MADPastry: A DHT Substrate for Practicably Sized MANETs

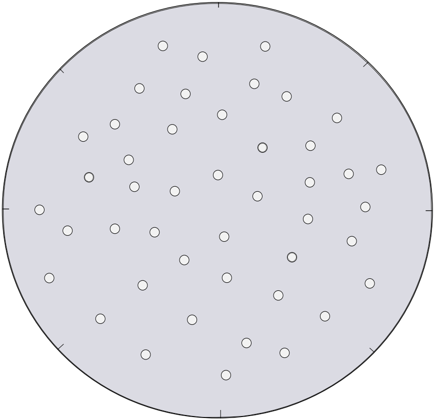
Paper by Thomas Zahn, Jochen Schiller, Institute of Computer Science, Freie Universität Berlin, Germany

Elmar Haussmann

Computer Networks and Telematics, University of Freiburg, Germany

February 15, 2010

Motivation



- How can we know which node provides a specific service?
- How do we route between nodes?

Outline

Ad-hoc On Demand Distance Vector Routing (AODV)

or *How do we route between nodes?*

AODV is:

- a re-active routing protocol for MANETs.
- based on distance vector routing.

It is designed to guarantee:

- loop-freedom by using sequence numbers.
- low bandwidth demand by avoiding global advertisements.
- quick reactions to error situations.

AODV Routing Basics

Idea:

- Each node maintains a monotonically increasing sequence number

Route setup:

1. Request routes by broadcasting a Route Request (RREQ) only when required
2. Wait for a Route Response (RREP) that is unicast back
3. Store freshness of generated routes using the sequence number of the destination

Route Request Travelling from A to E

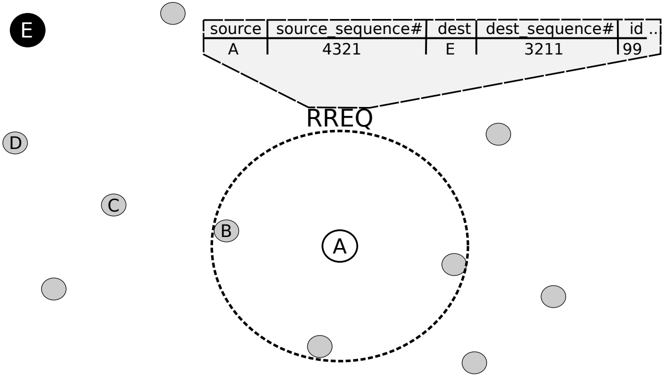


Figure: A route request issued by A.

Route Request Travelling from A to E

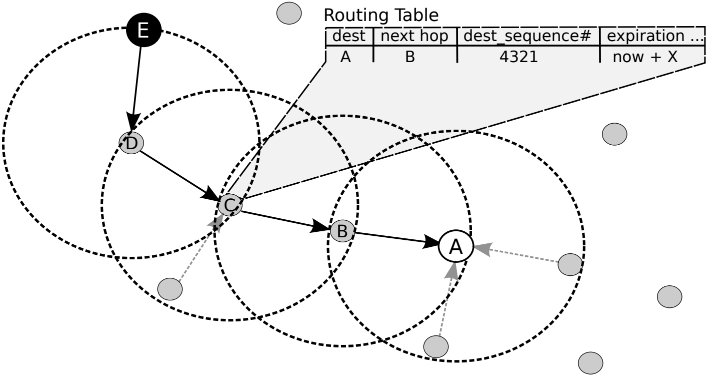


Figure: Reverse Path Setup from E to A. After [2].

Route Response Travelling from E to A

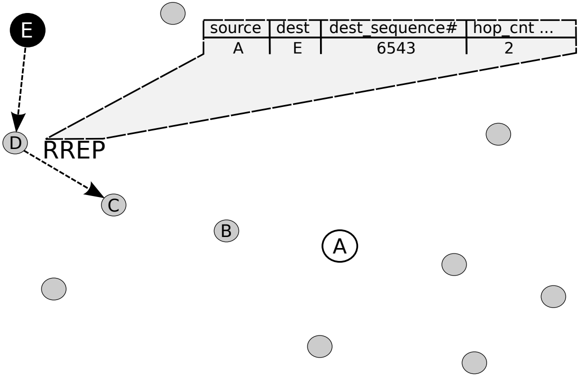


Figure: A Route Reply Issued by E.

Route Response Travelling from E to A

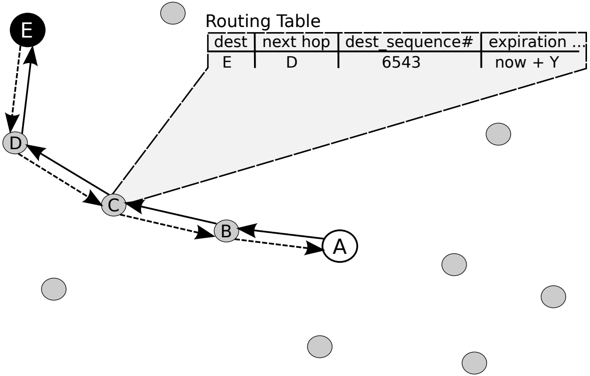


Figure: Forward Path Setup from A to E. After [2].

Resulting Path

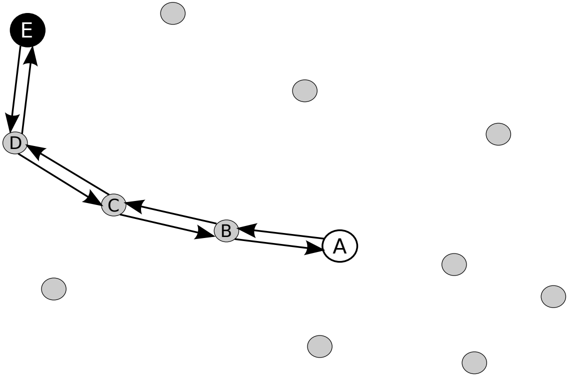


Figure: Resulting Paths Between A and E.

Further Features

- RREQs can be answered by intermediate nodes that have a valid route
- Use of small *Hello* messages to disseminate neighborhood information
- Monitor routes for traffic and drop unused routes
- Notify neighbors using active routes of link failures

Summary

- Broadcast RREQ and unicast back RREP
- Nodes maintain only routes that are needed
- Broadcasts are avoided when possible
- Loop-free routes guaranteed by sequence numbers

Specifically designed and thus ideal candidate for mobile ad hoc networks

Outline

Pastry

or *How do we know which node provides a service?*

Pastry:

- A peer-to-peer overlay network based on Distributed Hash Tables
- Developed in Cambridge (Microsoft Research)
- Provides fault-tolerant, scalable object/service location in distributed networks
- Optimized for latency to the underlay

Basic Network Structure 1/2

- Each peer and object has a unique 128-bit ID distributed uniformly
- IDs are interpreted to base $B = 2^b$ (usually $B = 16$)
- Peers are responsible for the objects they are numerically closest to

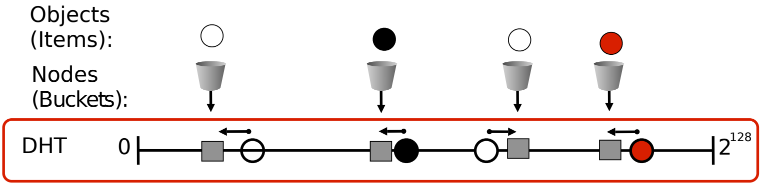


Figure: DHT Illustration after [4].

Basic Network Structure 2/2

- Given an object ID Pastry indirectly routes a message to the peer responsible for it
- Service lookup possible by e.g. hashing some metadata to ID space and sending a message to node responsible for it
- Pastry only delivers message to destination, everything else part of application implementation
- Each node maintains three sets of peers: routing table R, leaf set L, neighborhood set M

Routing Table R

Each entry associates a node ID with an (IP-)address

Definition

The routing table of each node p contains for each prefix z of p's ID a node with prefix $z \circ j$ ($j \in B$ and $z \circ j$ no prefix of p) [1]

Example:

- $B = 16 \Rightarrow j \in 0, 1, 2, 3 ... E, F$
- Node ID of p 75A10F and $z =$75A
- p knows nodes with prefix 75A[0,2-F]

Routing Table Example

NodeId 3627

Leaf set

SMALLER				LARGER			
3214	3352	3417	3521	3672	3721	4324	4621

Routing table

-0-312	-1-132	-2-632	**3**	-4-324	-5-321	-6-023	-7-155
3-0-43	3-1-27	3-2-14	3-3-52	3-4-17	3-5-21	**6**	3-7-21
36-0-1	36-1-5	**2**	-	-	-	-	36-7-2
362-0-	-	-					**7**

Neighborhood set

1132	2114	2721	5321	5145	5412	7155	7713

Figure: Pastry routing table example for $B = 8$. After [3].

Leaf Set L

Definition

The leaf set contains the $|L|/2$ nodes with next higher ID and the $|L|/2$ nodes with next lower ID

A ring-structure is formed through all overlay nodes

Leaf Set Example

NodeId 3627

Leaf set

SMALLER				LARGER			
3214	3352	3417	3521	3672	3721	4324	4621

Routing table

-0-312	-1-132	-2-632	**3**	-4-324	-5-321	-6-023	-7-155
3-0-43	3-1-27	3-2-14	3-3-52	3-4-17	3-5-21	**6**	3-7-21
36-0-1	36-1-5	**2**	-	-	-	-	36-7-2
362-0-	-	-					**7**

Neighborhood set

1132	2114	2721	5321	5145	5412	7155	7713

Figure: Pastry leaf set example for $|L| = 8$. After [3].

A Graphical Example of Routing Table and Leaf Set

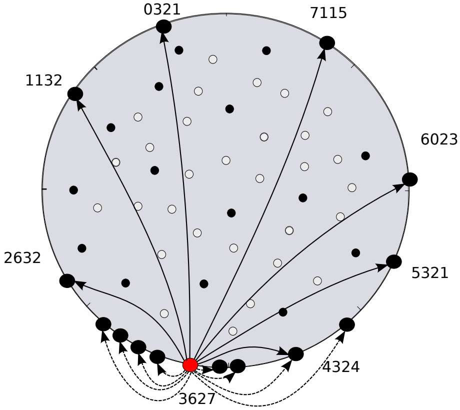


Figure: Illustration of First Pastry Routing Table Row and Leaf Set. After [4].

Routing Algorithm of Pastry

Assume node p sends a message M to some ID X

- if the ID X is already within the range of the leaf set of p send it to the node with smallest numerical distance to X
- otherwise forward to a node p' that shares at least one more prefix-digit with X than p
  - if no such node exists send to a node p' that shares the same prefix length with X but is numerically closer

A Graphical Pastry Routing Example

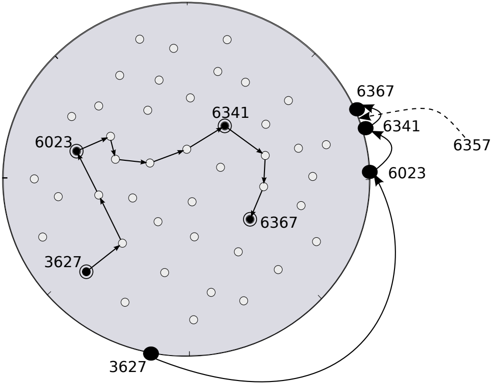


Figure: Pastry routing from 3627 to 6357. After [6].

Pastry Maintenance and Routing Performance

- Maintenance, join and repair procedures try to keep routing tables consistent
- Routing always converges if the leaf set is correct
- Expected number of routing hops $O(log_{2^b} N)$
- Worst case $O(N/|L|)$
- Close overlay IDs have no direct relation to physical proximity (overlay stretch)

Outline

Motivation for MADPastry

- Current P2P protocols are designed for the Internet
- Current P2P networks do not consider physical proximity their top priority
- A lot of MANET routing protocols need to revert to broadcasting
- Maintenance of DHTs is expensive

Zahn and Schiller combined and modified Pastry and AODV at the network layer to provide efficient indirect routing in MANETs: MADPastry

MADPastry Clusters

- In Pastry nodes close in the overlay can be arbitrarily far apart physically
- Introduce clusters
  - Define landmark keys dividing ID space equally
  - Nodes with ID closest to one of the landmark keys become landmark nodes
  - Landmark nodes send broadcast beacons disseminated by nodes of own cluster
  - Nodes join cluster of closest landmark node by assigning new ID with cluster-prefix

MADPastry Clusters Example

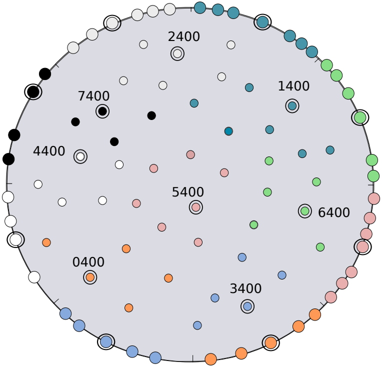


Figure: MADPastry Clusters Example. After [5].

Nodes that are physically close are also close in the overlay

MADPastry Routing Tables

The large routing table and leaf set of Pastry causes maintenance overhead

→ Reduce routing table size

- Use only one row with $log_{2^b} K$ columns
  (K number of landmark keys)
- Store pointer into each cluster

→ Reduce maintenance of leaf set

- Only proactively maintain closest left and right neighbor

- Each node also maintains standard AODV routing table

MADPastry Routing Table and Leaf Set Example

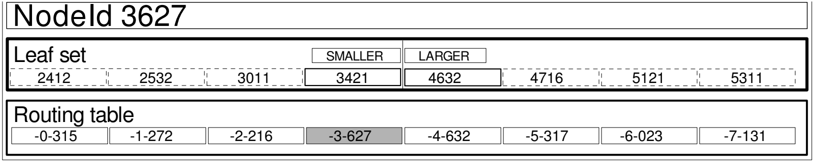


Figure: MADPastry routing table and leaf set for $K = 8$ and $B = 8$. After [3].

MADPastry Routing Procedure

Routing integrates physical and overlay routing:

Assume node p receives message to target ID X with overlay hop destination Y

- p can be the target of overlay hop ($p = Y$)
  - Proceed with standard Pastry routing
- p can be intermediate node on physical path of overlay hop
  - If $p's$ overlay ID is closer to X than Y consider overlay hop done
  - Else use AODV to route to overlay hop destination

ADOV Routing Procedure

What if AODV routing table contains no next hop for a physical destination?

- If node is already within target cluster broadcast the message within the cluster to the eventual target
- Else use standard AODV route discovery

MADPastry Routing Example

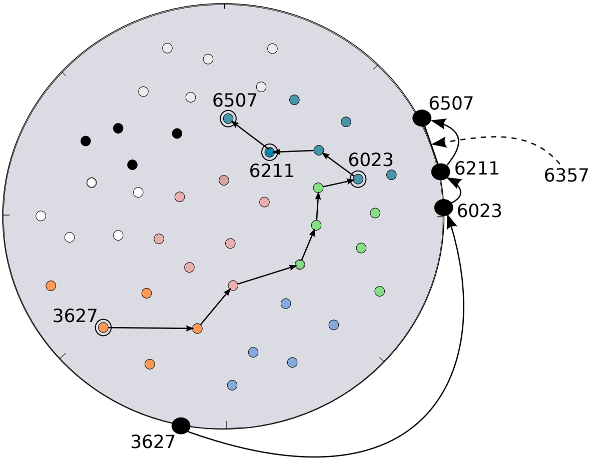


Figure: MADPastry Routing Example from 3627 to 6357. After [6].

MADPastry Routing Broadcast Example

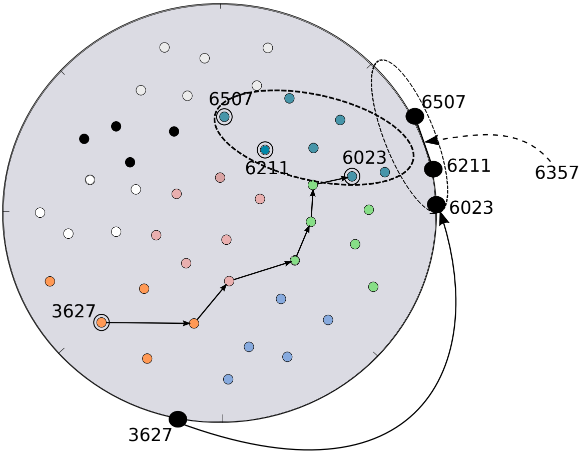


Figure: MADPastry Routing Example from 3627 to 6357 with Broadcast. After [6].

Further Improvements

- Use overhearing of packets
  - Extract information of overheard packets and fill AODV or Pastry routing table
- Each node periodically sends beacons through own cluster to fill leaf sets

MADPastry Packet

source sequence_#	previous phys. hop sequence_#	overlay ID source	overlay ID previous phys. hop	...

Figure: MADPastry packet contents extract.

Problems Caused by Changes 1/2

- MADPastry routing table is smaller than Pastry routing table - sacrifices scalability
  - Authors consider MANETs with up to 1000 nodes realistic
  - Given e.g. $K = 16$, $b = 4$, $L = 16$, clusters with 60 nodes are formed
  - One hop to target cluster and for intra-cluster routing 8 hops worst case $(62.5/|L|/2)$

Problems Caused by Changes 2/2

- Pastry leaf set only guarantees correct left and right neighbor
  - Enough to guarantee correct routing
  - Overhearing of packets additionally fills leaf set
- Overhearing of packets does not perform in low traffic networks
  - According to authors MADPastry not designed for low traffic networks
  - Use broadcast agent instead

Outline

Simulation Scenario

- Use ns-2 for simulation
- Model MANETs with varying either:
  - Network size: 100 or 250 nodes
  - Node speed: between 0.1 m/s and 5 m/s
  - Lookup rate: 1 lookup per 1s, 10s and 60s
- A square plane with 100 nodes/km^2 is used
- Compare to broadcast agent, and Pastry with AODV implementation without clusters

Success Rate

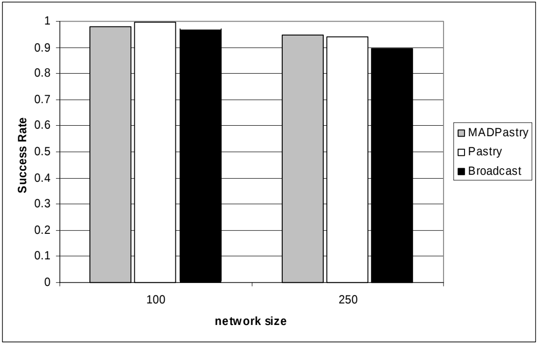


Figure: MADPastry Success Rates. From [5].

Success rates are similiar or better

Traffic

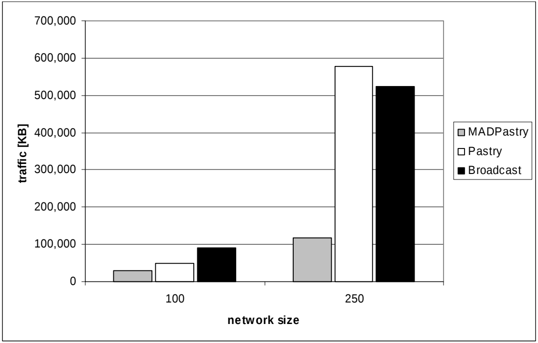


Figure: Traffic Generated by MADPastry. From [5].

Consideration of physical locality important

Node Speed

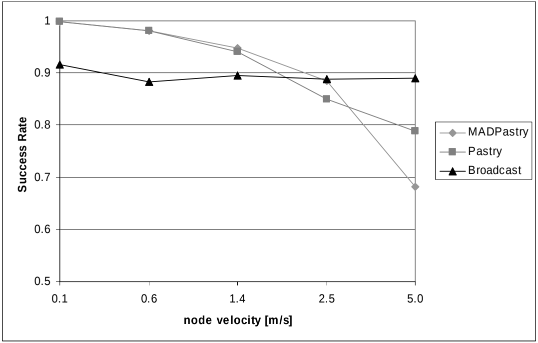


Figure: MADPastry Performance with Various Node Speeds. From [5].

Routes break and nodes join and leave clusters frequently

Handovers

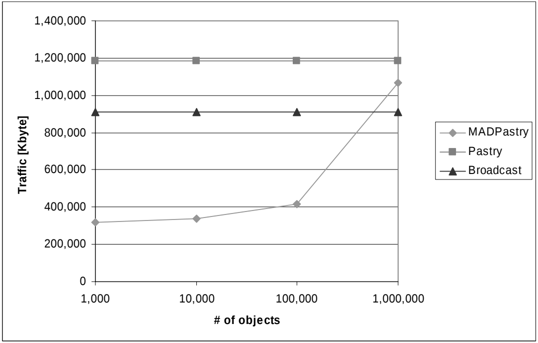


Figure: Effect of Handovers on MADPastry Performance. From [5].

Number of objects affects performance

Back to Motivation

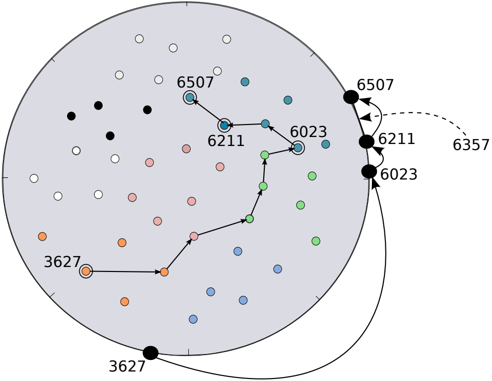


Figure: MADPastry routing. After [6].

- How can we know which node provides a specific service?
- How do we route from A to B?

Hash e.g. GPS Service to ID space and use MADPastry to route there

What MADPastry Provides

- Distributed application services (e.g. service lookup) can efficiently be provided in MANETs using e.g. MADPastry
- Certain limits apply (speed, lookup rates, number of objects)
- Simulation results indicate that explicit representation of locality is essential
- Concept of integrating application and network layer pays off

Open Questions and Remarks

- The number of P2P applications developed dictates the need for P2P networks in MANETs
- Why Pastry?
- Is the assumption of 1000 node MANETs realistic (in future)?
- MADPastry will not scale well for (very) large networks
- Are there maybe simpler solutions?
- Will other routing protocols perform better?

Thank you for your attention.

Thank you for your attention.

References I

Peter Mahlmann and Christian Schindelhauer.
Peer-to-Peer Netzwerke.
Springer-Verlag, 2007.

Charles Perkins and Elizabeth Royer.
Ad-hoc on-demand distance vector routing.
In *In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1997.

References II

Antony I. T. Rowstron and Peter Druschel.
Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems.
In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350, London, UK, 2001. Springer-Verlag.

Christian Schindelhauer.
Lecture algorithms and methods for distributed storage, Wintersemester 2008/2009.

References III

Thomas Zahn and Jochen Schiller.
MADPastry: A DHT Substrate for Practicably Sized MANETs.
In *Proc. of the 5th Workshop on Applications and Services in Wireless Networks (ASWN2005)*, 2005.

Thomas Zahn and Jochen Schiller.
DHT-based Unicast for Mobile Ad Hoc Networks.
In *Proceedings of the 4th IEEE PerCom Workshops. 3rd IEEE International Workshop on Mobile Peer-to-Peer Computing (MP2P'06)*, Washington, DC, USA, 2006. IEEE Computer Society.