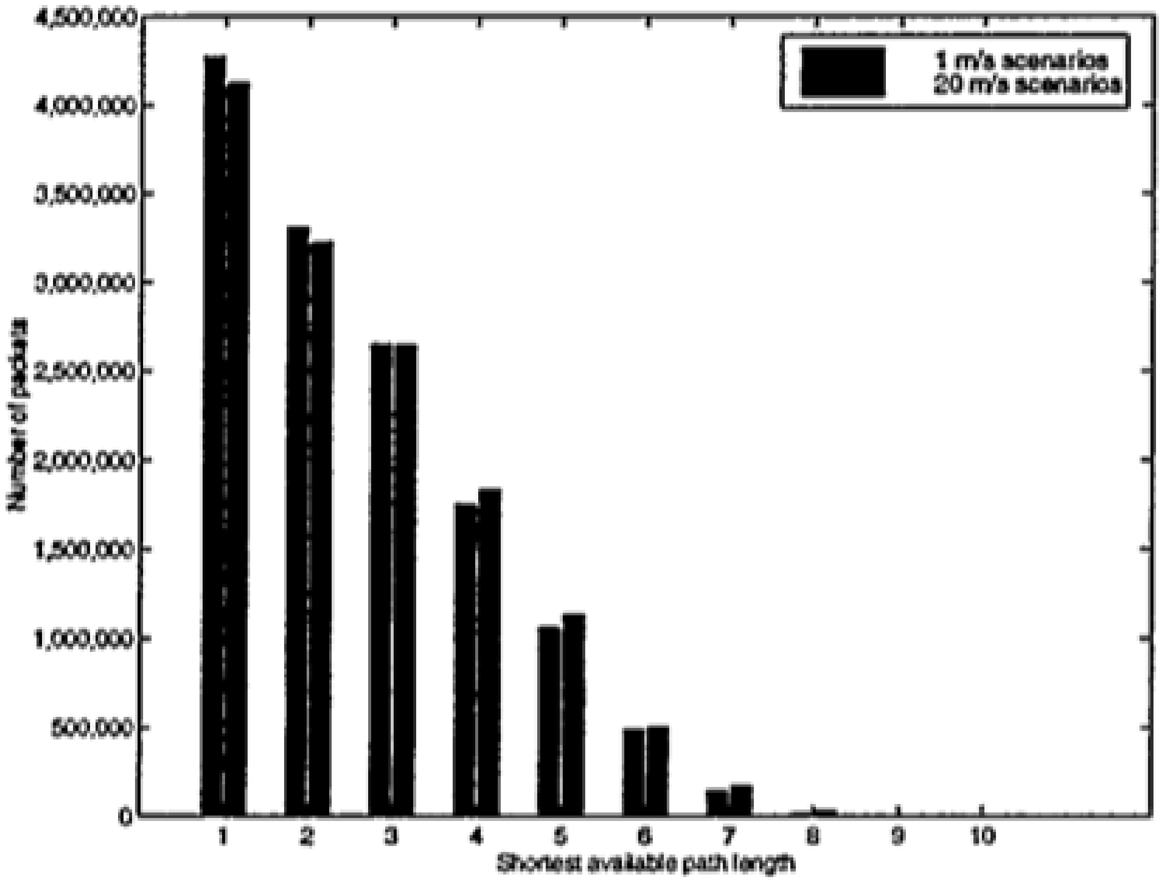


Figure 1 Distribution of the shortest path available to each application packet originated over all scenarios.

Link Connectivity Changes

- Number of times that a node goes in or out of range of another node

Table V Average number of link connectivity changes during each 900-second simulation as a function of pause time.

Pause Time	# of Connectivity Changes	
	1 m/s	20 m/s
0	898	11857
30	908	8984
60	792	7738
120	732	5390
300	512	2428
600	245	1270
900	0	0

Routing Overhead

- “Total number of packets transmitted during the simulation. For packets sent over multiple hops, *each* transmission of the packet (each hop) counts as one transmission”

Routing Overhead

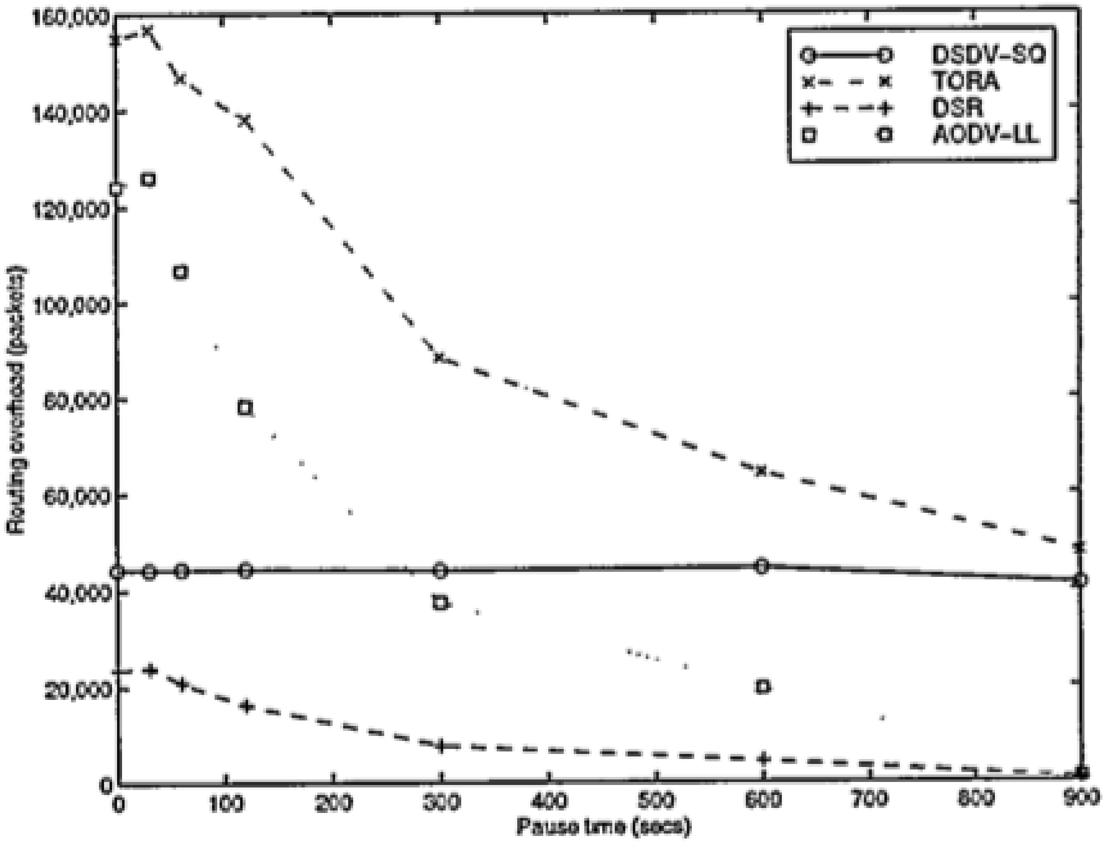


Figure 3 Comparison between the four protocols of the number of routing packets sent (routing overhead) as a function of pause time. Pause time 0 represents constant mobility.

Routing Overhead

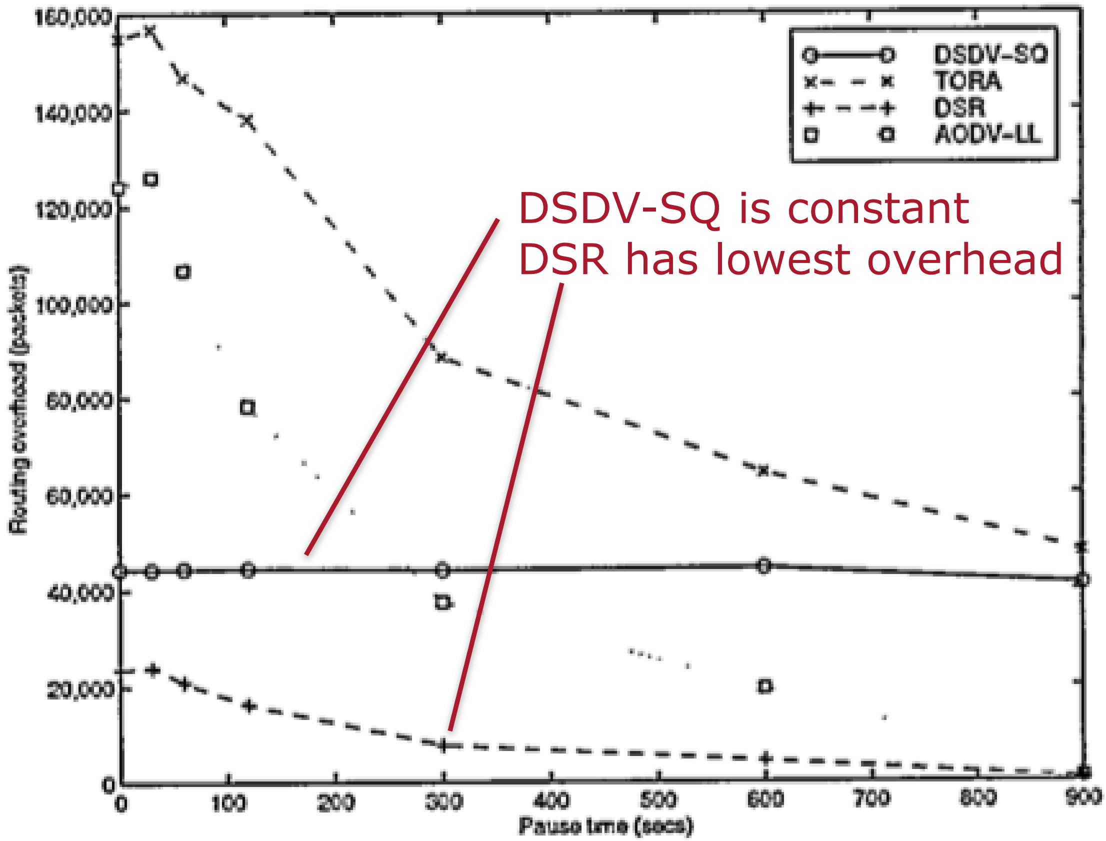


Figure 3 Comparison between the four protocols of the number of routing packets sent (routing overhead) as a function of pause time. Pause time 0 represents constant mobility.

Packet Delivery Ratio

- ratio between number of packets originated by the application layer CBR sources and the number of packets received at the destination. Higher is better.

Packet Delivery Ratio

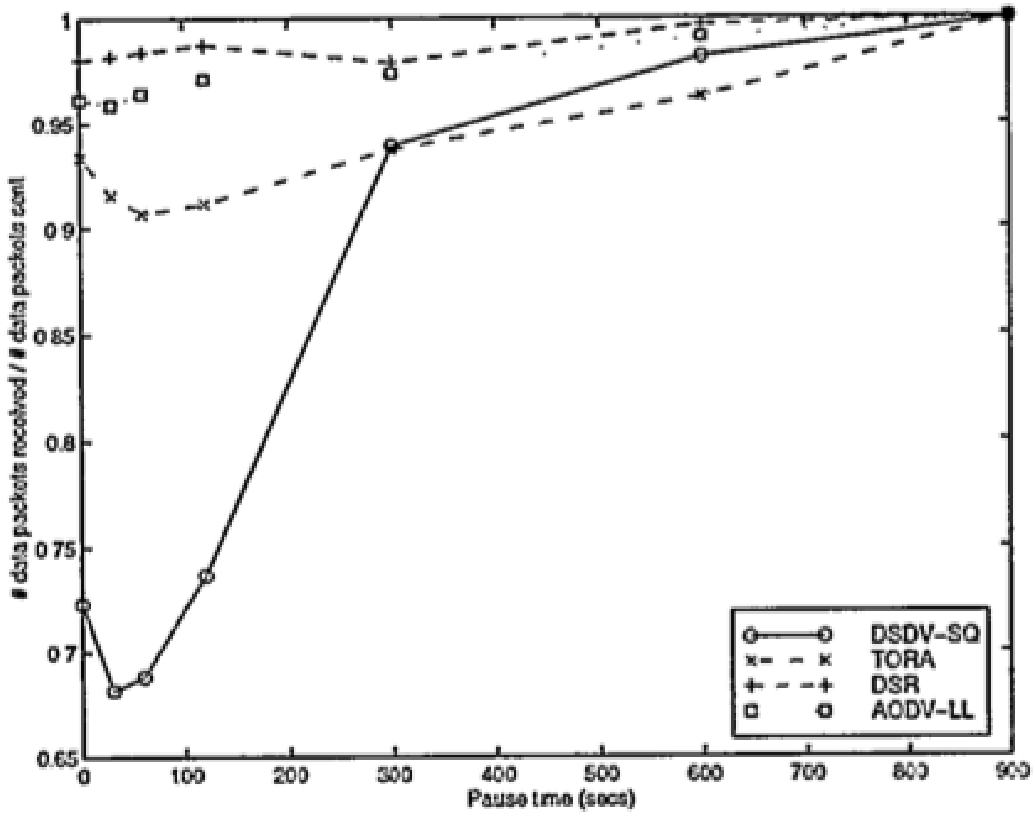
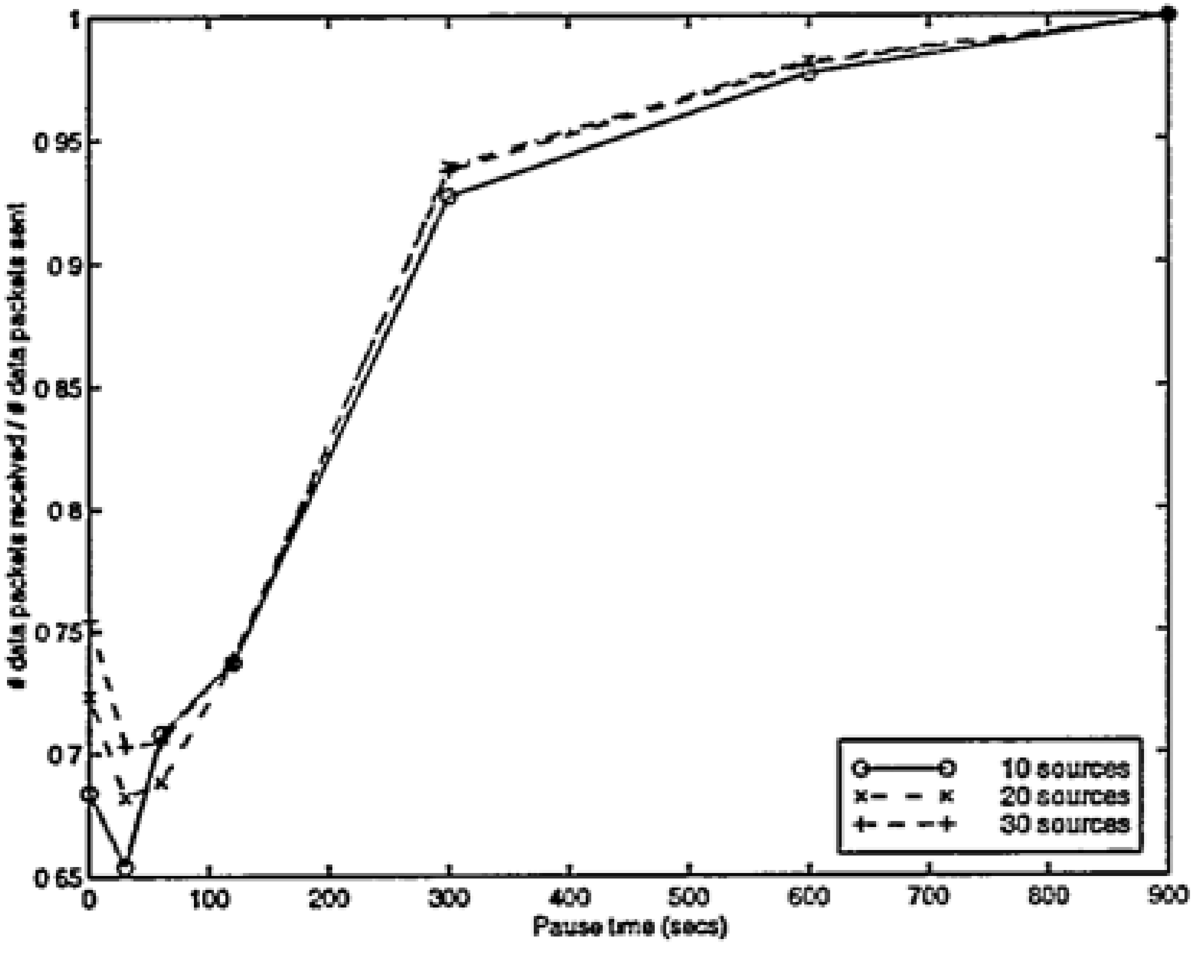


Figure 2 Comparison between the four protocols of the fraction of application data packets successfully delivered (packet delivery ratio) as a function of pause time. Pause time 0 represents constant mobility.

Packet Delivery Ratio – Varying the Number of Sources

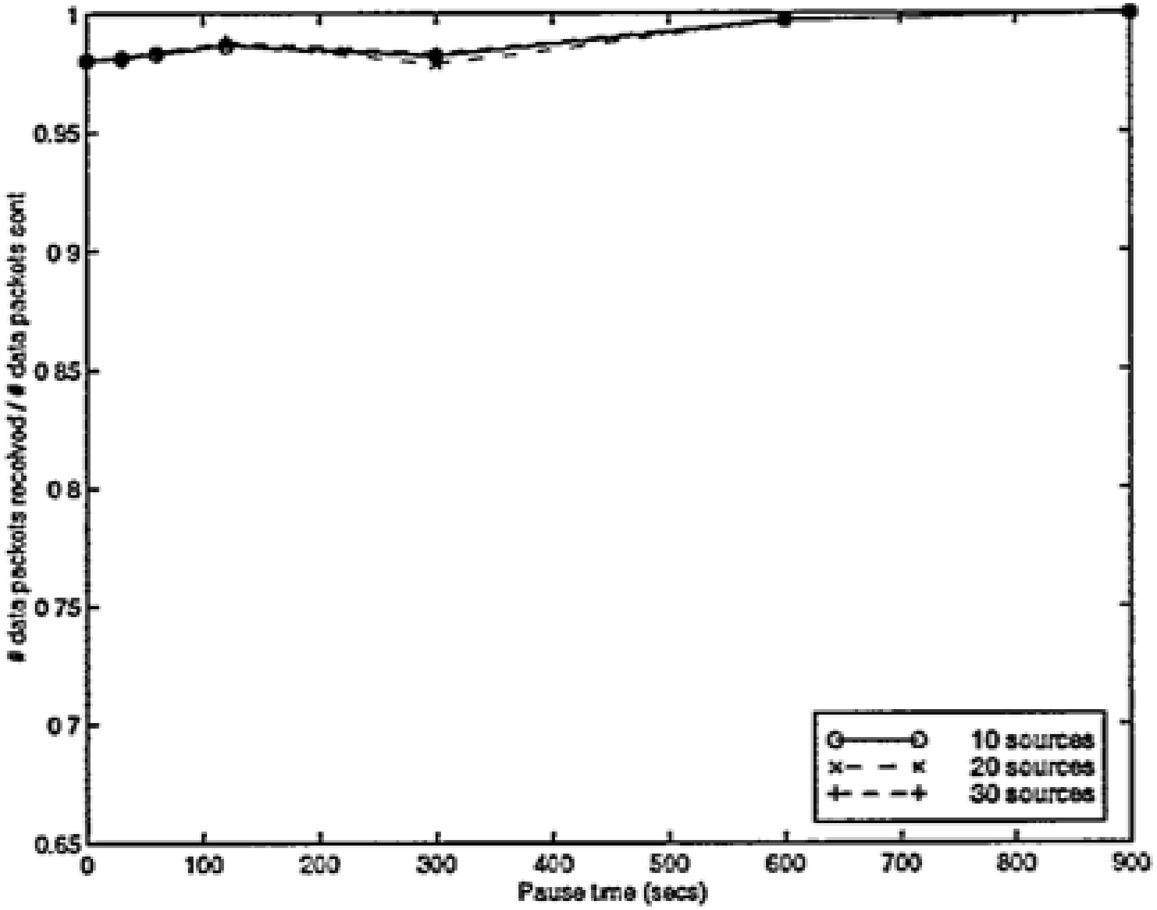
- Figure 4 shows several charts, each chart has a protocol responds to 10, 20 and 30 CBR sources based on pause time.
- Higher values are better

Packet Delivery Ratio – DSDV-SQ



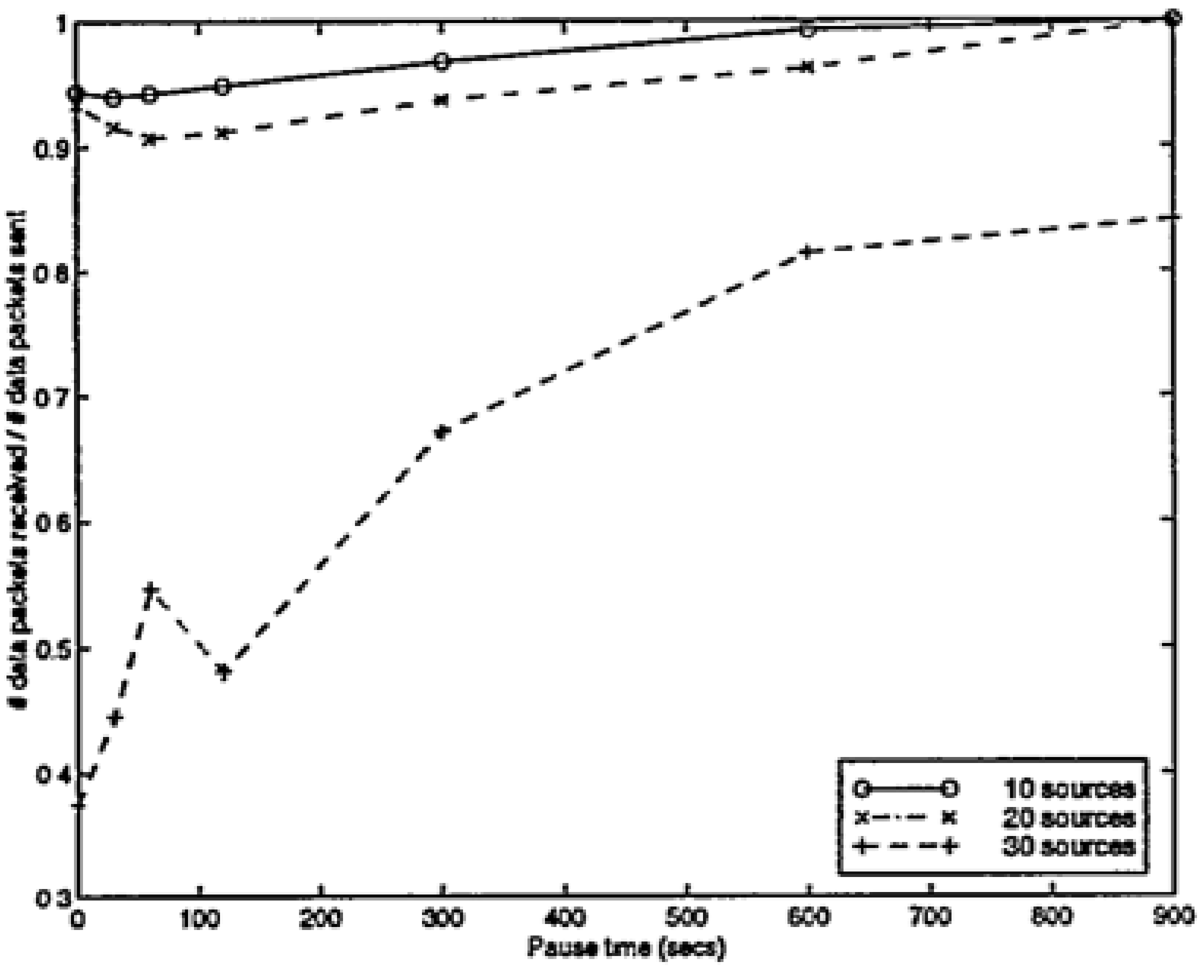
(a) DSDV-SQ

Packet Delivery Ratio - DSR



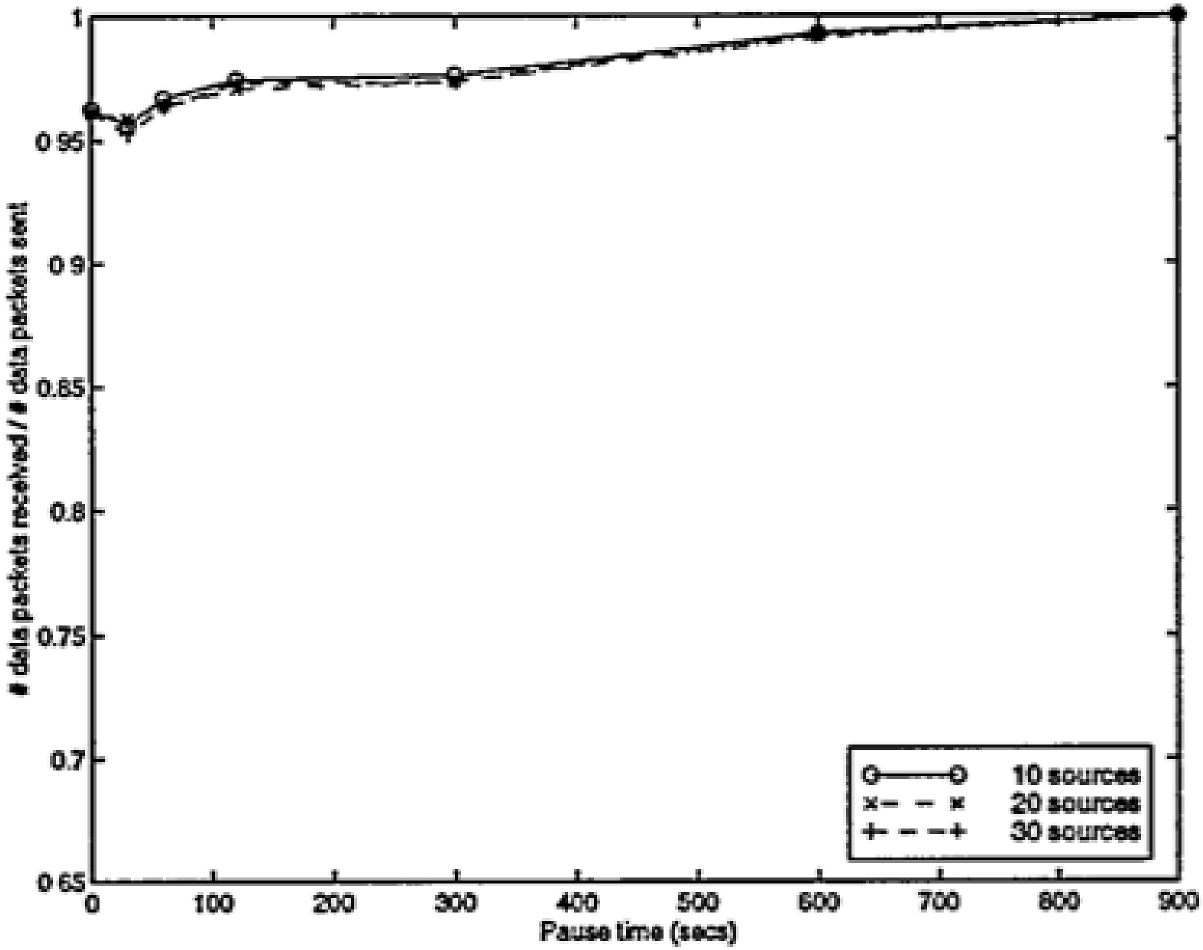
(b) DSR

Packet Deliver Ratio - TORA



(c) TORA

Packet Delivery Ratio – AODV-LL



(d) AODV-LL

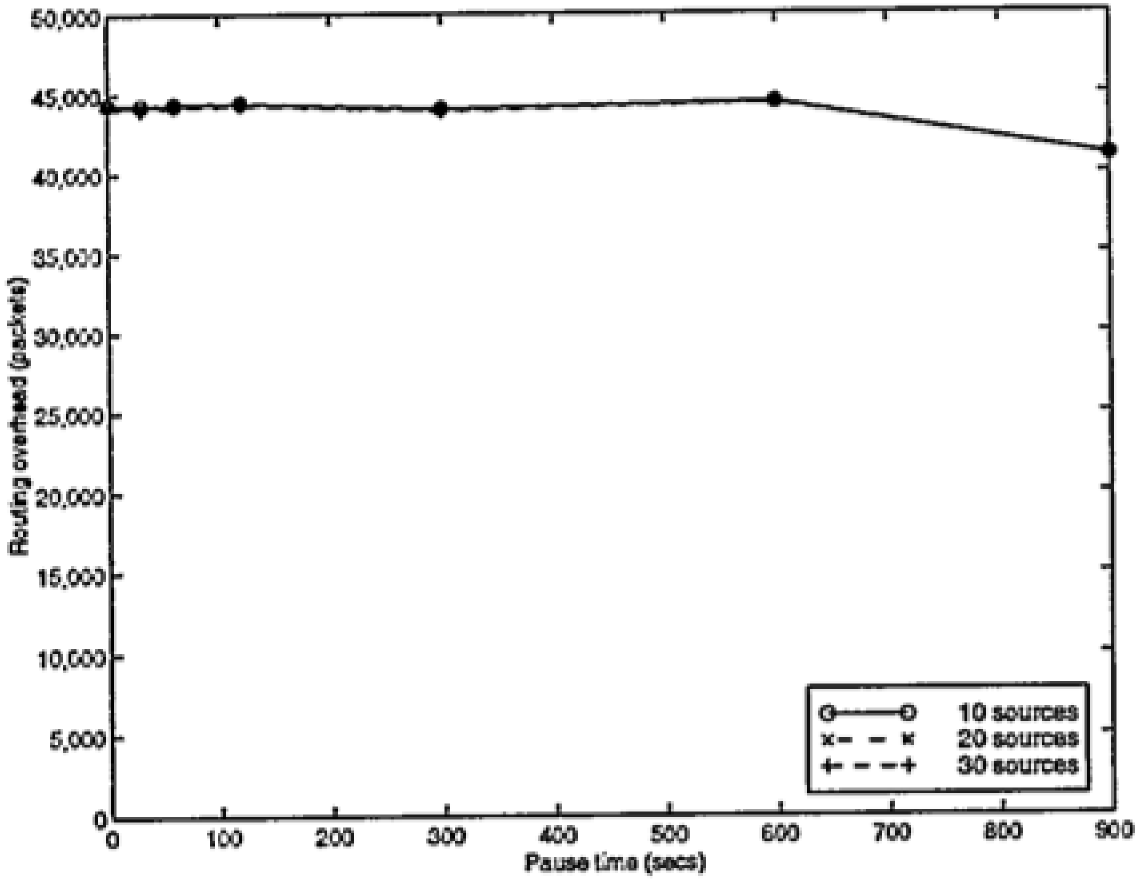
Packet Delivery Ratio

- DSR and AODV-LL have good performance at most pause times.
  - Number of sources does not affect performance
- DSDV-SQ and TORA perform poorly at high levels of mobility
- TORA only protocol that's significantly affected by a larger number of sources

Routing Overhead

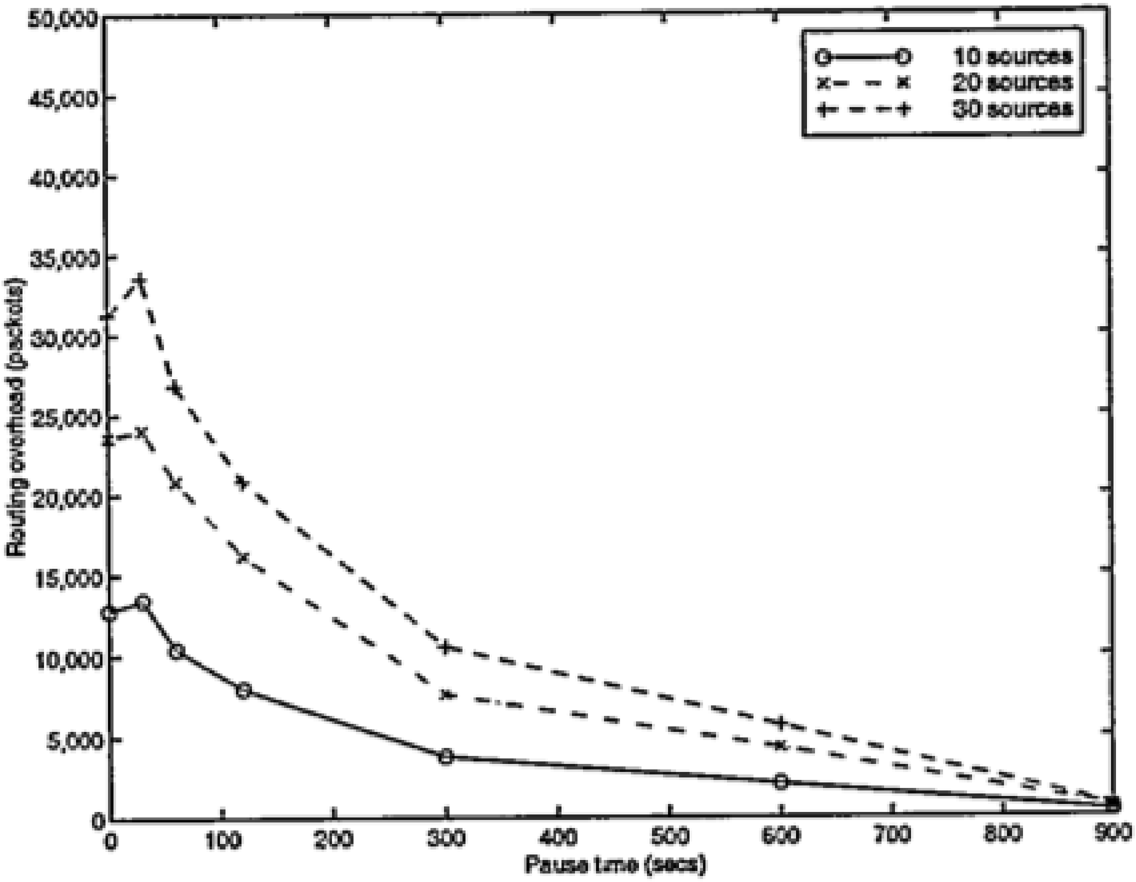
- Number of packets that each protocol is generating
- Charts in Figure 3 show a single protocol each and vary the number of sources
- Lower values are better

Routing Overhead – DSDV-SQ



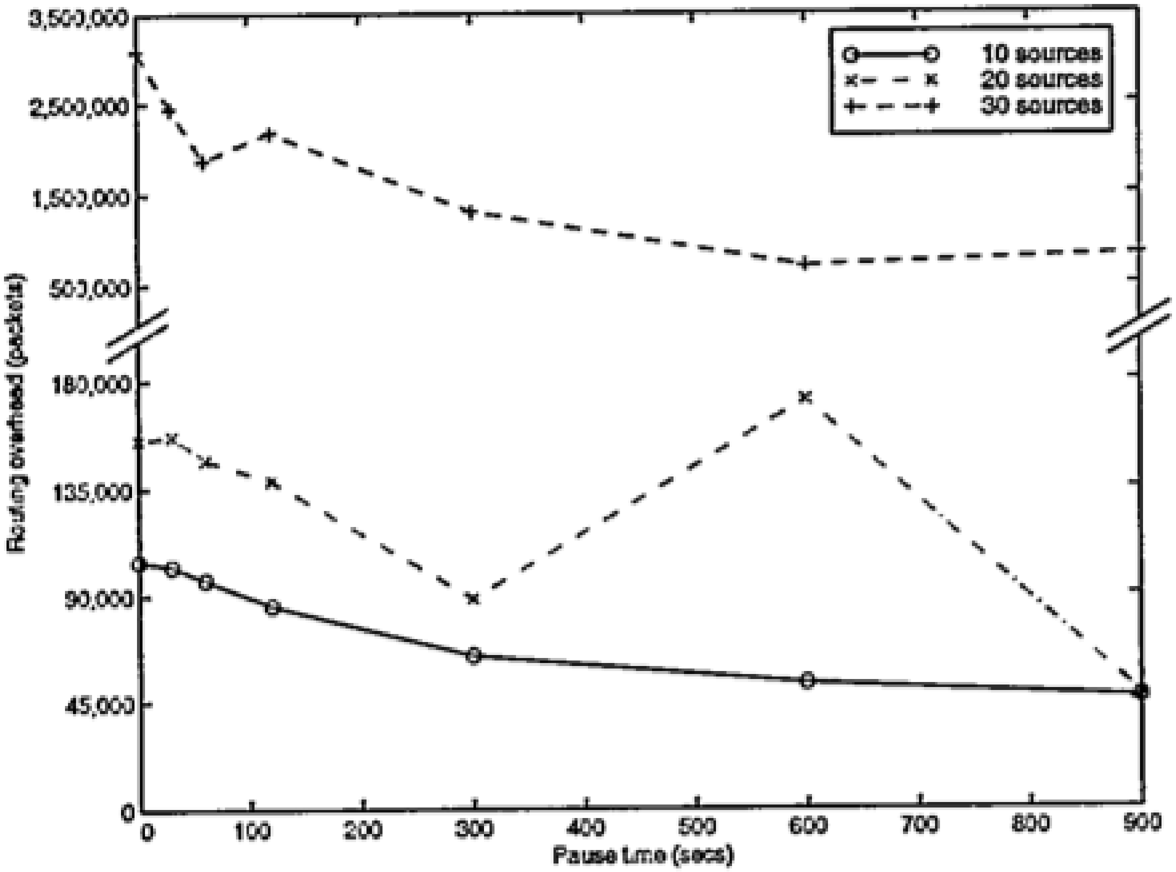
(a) DSDV-SQ

Routing Overhead - DSR



(b) DSR

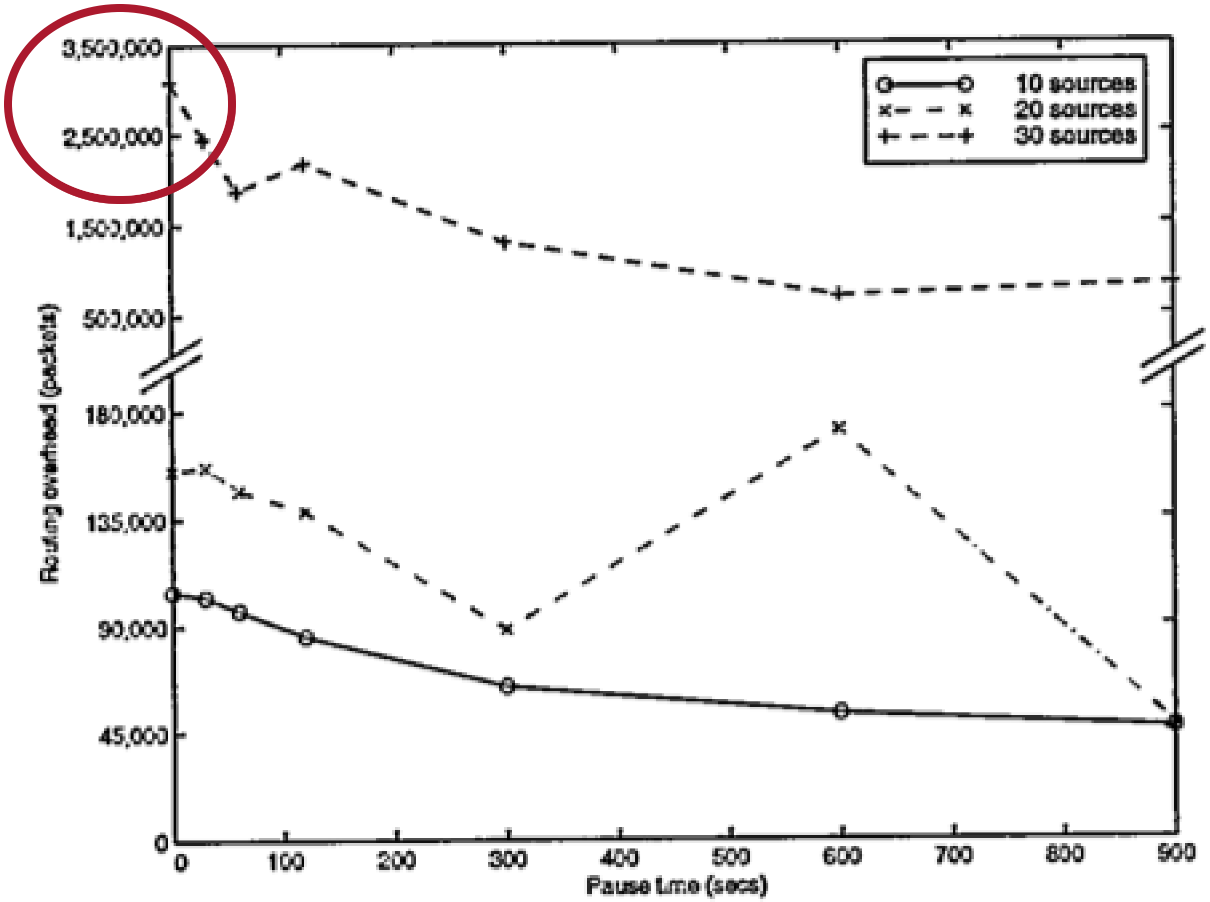
Routing Overhead - TORA



(c) TORA

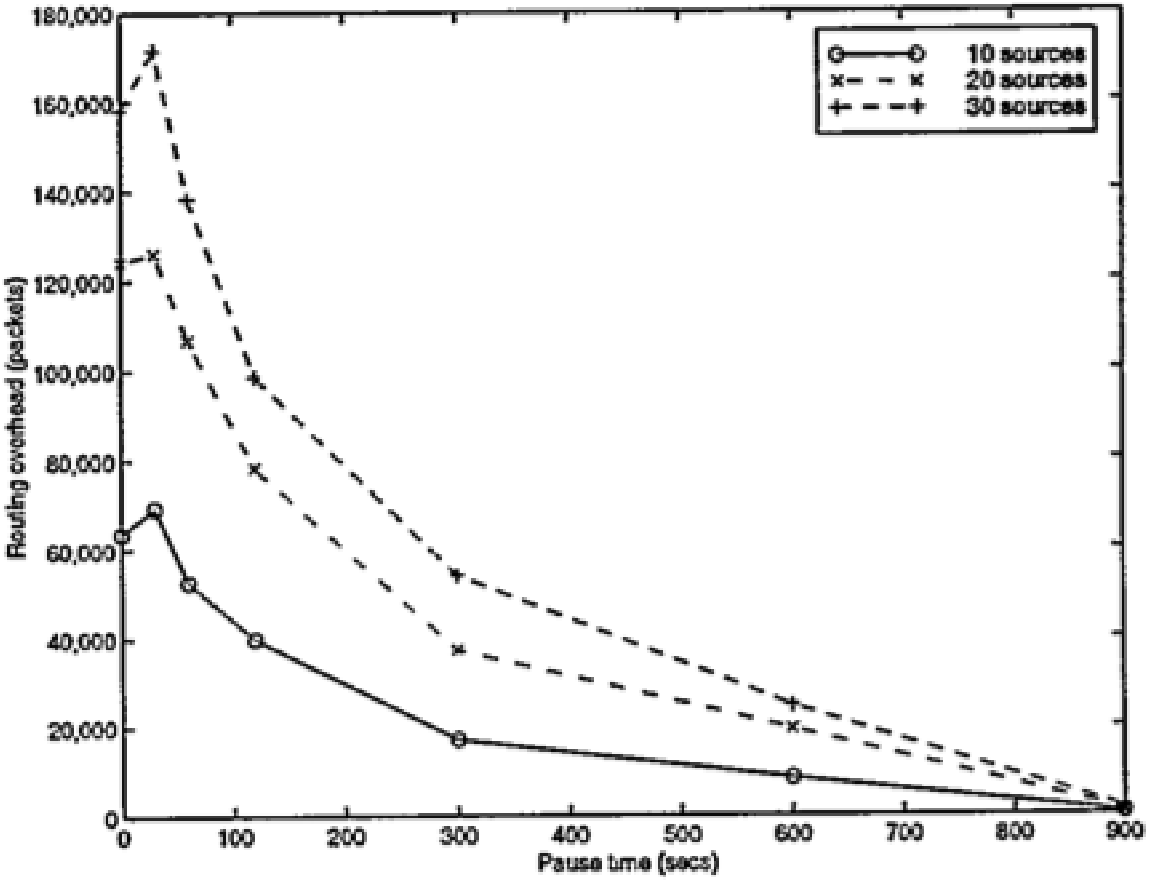
Routing Overhead - TORA

Millions!



(c) TORA

Routing Overhead – AODV-LL



(d) AODV-LL

Routing Overhead

- DSR and AODV-LL show similar curves, but AODV-LL generates 4 times as many packets!
  - Remember AODV-LL is based on DSR, but also has routing state at the nodes
- DSDV-SQ has a constant amount of overhead
  - Periodic beacons at fixed time intervals
- TORA generates many packets
  - Authors state congestion collapse from too many MAC layer collisions, which caused it to think the links were down and this generated UPDATE packets
  - Each UPDATE packet requires reliable delivery, which wasn't possible because of MAC collisions. This triggered retransmits.
  - Positive feeback loop eventually consumed the network

Path Optimality

- “The difference between the number of hops a packet took to reach its destination and the length of the shortest path that physically existed through the network when the packet was originated”
- How good are these routes?
- Only a bar at 0 is perfect, anything above 0 means extra hops

Path Optimality

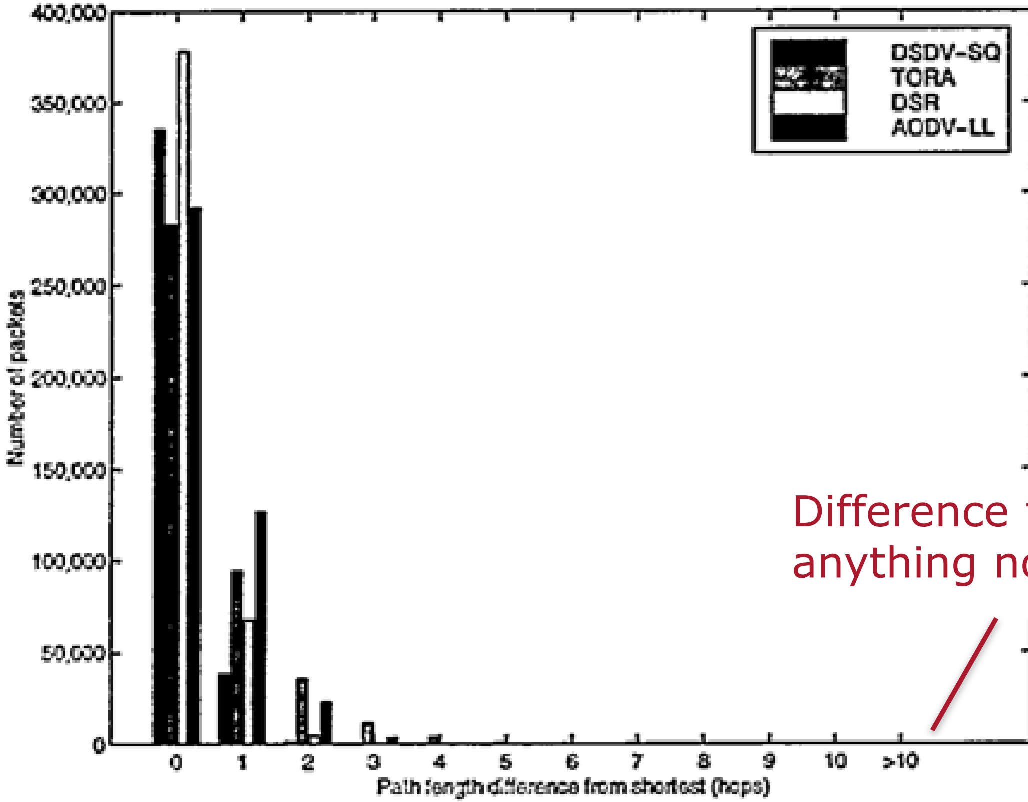


Figure 6 Difference between the number of hops each packet took to reach its destination and the optimal number of hops required. Data is for 20 sources.

Difference from shortest, anything not 0 is bad

Path Optimality

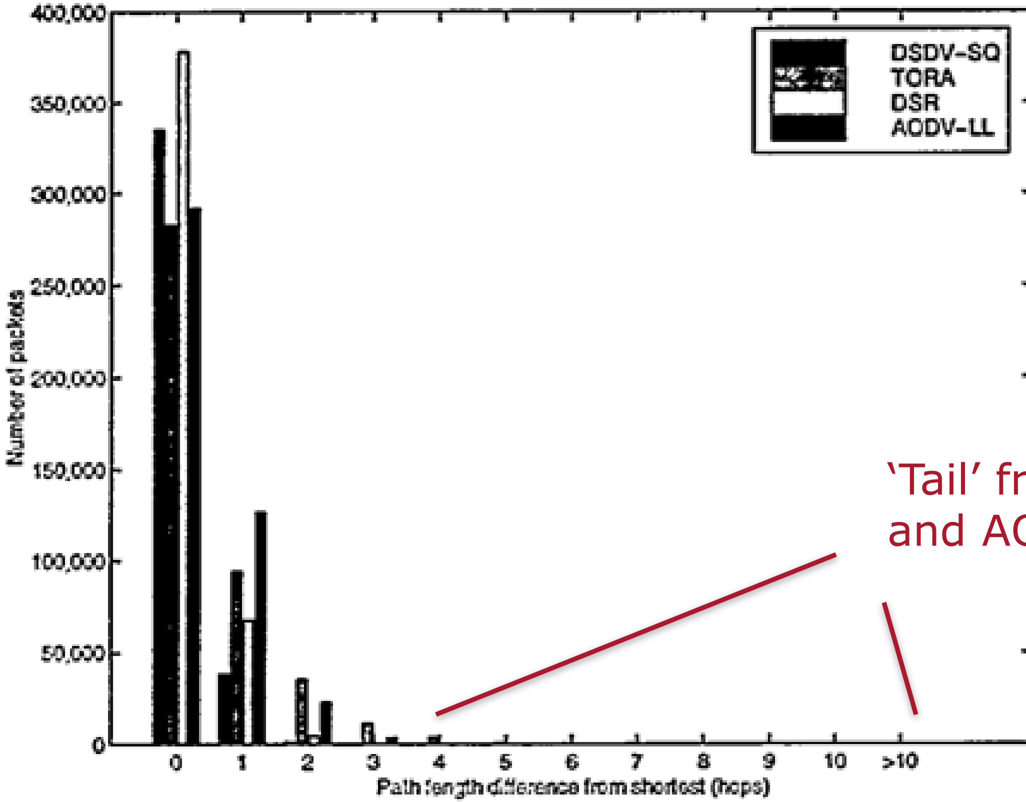


Figure 6 Difference between the number of hops each packet took to reach its destination and the optimal number of hops required. Data is for 20 sources.

Path Optimality

- DVDS-SQ and DSR do well
- TORA and AODV-LL generate some non-optimal routes
- Authors note that TORA and AODV-LL perform better when mobility is low

Movement Speed

- Re-run some experiments with 1 m/s speed instead of 20 m/s

Movement Speed – Packets Delivered

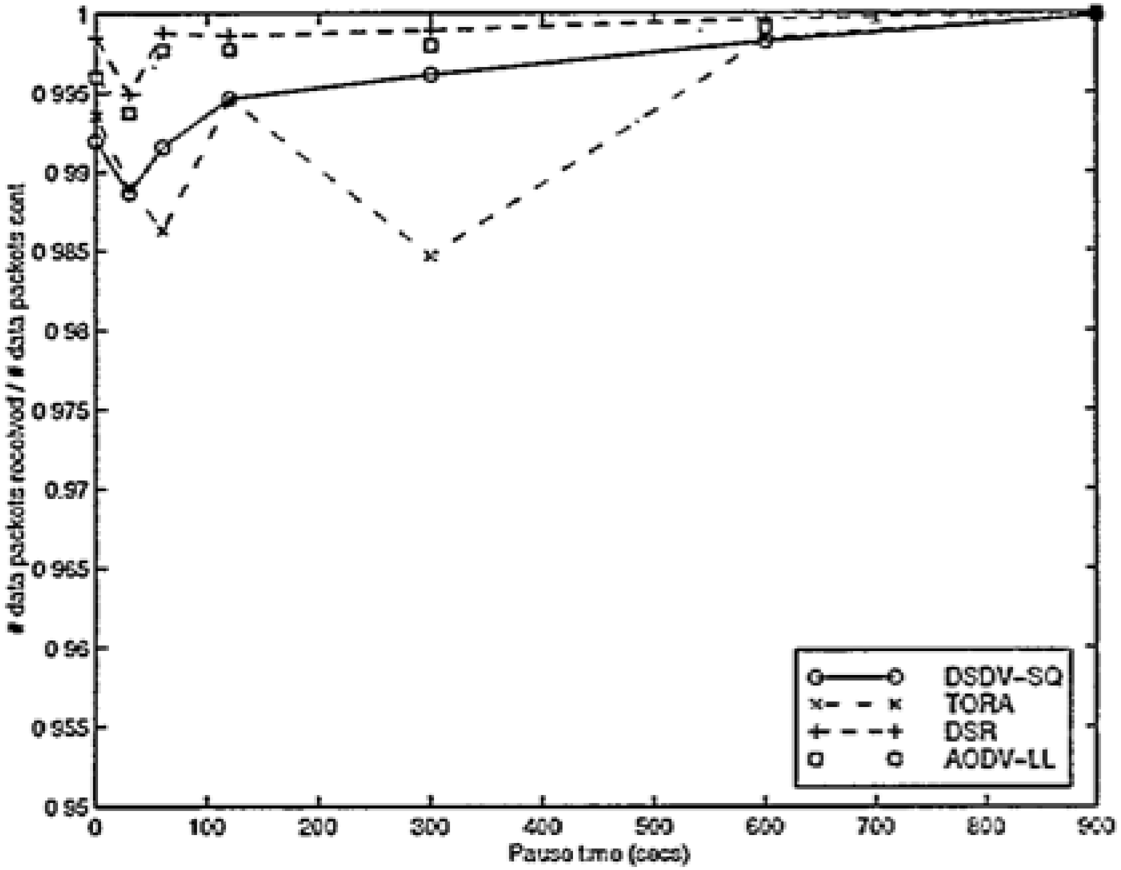


Figure 7 Comparison of the fraction of application data packets successfully delivered as a function of pause time. Speed is 1 m/s.

Movement Speed – Routing Overhead

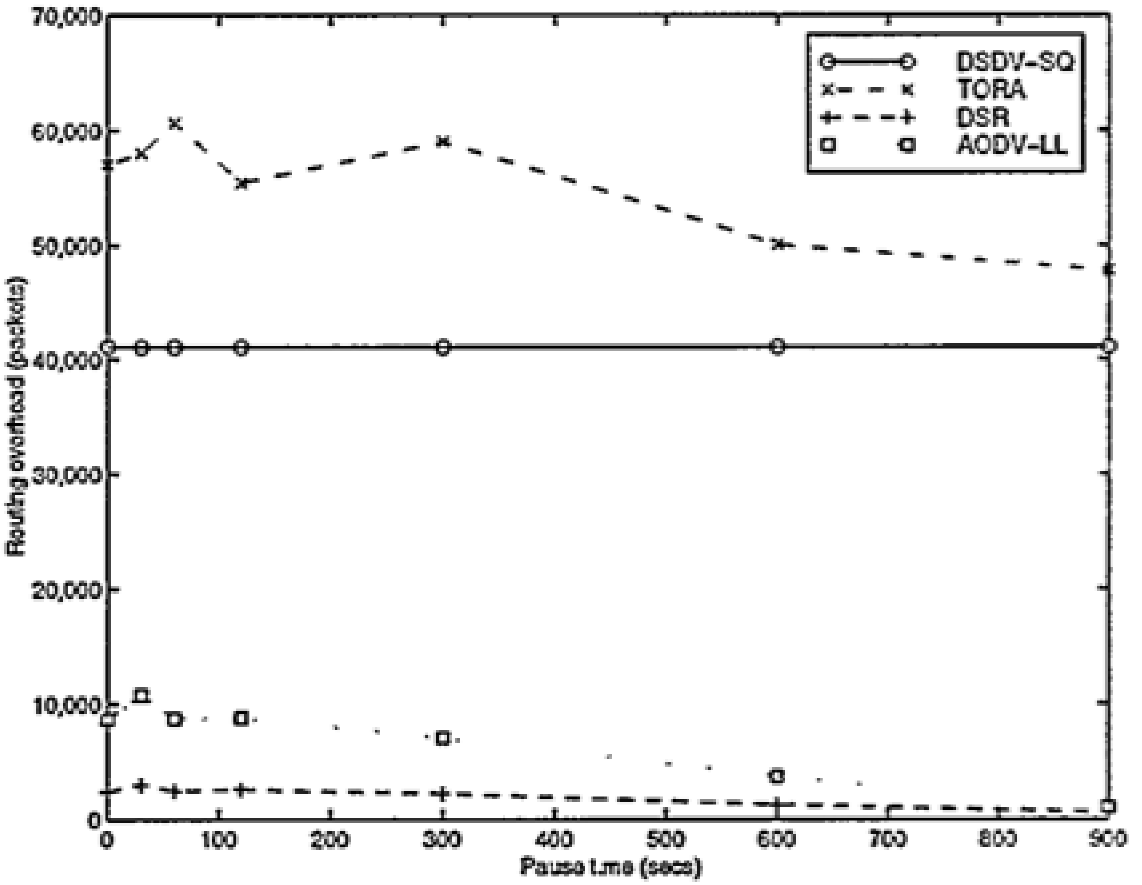


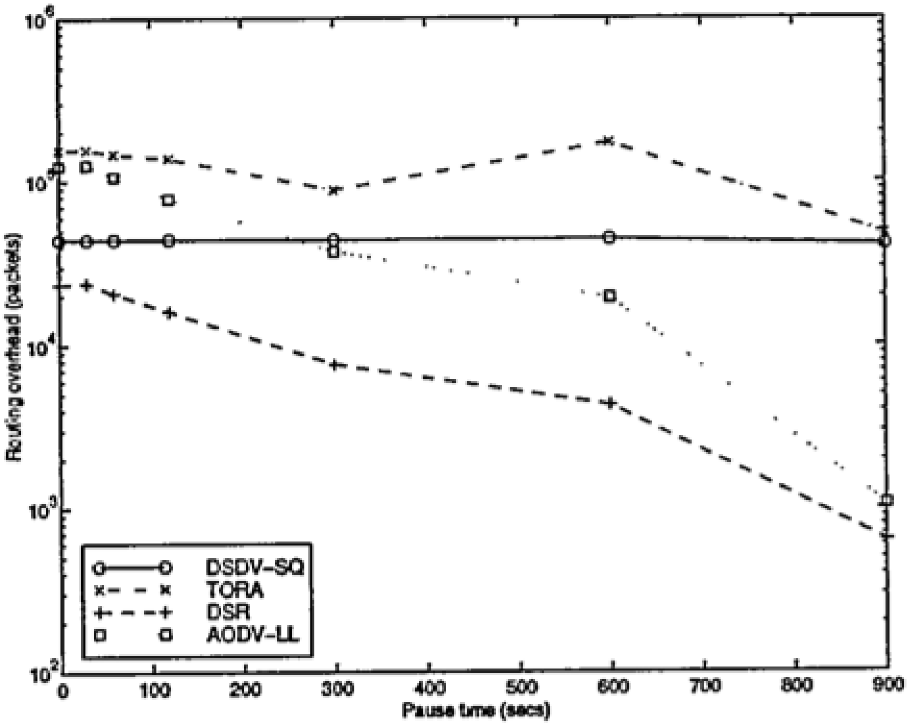
Figure 8 Comparison of the number of routing packets sent as a function of pause time. Speed is 1 m/s.

Movement Speed

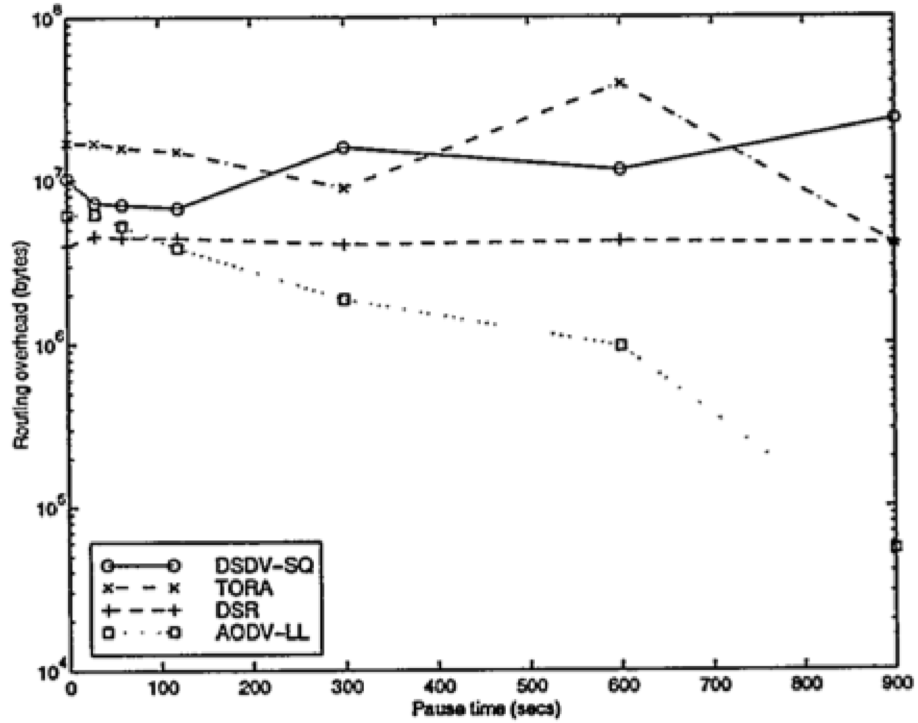
- DSR's caching is even more effective at low speeds!
  - Significantly better than AODV-LL
- DSDV-SQ still has constant overhead

Total Packet Overhead

- Includes data used to control routing in bytes
- DSR no longer as far out in front because entire route is contained in each packet



(a) Routing overhead in packets.



(b) Routing overhead in bytes.

DSDV Without SQ Addition

- Comparison of traditional DSDV without the additional update packets being sent whenever a sequence number changes
- In general routing overhead is lower, but reliability suffers except at very high mobility

DSDV Without SQ Addition

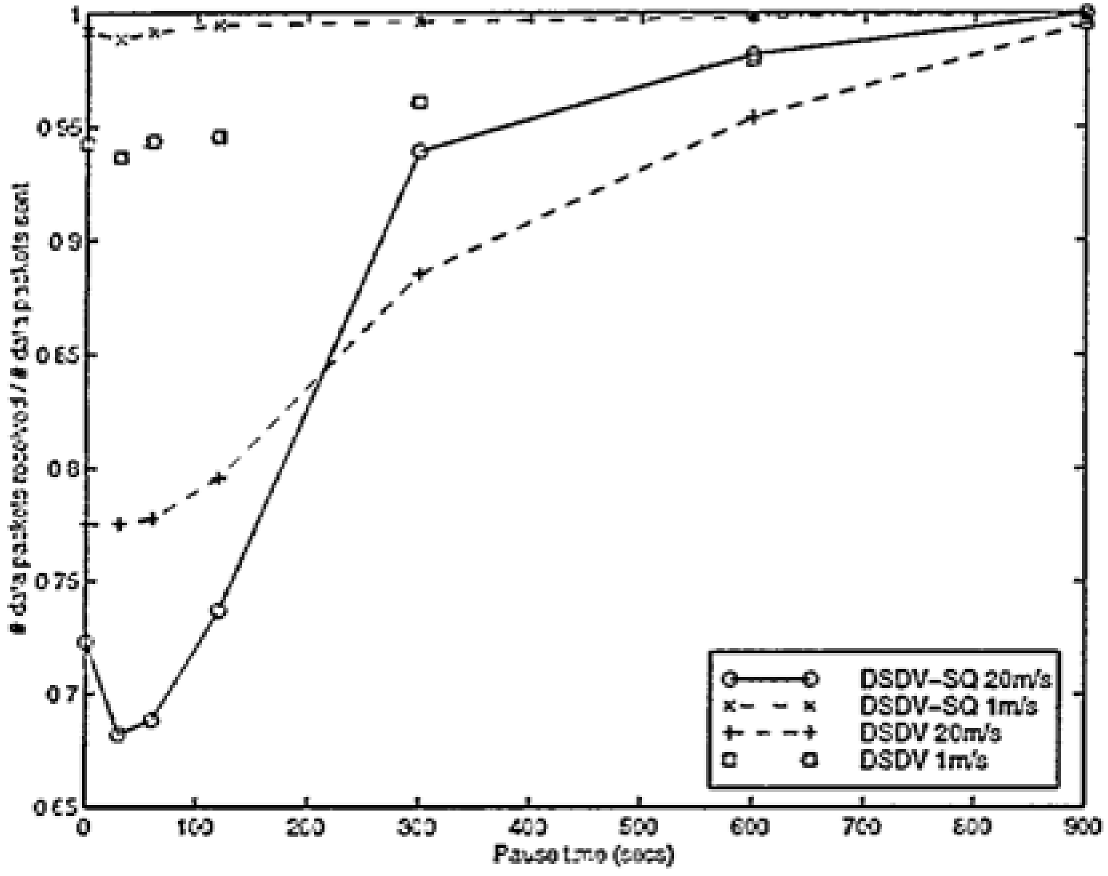


Figure 10 Fraction of originated data packets successfully delivered by DSDV-SQ and DSDV.

DSDV Without SQ Addition

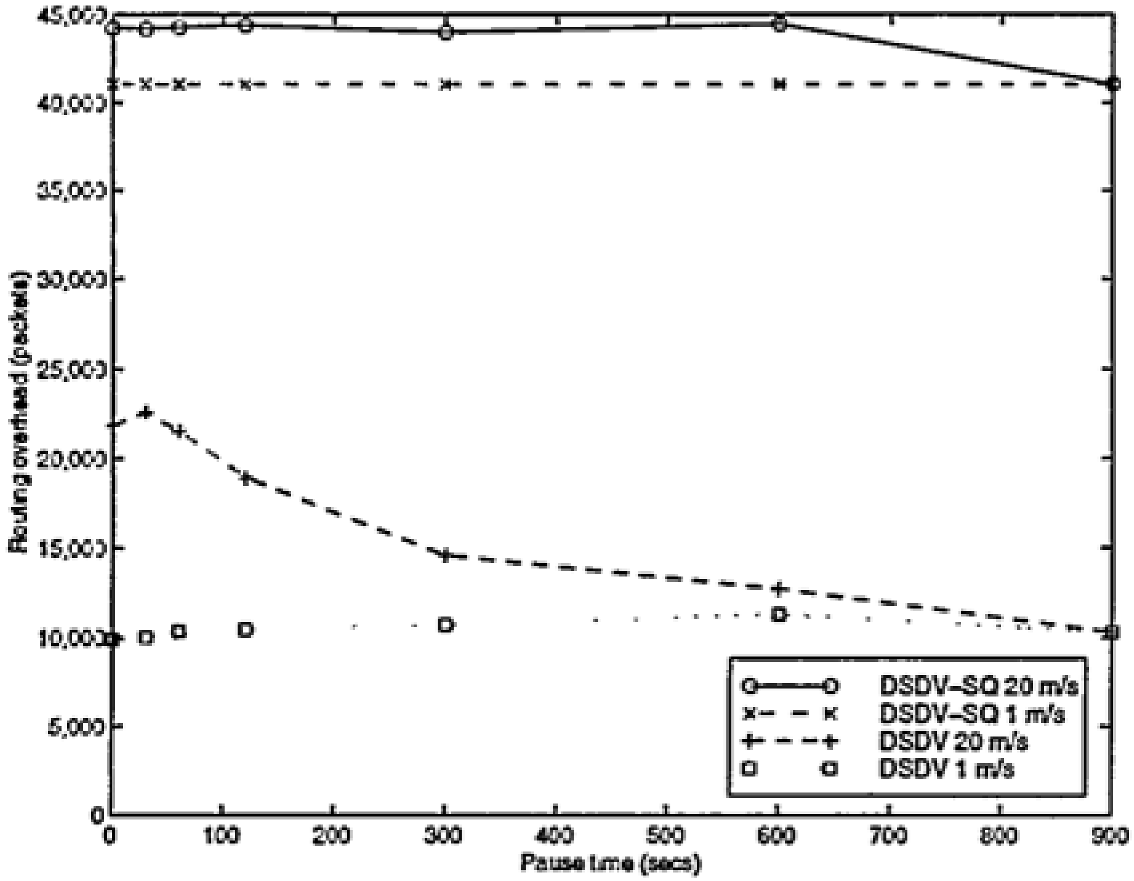


Figure 11 Routing overhead as a function of pause time for DSDV-SQ and DSDV.

Outline

- Introduction
- NS enhancements
- Protocols:
  - DSDV
  - TORA
  - DRS
  - AODV
- Evaluation
- **Conclusions**

Conclusions

- Large differences in the approaches of the protocols used and the performance of those protocols
- DSR appears to do better in most tests
- DSR is the only algorithm that does not require state at the nodes!
  - In high mobility situations routing state becomes stale and other protocols
  - DRS avoids this by rebuilding on most requests
  - DRS has promiscuous caching which helps reduce the number of packets sent

Conclusions (cont)

- Several good enhancements to NS
  - 802.11 MAC Layer
  - Physical Layer Simulator
  - Node mobility
- Some protocols (TORA) did not handle MAC collisions or lost packets well
  - Authors note previous TORA simulations were in 'ideal' environments
- Overall interesting comparison between protocols