Peer-to-Peer Systems and Applications

TECHNISCHE UNIVERSITÄT DARMSTADT

Lecture 7: Mobile P2P Systems

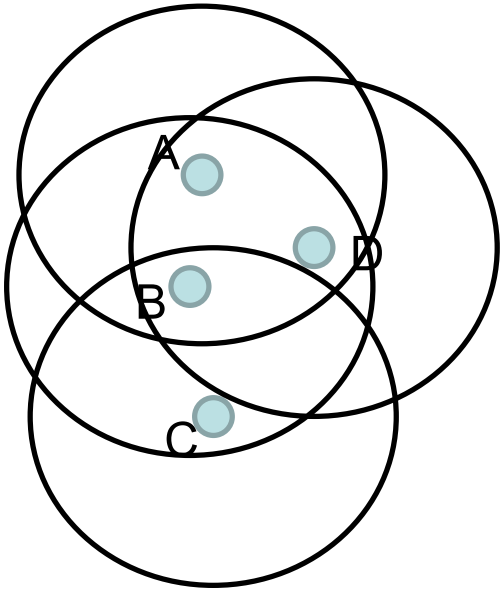
Chapter 24 and 25:

Part VIII: P2P in Mobile and Ubiquitous Environments

* Original slides for this lecture provided by David Hausheer, Matthias Wichtlhuber (TU Darmstadt, Germany), Rizal Mohd Nor (Kent State University), Thomas Zahn (Freie Universität Berlin).

0. Mobile Systems - Motivation

- Limited radio transmission range, low data rates
- Limited resources of mobile devices
  - Battery power
  - Computational power
  - Memory
  - Bandwidth
- Unpredictable terminal mobility
- High delay and jitter
- Temporary loss of connection



0. Lecture Overview

1. Traffic Routing in Multi-Hop Networks
    1. Problem
    2. Protocol Classification
    3. AODV
    4. PROPHET

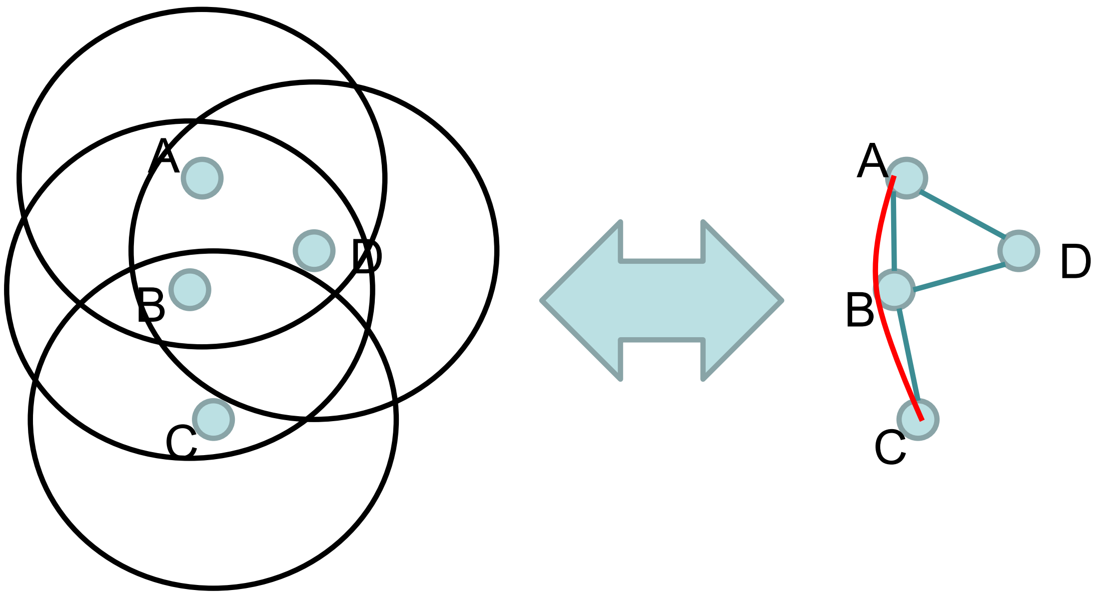
2. Why conventional DHTs Won't Work in MANETs
    1. Locality Awareness
    2. Physical Route Discovery
    3. DHT Maintenance

3. MADPastry
    1. Random Landmarking
    2. Routing Tables
    3. Unicast
    4. Simulation
    5. Conclusion

1. Traffic Routing in Multi-Hop Networks

Problem, Protocol Classification, AODV, PROPHET

1.1. Problem Definition/Motivation

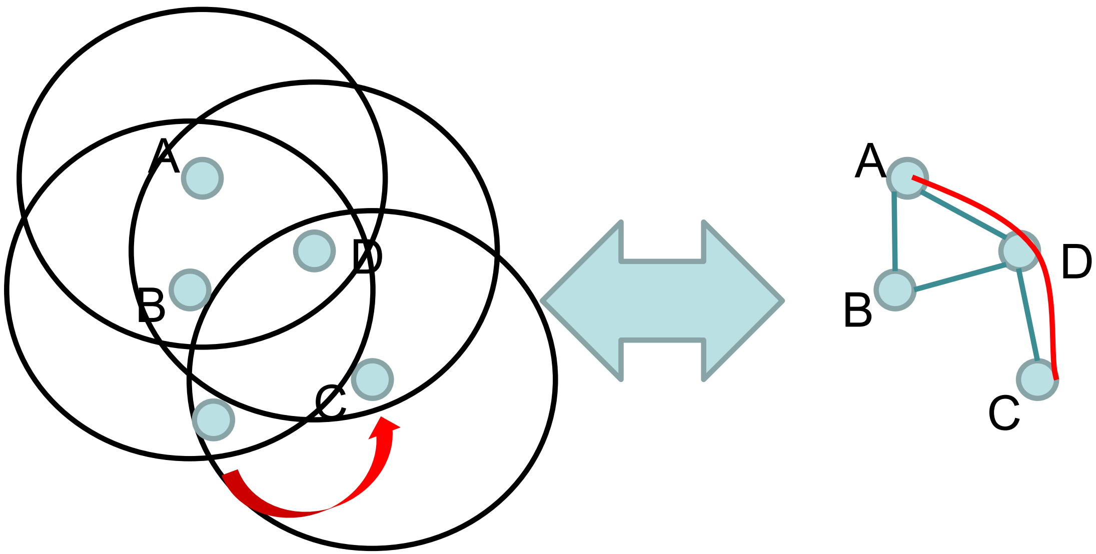


physical topology

logical tolopology

- If A wants to reach C, multi-hop traversion is needed
- Mobility causes frequent route changes

1.1. Problem Definition/Motivation



physical topology

logical tolopology

- If A wants to reach C, multi-hop traversion is needed
- Mobility causes frequent route changes

1.2. Classification of Mobile Routing Protocols

Protocol Class		Examples
Flat routing Exploit topological information	**Proactive protocols** Maintain routes actively	Open Link State Routing (OLSR)
	Reactive protocols Find route when needed	Dynamic Source Routing (DSR), Ad-hoc On Demand Distance Vector Routing (AODV)*
Hierarchical Routing Nodes communicate only with nodes of same group or hierarchy level		Hierarchical State Routing (HSR)
Geographic Routing Utilize location information for routing process		Location-Aided Routing (LAR)
Probabilistic Routing Do not assume route always exists, use store-carry-forward principle		Probabilistic Routing Protocol using History of Encounters and Transitivity (PROPHET)*

* Will be discussed in detail in the course of this lecture

1.3. Ad-hoc On Demand Distance Vector Routing (AODV)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Developed by Perkins and Royer (1999)
  - Flat routing scheme, all nodes have equal role
  - Reactive protocol, routes are only calculated, when a node wants to send a package
    - Ensures small node state
  - Baseline protocol for all scientific publications in the area of MANET routing
    - Any author has to prove superior performance compared to AODV

[Charles E. Perkins and Elizabeth M. Royer. "Ad hoc On-Demand Distance Vector Routing." Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.]

1.3. AODV Messages

- RREQ (Route REQuest) → Requesting a route to a certain destination

Source Address (IP)	Request ID	Destination address (IP)	Source Sequence #	Destination Sequence #	Hop Count

- RREP (Route REPly) → Reply to a RREQ

Source Address (IP)	Destination Address (IP)	Destination Sequence #	Hop Count	Lifetime

- RERR (Route ERRor) → Signalling of dead routes

Source Address (IP)	Destination Address (IP)	Destination Sequence #	Hop Count	Lifetime

1.3. Node State

- Local counters:
  - Request ID → Incremented on every Route Request
  - Sequence Number → Incremented on every Route Request <u>and</u> Route Reply

Req.-ID: 10		Loc. Seq. #: 11	
Dest.	**Link**	**Hops**	**Seq. #**
IP_E	B	1	12
IP_G	D	1	15

- Routing table:
  - Keyed by destination
  - Link describes next hop
  - Hop number indicates distance of destination
  - Sequence number indicates freshness of route
    - Higher sequence number → fresher route to destination

1.3. Route Requests

- Unknown route→send Route REQuest
  - Example: A queries route to E

Source Address (IP)	Request ID	Destination address (IP)	Source Sequence #	Destination Sequence #	Hop Count
IP_A	Loc. Request ID	IP_E	Loc. Seq. $\#_A$	Seq. $\#_E$ (routing table)	0

- Request ID: Counter incremented each time A issues a RREQ packet
  - (Source Address, Request ID) is used to identify RREQ
- Source Sequence #: sequence number to be used in the route entry pointing towards A (= A's current sequence number)
- Destination Sequence #: latest sequence number received in the past by A for any route towards the destination (if never seen, 0)

1.3. Route Requests

TECHNISCHE UNIVERSITÄT DARMSTADT

- Route REQuest is flooded in the network

- Algorithm for processing a recevied RREQ
  1. If (Source Address, Request ID) is in local history, stop processing, else add tupel to local history.
  2. Look up Destination Address in routing table.
     If (local route's Sequence Number >= Sequence Number of RREQ), return route using Route REPly, else increment Hop Count and create reverse routing entry pointing to the node the packet was received from.

1.3. Route Replies

- RREQ is answered by Route REPly
  - Example: E gets RREQ (A→E) and answers

Source Address (IP)	Destination Address (IP)	Destination Sequence #	Hop Count	Lifetime
IP_A	IP_E	Loc. Seq. #	# hops from RREQ	Some constant

- Destination sequence number is the local sequence number <u>or</u> sequence number from routing table, if intermediate nodes answers

1.3. Route Replies

- Any node processes RREP packages and extracts useful information

- Node I enters information from RREP to local routing table, if one of the following conditions is met:
  1. I does not know a route to the destination
  2. The RREP contains a fresher route than the currently known route
  3. The freshness of the route is equal, but the route is shorter

1.3. Route Discovery

- Example: A requests route to E
  - A's routing entry to E is stale

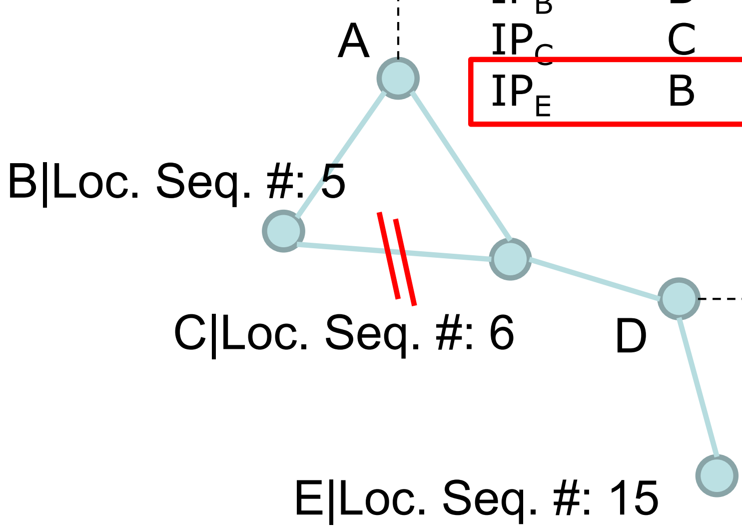


Table at A:

Req.-ID: 10		Loc. Seq. #: 11		
Dest.	**Link**	**Hops**	**Seq. #**	
IP_B	B	1	5	
IP_C	C	1	6	
IP_E	B	4	14	Stale entry

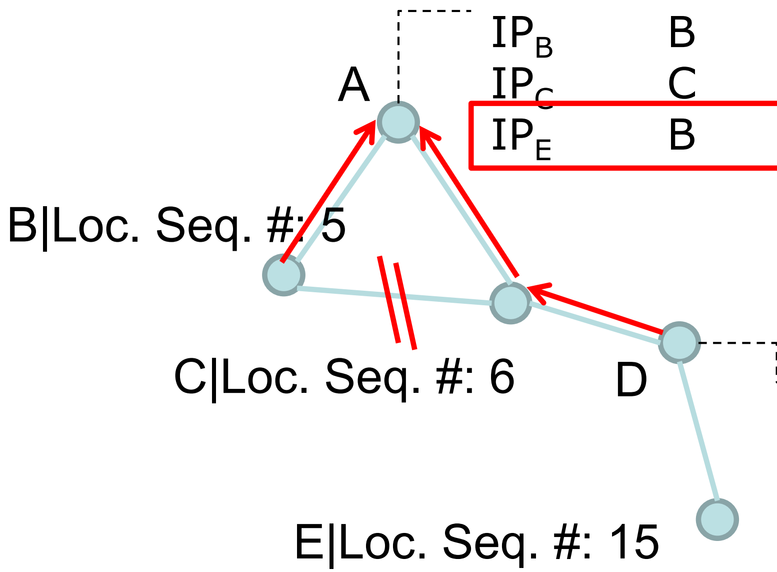
Table at D:

Req.-ID: 10		Loc. Seq. #: 11	
Dest.	**Link**	**Hops**	**Seq. #**
IP_E	E	1	15
IP_C	C	1	6

1.3. Route Discovery

- Example: A requests route to E
  - Flood route request
  - Nodes that do not have a route create reverse route entry

Req.-ID: 10		Loc. Seq. #: 11		
Dest.	Link	Hops	Seq. #	
IP_B	B	1	5	
IP_C	C	1	6	
IP_E	B	4	14	Stale entry

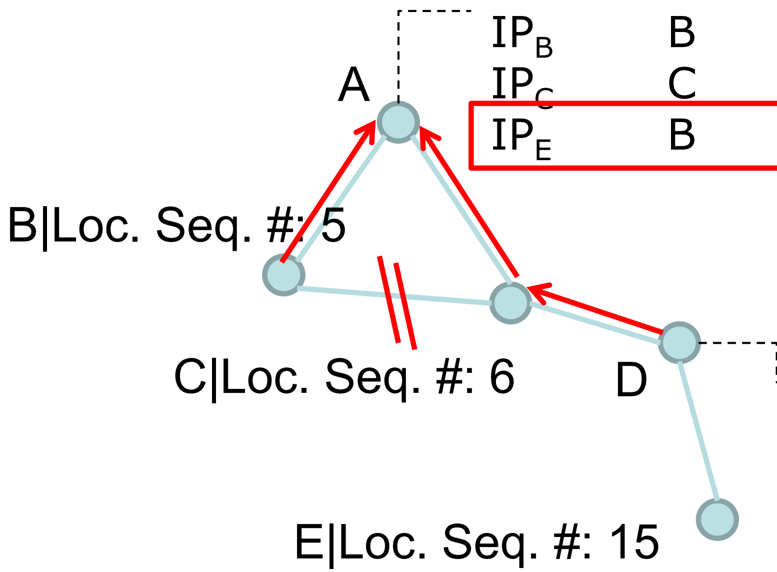


Req.-ID: 10		Loc. Seq. #: 11	
Dest.	Link	Hops	Seq. #
IP_E	E	1	15
IP_C	C	1	6

1.3. Route Discovery

- Example: A requests route to E
  - D has route to E with a higher Seq. # than A (15 > 14)
  - D sends back Route REPly and increments it`s Loc. Seq. #

Req.-ID: 10		Loc. Seq. #: 11		
Dest.	**Link**	**Hops**	**Seq. #**	
IP_B	B	1	5	
IP_C	C	1	6	
IP_E	B	4	14	Stale entry

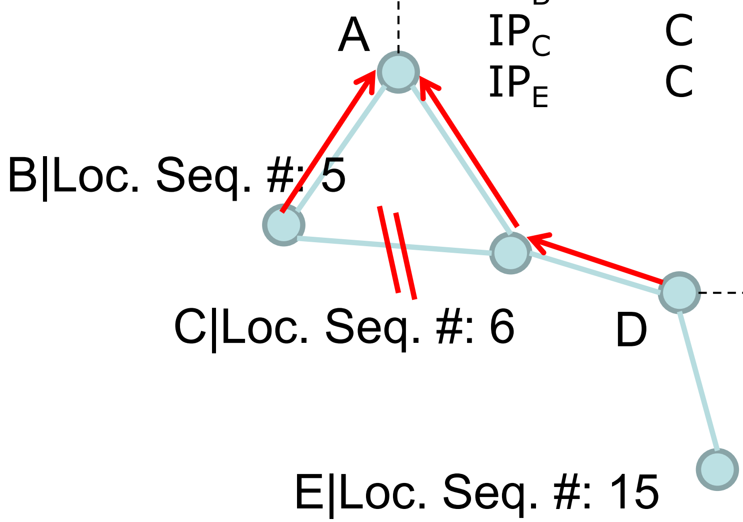


Req.-ID: 10		Loc. Seq. #: 12	
Dest.	**Link**	**Hops**	**Seq. #**
IP_E	E	1	15
IP_C	C	1	6

1.3. Route Discovery

- Example: A requests route to E
  - A and all intermediate nodes (C) update routing table

Req.-ID: 10		Loc. Seq. #: 11	
Dest.	**Link**	**Hops**	**Seq. #**
IP_B	B	1	5
IP_C	C	1	6
IP_E	C	3	15



Req.-ID: 10		Loc. Seq. #: 12	
Dest.	**Link**	**Hops**	**Seq. #**
IP_E	E	1	15
IP_C	C	1	6

1.3. HELLO and RERR Messages

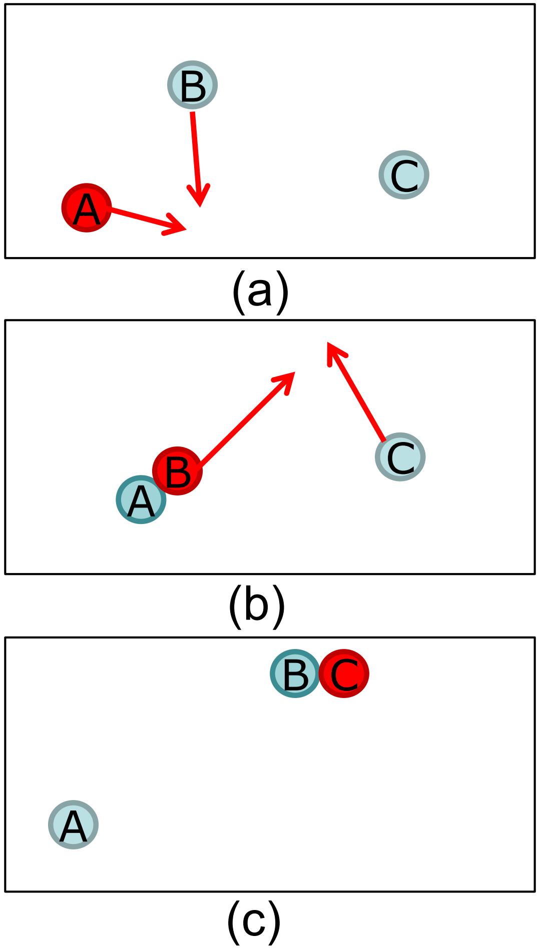
TECHNISCHE UNIVERSITÄT DARMSTADT

- HELLO messages are broadcasted periodically to detect neighbours
  - A HELLO message is a RREP packet with a Lifetime of 1 hop
  - Receiving nodes update their routing tables

- RERR messages are sent by intermediate nodes when route is lost
  - E.g. intermediate node cannot detect next hop, because neighbor has left network
  - RERR messages travel back to sender, which can issue a new RREQ

1.4. Probabilistic Routing Protocol using History of Encounters and Transitivity (PROPHET)

- Probabilistic Routing
  - Routing in intermittently connected networks
    - Existence of a route (A→B) cannot be guaranteed
    - **Examples:** satellite communication, military applications
  - Routing utilizes store-carry-forward principle
    - Nodes store packages and forward them as they meet new nodes → exploitation of mobility



[A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," SIGMOBILE Mob. Comput. Commun. Rev., vol. 7, no. 3, pp. 19–20, 2003.]

1.4. Delivery Predictability

- Which node should be selected as the next hop?
  - $P_{(a,b)} \in [0,1]$ defines the probability of a message to be delivered at every node a for each destination node b
  - Whenever a node is encountered, delivery predictability is updated: $P_{(a,b)} = P_{(a,b)_{old}} + \left(1 - P_{(a,b)_{old}}\right) \times P_{init}$, with $P_{init} \in [0,1]$ as initialization factor
  - Delivery predictability ages as nodes are not encountered again: $P_{(a,b)} = P_{(a,b)_{old}} \times \gamma^k$, with $\gamma \in [0,1)$ as aging factor and k representing the elapsed time since the last contact
  - Transitivity is considered with a scaling factor $\beta \in [0,1]$: $P_{(a,c)} = P_{(a,c)_{old}} + \left(1 - P_{(a,c)_{old}}\right) \times P_{(a,b)} \times P_{(b,c)} \times \beta$. Therefore, routing vectors are exchanged.

1.4. PROPHET Forwarding Strategies

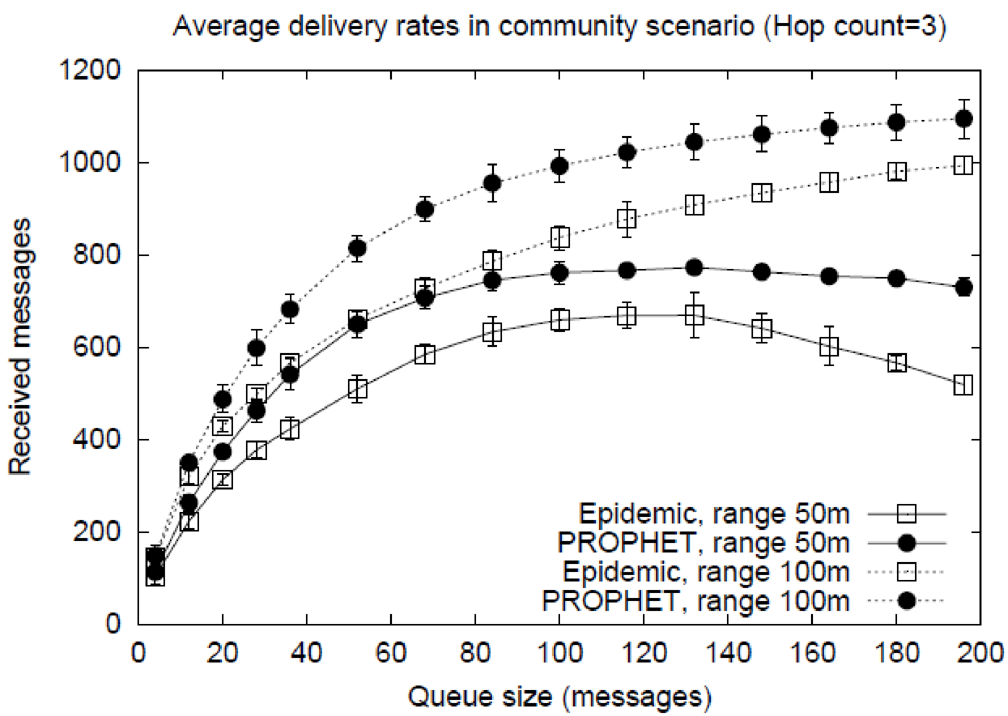
TECHNISCHE UNIVERSITÄT DARMSTADT

- In some cases it might be sensible to select a fixed threshold and only give a message to nodes that have a delivery predictability over that threshold for the destination of the message.

- Design alternatives
  - Distributing a message to a large number of nodes
    - This will increase the probability of delivering a message to its destination, but in return, more system resources will be wasted
  - Giving a message to only a few nodes (maybe even just a single node)
    - This will use little system resources, but the probability of delivering a message is probably lower, and the incurred delay high.

- Selected strategy is a rather simple forwarding strategy
  - When two nodes meet, a message is transferred to the other node if the delivery predictability of the destination of the message is higher at the other node.

1.4. Performance

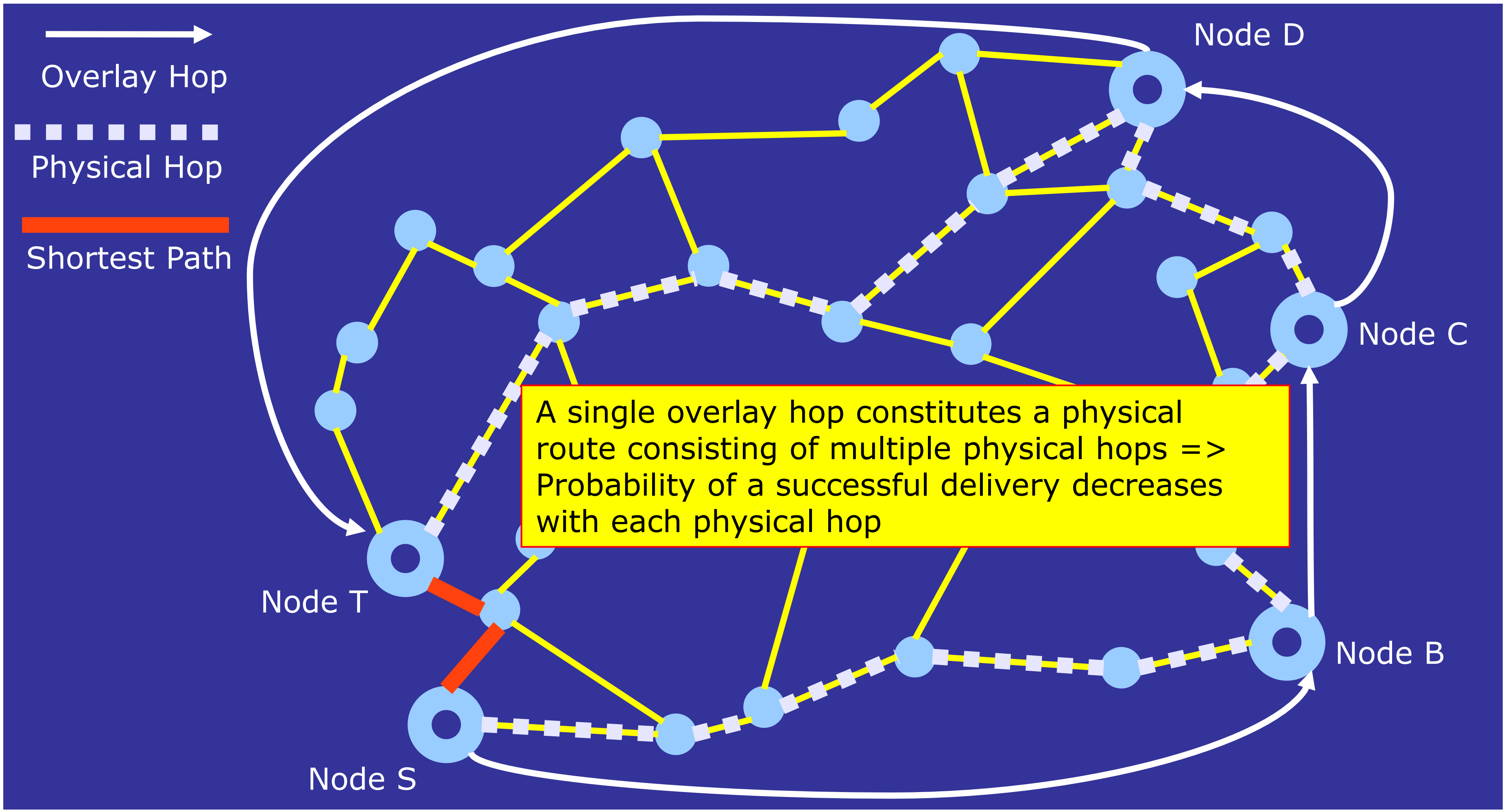


- When compared to spreading packets to every node (Epidemic), PROPHET performs superior.

2. Why conventional DHTs Won't Work in MANETs

Locality Awareness, Physical Route Discovery, DHT Maintenance

2.1. Locality Awareness

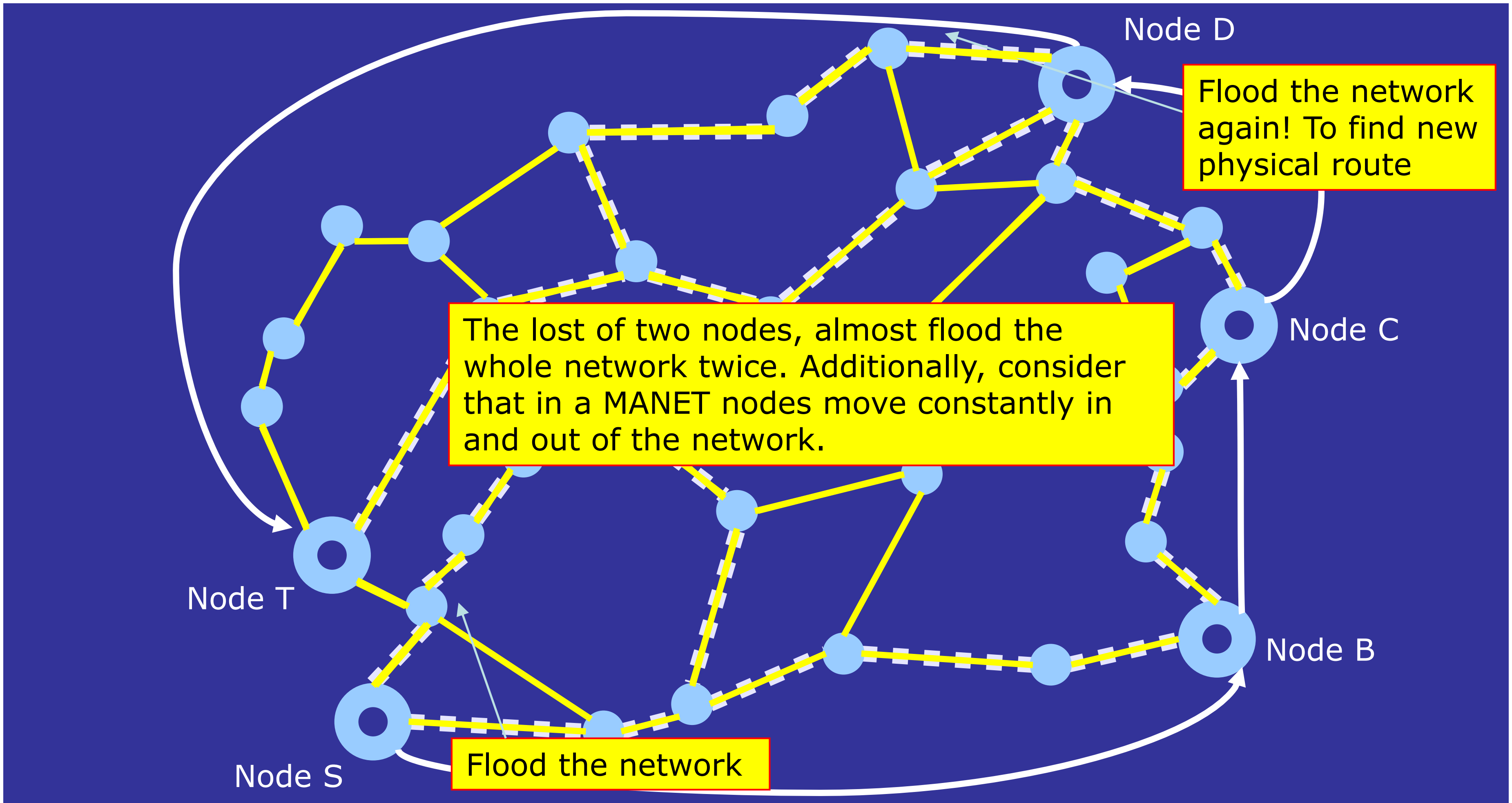


A single overlay hop constitutes a physical route consisting of multiple physical hops => Probability of a successful delivery decreases with each physical hop

2.2. Physical Route Discovery

- Unlike the wired Internet with its comparatively stable infrastructure, the topology of a MANETs changes constantly.
- Routing protocols for MANETs are predominantly concerned with rediscovering routes between nodes.
- Hence, the expensive physical route discoveries in MANETs can quickly cancel out the efficiency of DHT-based overlay routing.

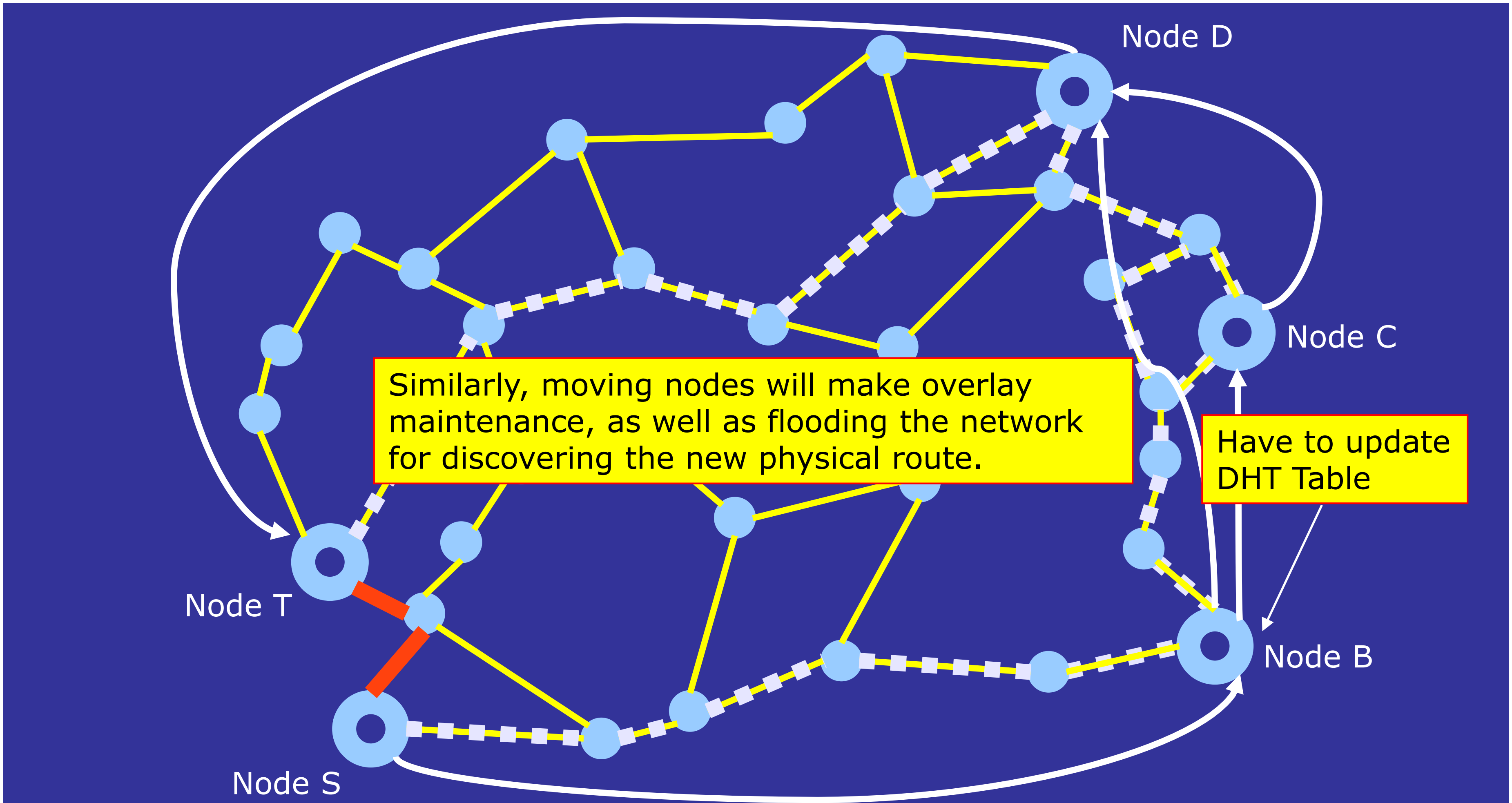
2.2. Physical Route Discovery



2.3. DHT Maintenance

- DHTs impose certain requirements that their routing table entries have to match.
- Depending on the structure and size of their routing tables, this overlay maintenance can incur significant amount of traffic.
- Extra overlay maintenance traffic can easily add a sizeable portion to the overall traffic, which will further increase the probabilities of collision and use up precious bandwidth.

2.3. DHT Maintenance



3. MADPastry

Random Landmarking, Routing Tables, Unicast, Simulation, Conclusion

3.0. MADPastry

- MADPastry combines ad hoc routing (AODV) and P2P overlay routing (Pastry) at the network layer
  - provides **indirect – i.e. key-based** – routing in MANETs
  - explicitly considers locality in the construction of its overlay

- DHT-based distributed network applications from the Internet can be ported to MANETs
  - e.g. name services, messaging systems, event-notification, storage systems

- Thomas Zahn, Jochen Schiller: MADPastry: A DHT Substrate for Practicably Sized MANETs: Proc. of ASWN, 2005

3.1. MADPastry – Random Landmarking (RLM)

- No fixed landmark nodes, landmark keys instead:
  - 0800..00, 1800..00, ..., F800..00
- Node currently closest to a landmark key becomes temporary landmark node
- Temporary landmark node sends periodic beacons to form physical clusters of common overlay ID prefixes
- Nodes overhear these beacon messages and periodically determine the physically closest temporary landmark node
  - Assumes same overlay ID prefix
  - If need be, a node assigns itself a new overlay ID sharing the same prefix with the new closest temporary landmark node
  - Physically close nodes are also likely to be close in the overlay

3.1. MADPastry – Random Landmarking (RLM)

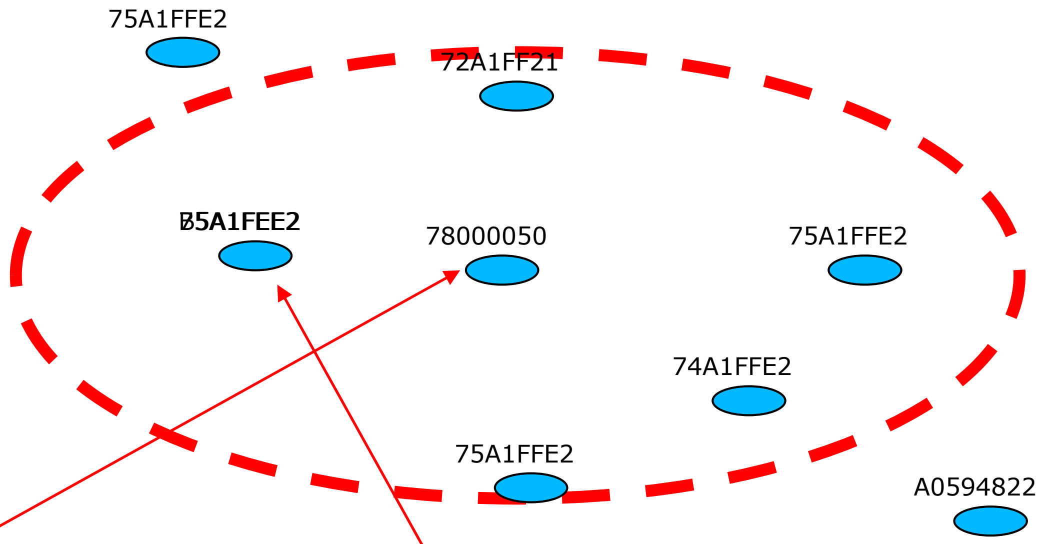
1) Initially, all nodes assigns itself overlay IDs

2) Listen for Beacons, to find out if they are close to any Landmark keys, if not check to see if they are responsible for being temporary landmark node

3) Node currently closest to a landmark key, becomes temporary landmark Node (RLM)

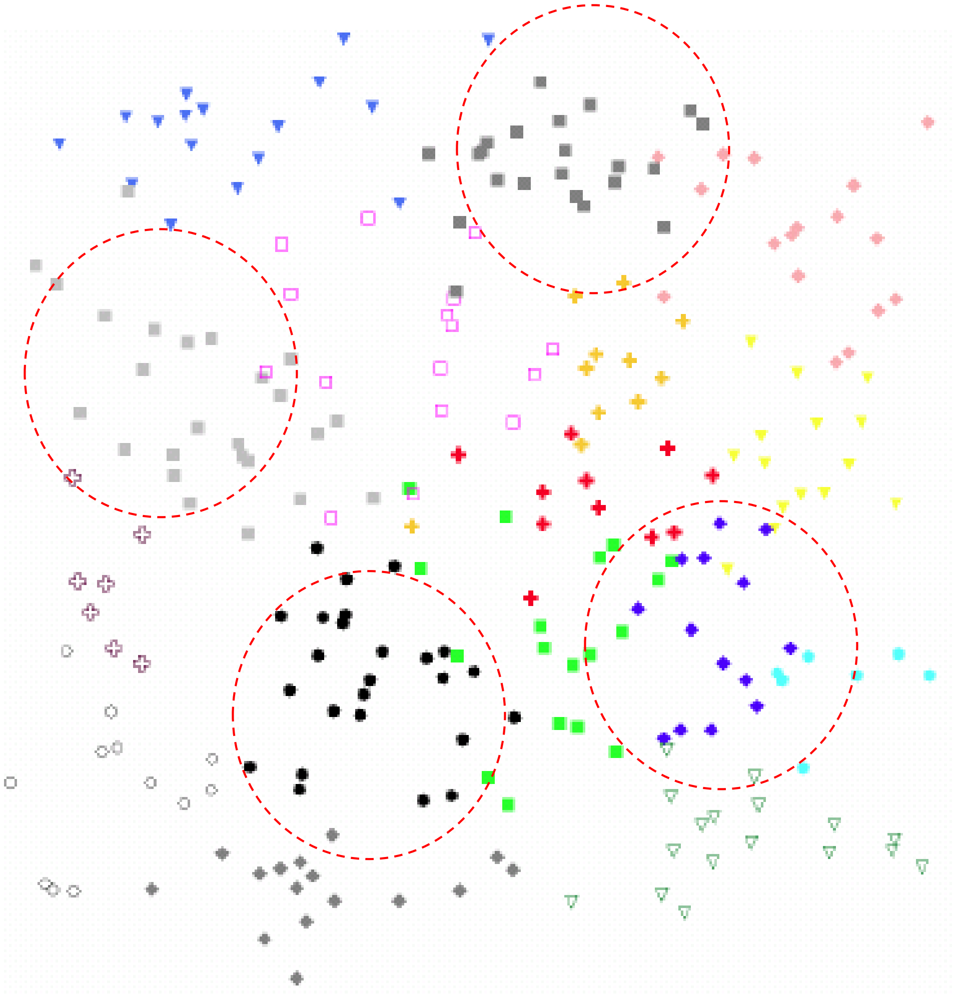
4) The temporary Landmark Node periodically issue beacon messages

5) Node associates itself with closest temporary landmark if it has the same prefixes



6) If need be, a node assigns itself a new overlay id sharing the same prefix with the new closest temporary landmark node.

3.1. MADPastry - Spatial Topology



3.2. MADPastry – Routing Tables

- MADPastry maintains three different routing tables:
  - Stripped down Pastry routing table that only contains Landmark Key
  - Standard Pastry Leaf Set for Indirect Routing (only left and right node accurate)
  - AODV routing table for actual physical routes of overlay hops.

Node 3BBA1234

Pastry Routing Table

row	0	1	2	3	4	5	6	7
0	03761261 nodeID 12	1BE4873B nodeID 78	2BBAEF29 nodeID 117		455D125F nodeID 54	5AC101E6 nodeID 67	6FF47C7A nodeID 151	711C4B01 nodeID 109

Pastry Routing Table cont'd

row	8	9	A	B	C	D	E	F
0	86596535 nodeID 27	9354C24B nodeID 49	A7AA51C6 nodeID 243	B7CF5174 nodeID 126	CA52CE41 nodeID 17	D301A17E nodeID 61	E55C0772 nodeID 81	F105B6FA nodeID 97

Leaf Set

smaller	larger
379E2070 nodeID 47	3CEF7003 nodeID 57
390B56E1 nodeID 72	3D42FE1C nodeID 192
3B76A92E nodeID 63	3DF4102F nodeID 136
3C017EEA nodeID 31	3F02CD52 nodeID 44

AODV Routing Table

Dest	Next Hop	Other	Dest	Next Hop	Other	Dest	Next Hop	Other
12	47	...	57	57	...	109	192	...
17	57	...	61	192	...	117	72	...
27	136	...	63	63	...	126	47	...
31	31	...	67	136	...	136	136	...
44	44	...	72	72	...	151	57	...
47	47	...	78	44	...	192	192	...
49	72	...	81	72	...	243	192	...
54	57	...	97	136	...			

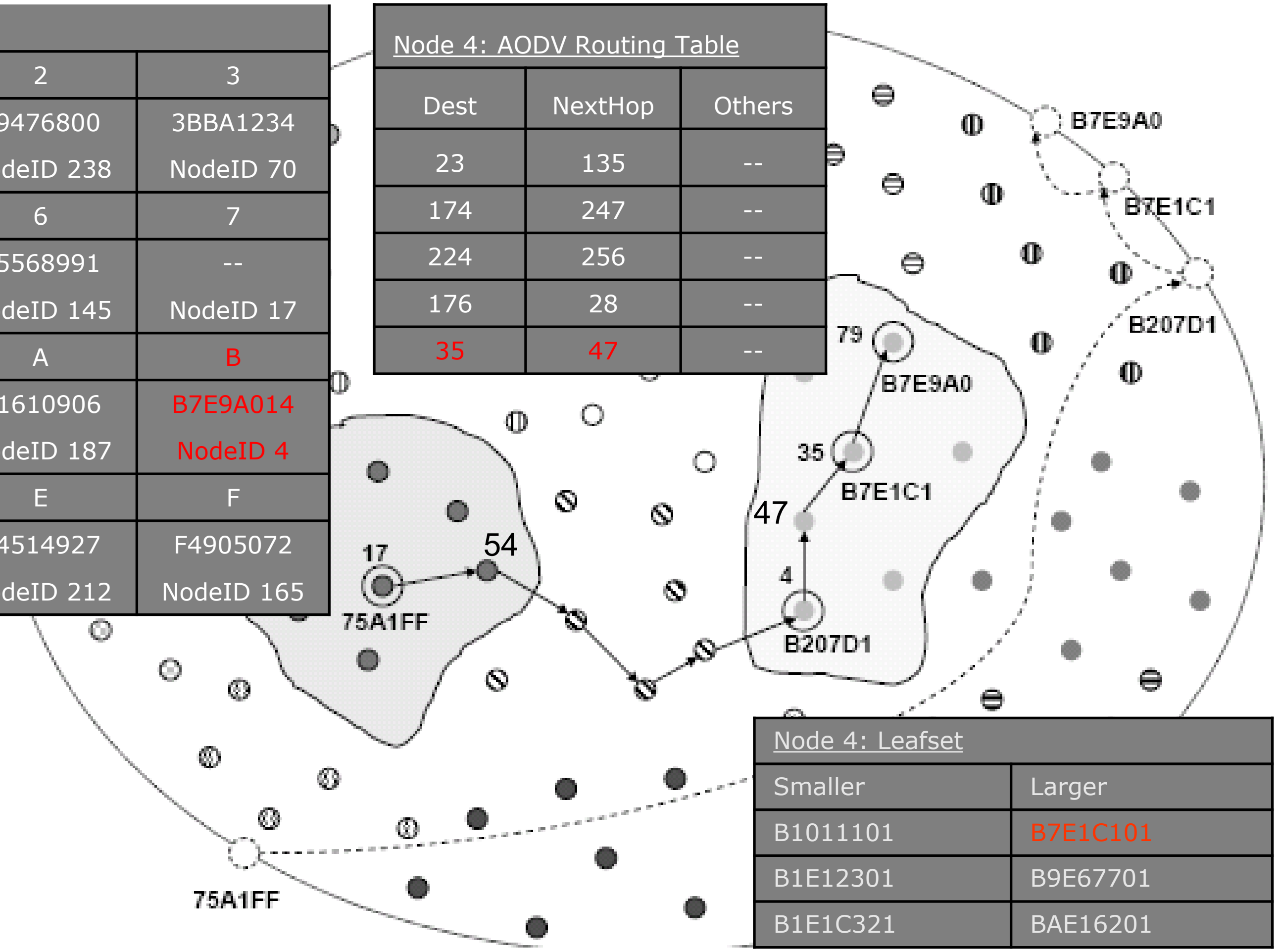
3.2. MADPastry - Routing

- When a node wants to send a packet to a specific key
  1. Consults its Pastry routing table and/or leaf set to determine the closest prefix match, as stipulated by standard Pastry.
  2. Consults its AODV routing table for the physical route to execute this overlay hop.
  3. Intermediate nodes on the physical path of an overlay hop consult their AODV table for the corresponding next physical hop.
  4. When a packet reaches the destination of an overlay hop, that node again consults its Pastry routing table and/or leaf set to determine the next overlay hop.
- This process continues until the packet reaches the eventual target node that is responsible for the packet key (whose overlay id is the numerically closest to the packet key).

3.2. MADPastry - Routing

Node 17: Pastry Routing Table				
Row	0	1	2	3
0	03438687 NodeID 70	16378092 NodeID 88	29476800 NodeID 238	3BBA1234 NodeID 70
	4	5	6	7
	45357677 NodeID 71	58712758 NodeID 184	65568991 NodeID 145	-- NodeID 17
	8	9	A	B
	84219169 NodeID 113	97434904 NodeID 160	A1610906 NodeID 187	B7E9A014 NodeID 4
	C	D	E	F
	C3739014 NodeID 233	D3237999 NodeID 86	E4514927 NodeID 212	F4905072 NodeID 165

Node 4: AODV Routing Table		
Dest	NextHop	Others
23	135	--
174	247	--
224	256	--
176	28	--
35	47	--



Node 17: AODV Routing Table		
Dest	NextHop	Others
56	256	--
132	25	--
61	87	--
85	34	--
4	54	--

Node 4: Leafset	
Smaller	Larger
B1011101	B7E1C101
B1E12301	B9E67701
B1E1C321	BAE16201

3.3. MADPastry's Unicast – Address Publication

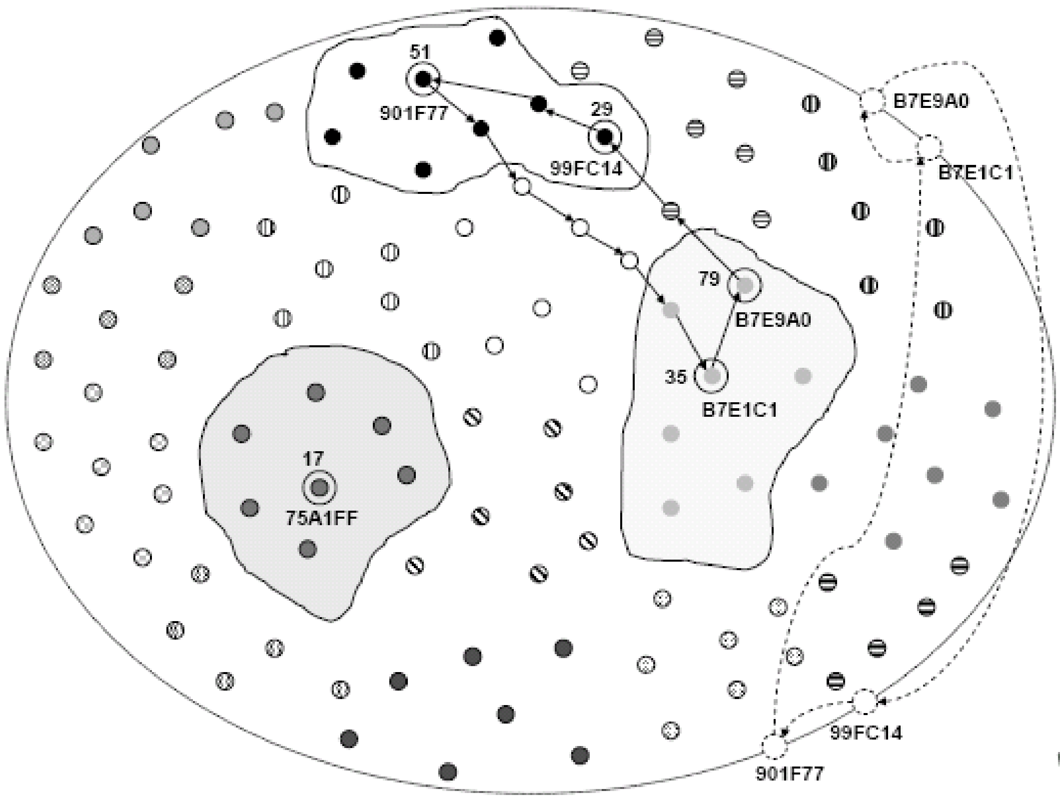
TECHNISCHE UNIVERSITÄT DARMSTADT

- Each nodes has exactly one temporary address server
- Address server stores its client's current overlay ID
- Node A hashes its node ID
  - address server key (ASK).
- Node A publishes its current overlay ID towards ASK
- Node currently responsible for node A's hash key becomes node A's address server

3.3. MADPastry's Unicast – Address Resolution

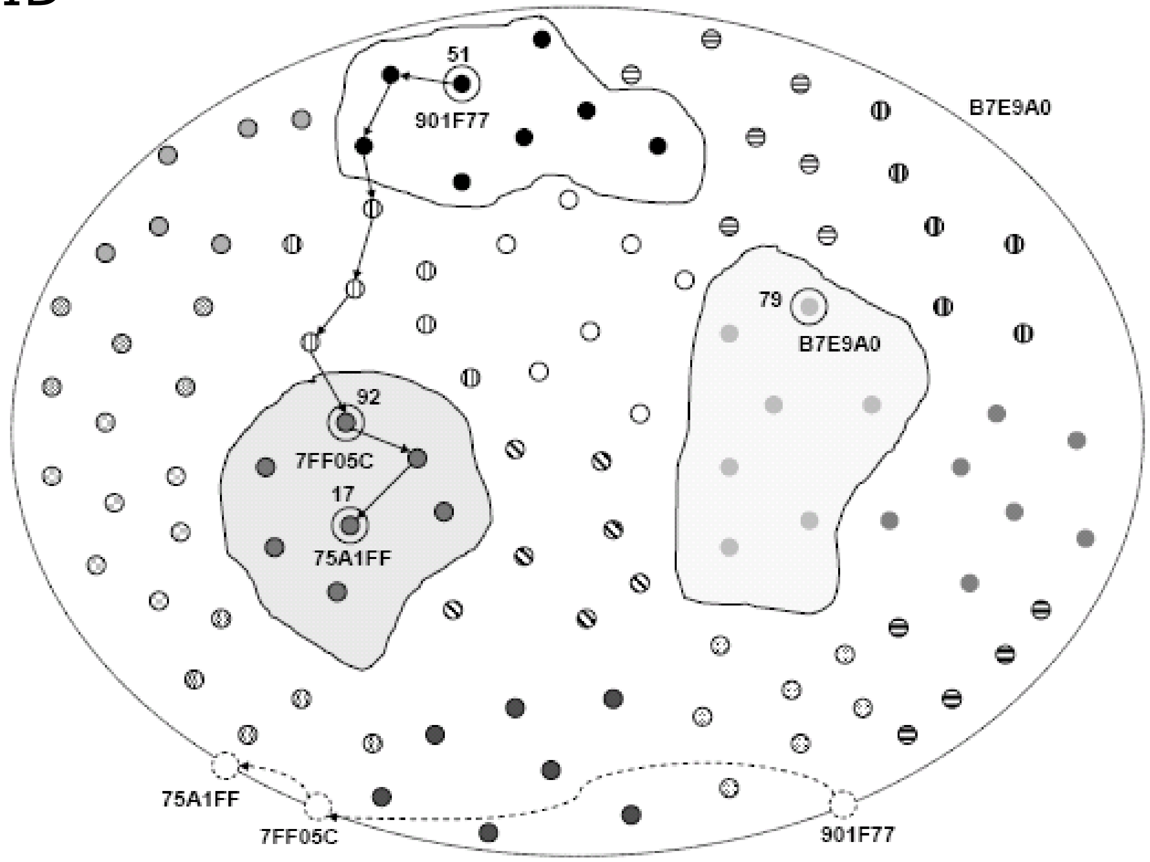
TECHNISCHE UNIVERSITÄT DARMSTADT

- Node A wants to communicate with node B
- Node A does not know node B's current overlay ID
- Node A hashes node B's net ID to get ASK
- Node A sends request towards ASK
- Node B's address server replies with node B's current overlay ID



3.3. MADPastry's Unicast

- Node A uses overlay ID from reply to send message to node B
- MADPastry delivers message using indirect routing



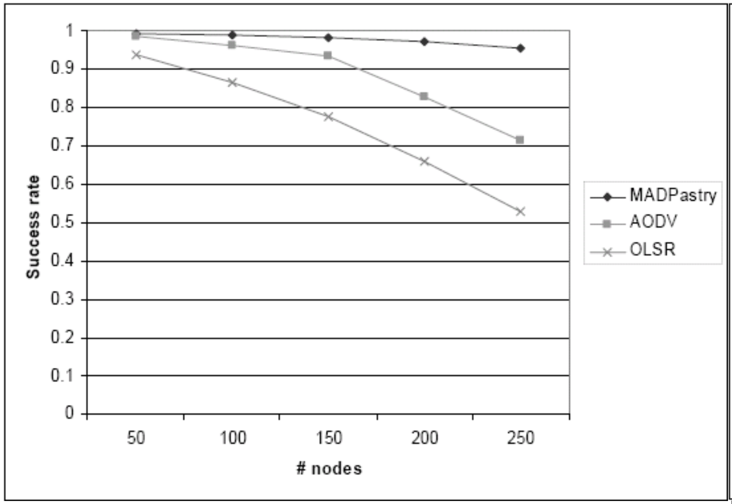
3.4. MADPastry Simulation Results

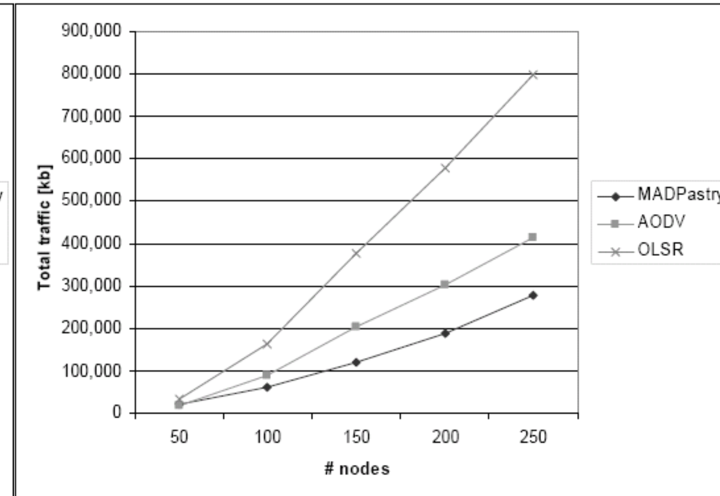
TECHNISCHE UNIVERSITÄT DARMSTADT

- Compare MADPastry's unicast against a popular reactive and proactive ad hoc routing protocol
  - AODV (reactive), OLSR (proactive)
- Simulations in ns2
- Varying network sizes (50,100,150,200,250)
- Varying node velocities (0.1,1.4, 2.5,5.0 m/s)
- 1 random request every 10s per node

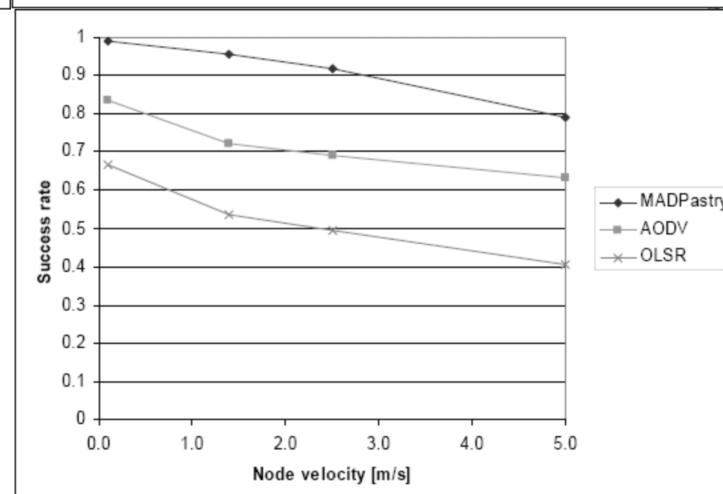
3.4. MADPastry Simulation Results

Fixed velocity:
1.4 m/s





Fixed network size:
250 nodes



$$\text{Success rate} = \frac{\#\,\text{successfully received responses}}{\#\,\text{sent requests}}$$

3.5. MADPastry - Conclusion

- MADPastry's unicast can outperform popular reactive and proactive ad hoc routing protocols
- MADPastry can also provide point-to-point unicasting
  - No need to maintain ad hoc routing protocol in parallel for DHT applications that use MADPastry handle their point-to-point routing
- In MANETs it can be advantageous to travel numerous short up-to-date routes instead of one long direct route