Performance Analysis of Stealth Distributed Hash Table with Mobile Nodes

Andy MacQuire, **Andrew Brampton**, Idris A. Rai and Laurent Mathy
Computing Department
Lancaster University, UK

Friday 18th March / Mobile Peer-to-Peer 2006

LANCASTER UNIVERSITY

Outline

LANCASTER UNIVERSITY

Problems with DHTs and Mobility?

- Assumes homogeneity
  - All nodes are treated equally (routing, storing etc.)
  - Similar bandwidth, processing power, uptime
  - Mobile environments are very heterogenous!
- Security (or lack thereof)
  - Sniffing, Man in the Middle, Routing Table Poisoning
  - Difficulties in supporting user authentication
  - Very easy to join/sniff wifi networks, need for increase security
- Churn
  - Wait for next slide...

LANCASTER UNIVERSITY

Problems with DHTs and Mobility?

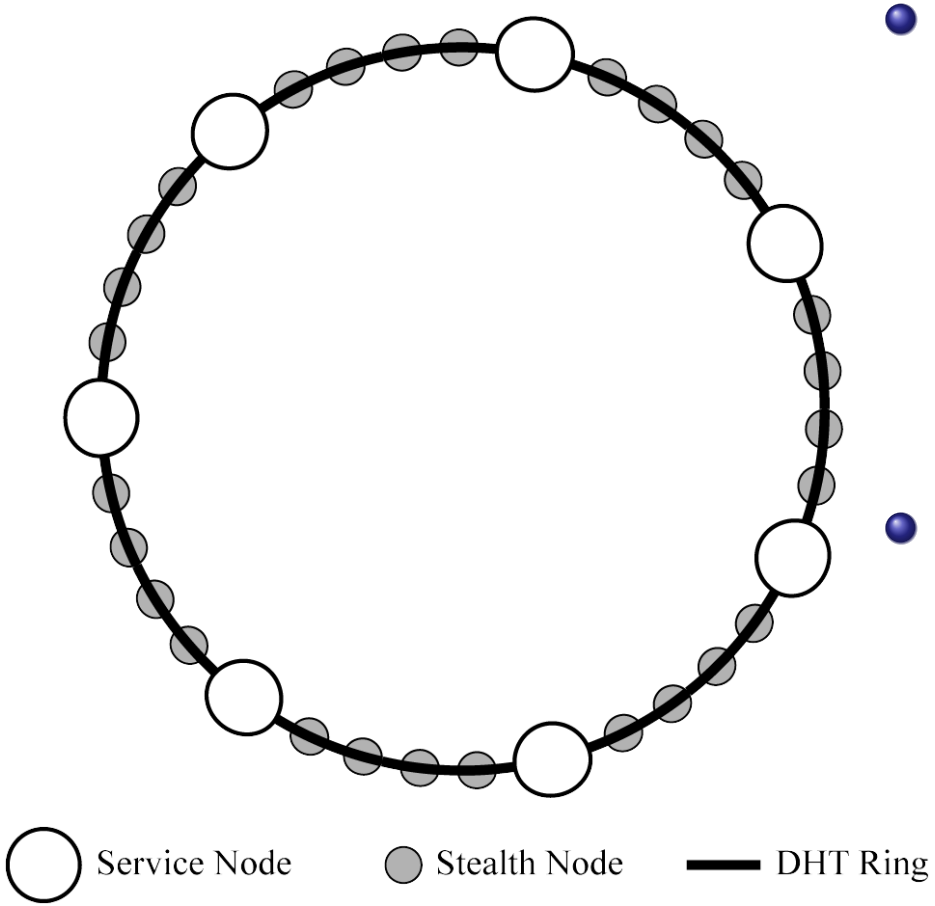
Mobility Churn

- Join forces routing updates
- Leaves make the routing tables stale

When does this happen?

- Loses connectivity when out of range
- Batteries prone to running dry
- When changing Access Point
  - Hand-over time
  - May retain IP address (No need to rejoin, but data may be stale)
  - May change IP address (May need to rejoin, or re-announce)

What are Stealth DHTs?



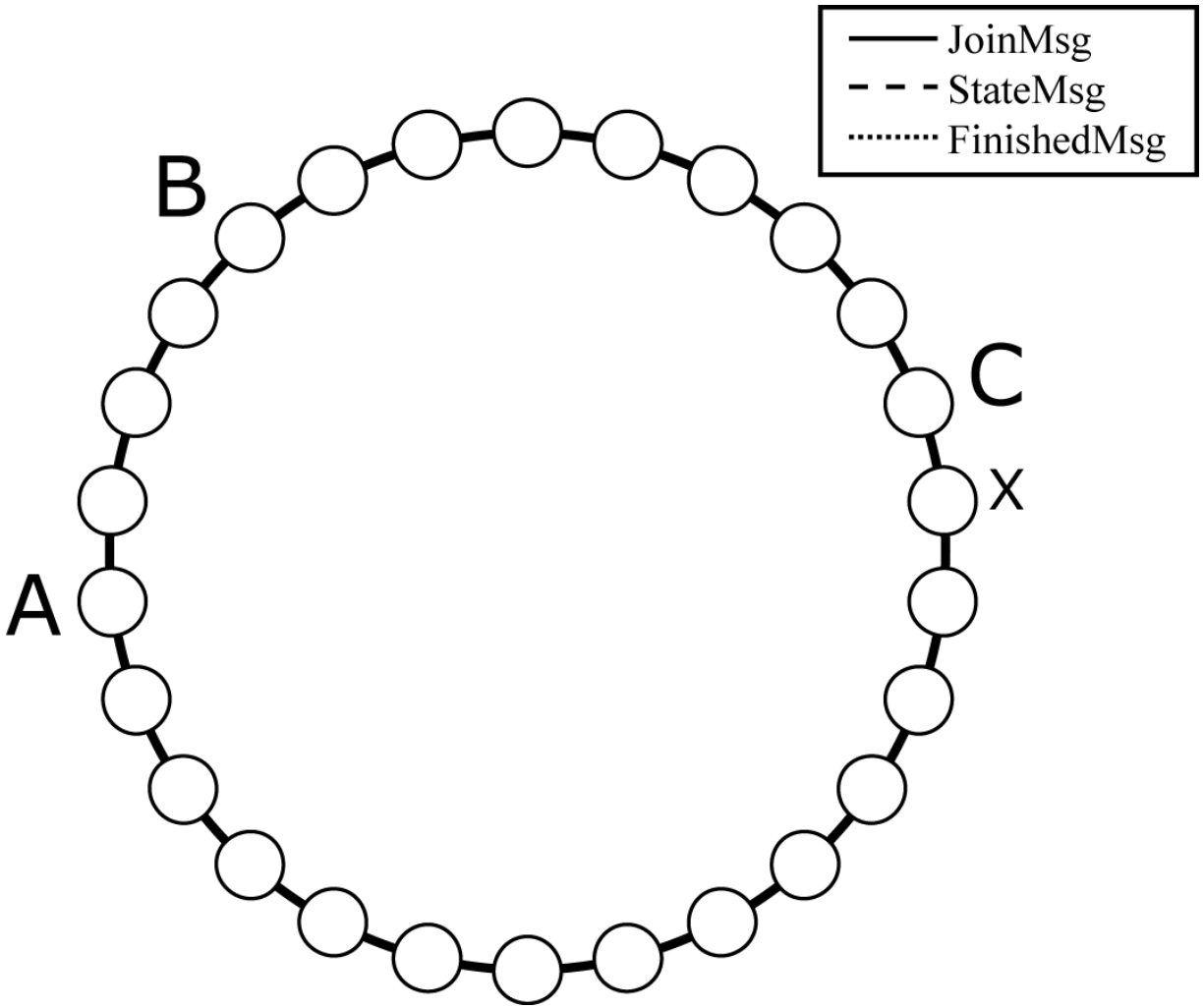
- Service Nodes
  - Assumed to be the more capable nodes
  - Responsible for forwarding and storing data
  - e.g. Wired Node
- Stealth Nodes
  - Assumed to be the less capable nodes
  - Have no responsibilities
  - Invisible to all nodes, including Service nodes
  - e.g. Wireless Node

LANCASTER UNIVERSITY

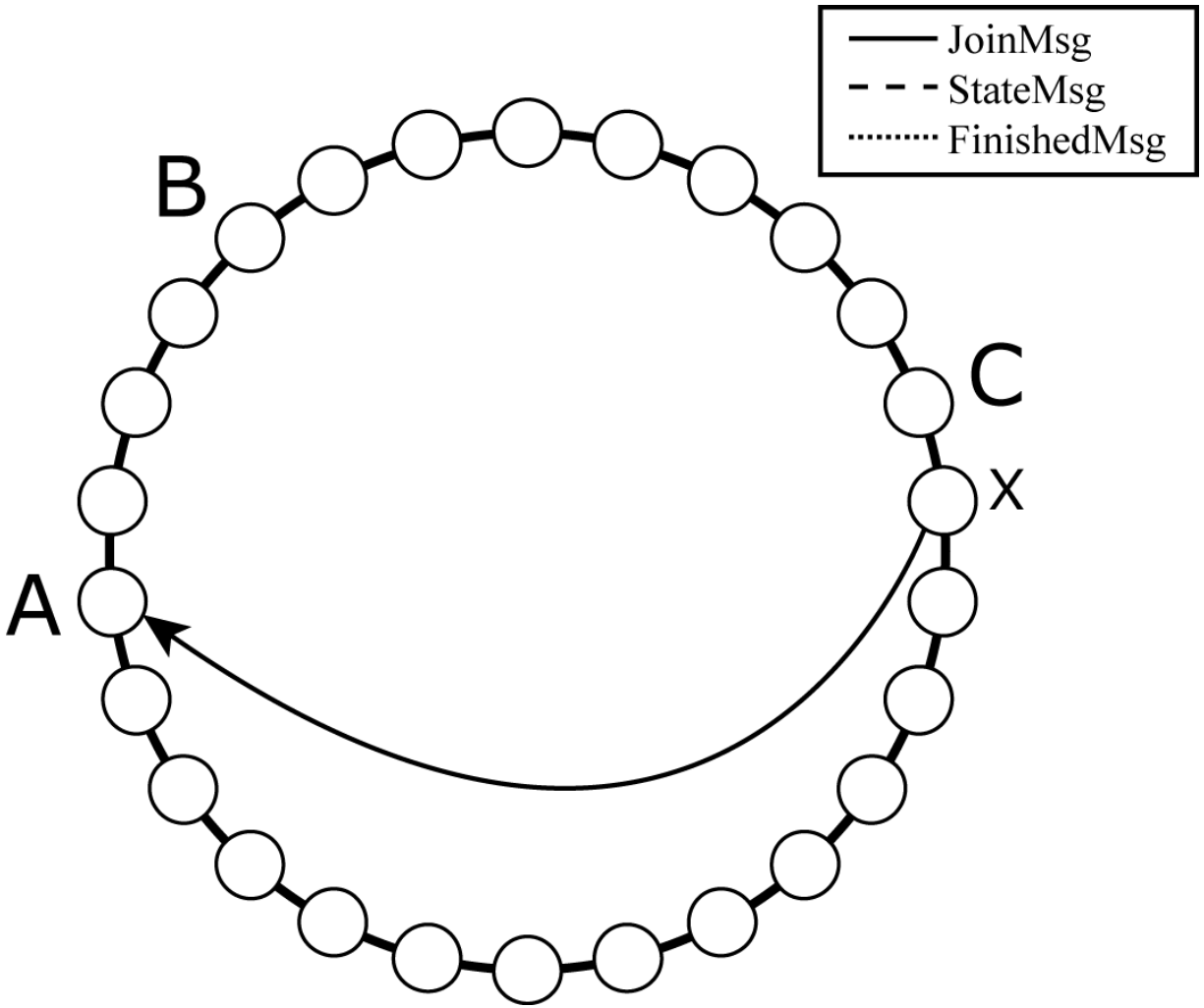
How does it help?

- Does not assume homogeneity
  - Nodes can now be divided based on their capabilities
  - More powerful nodes, take more responsibility
- Churn (joins and leaves) generates little to no overhead
  - Stealth join requires less overhead
  - Joining of Stealth nodes does not require routing updates
  - Stealth nodes leaving does not make routing tables stale
- Security ~~(or lack thereof)~~
  - Authentication for the Service nodes ensure that only authorised nodes route and store message
  - Stealth nodes cannot commit sniffing, corruption attacks

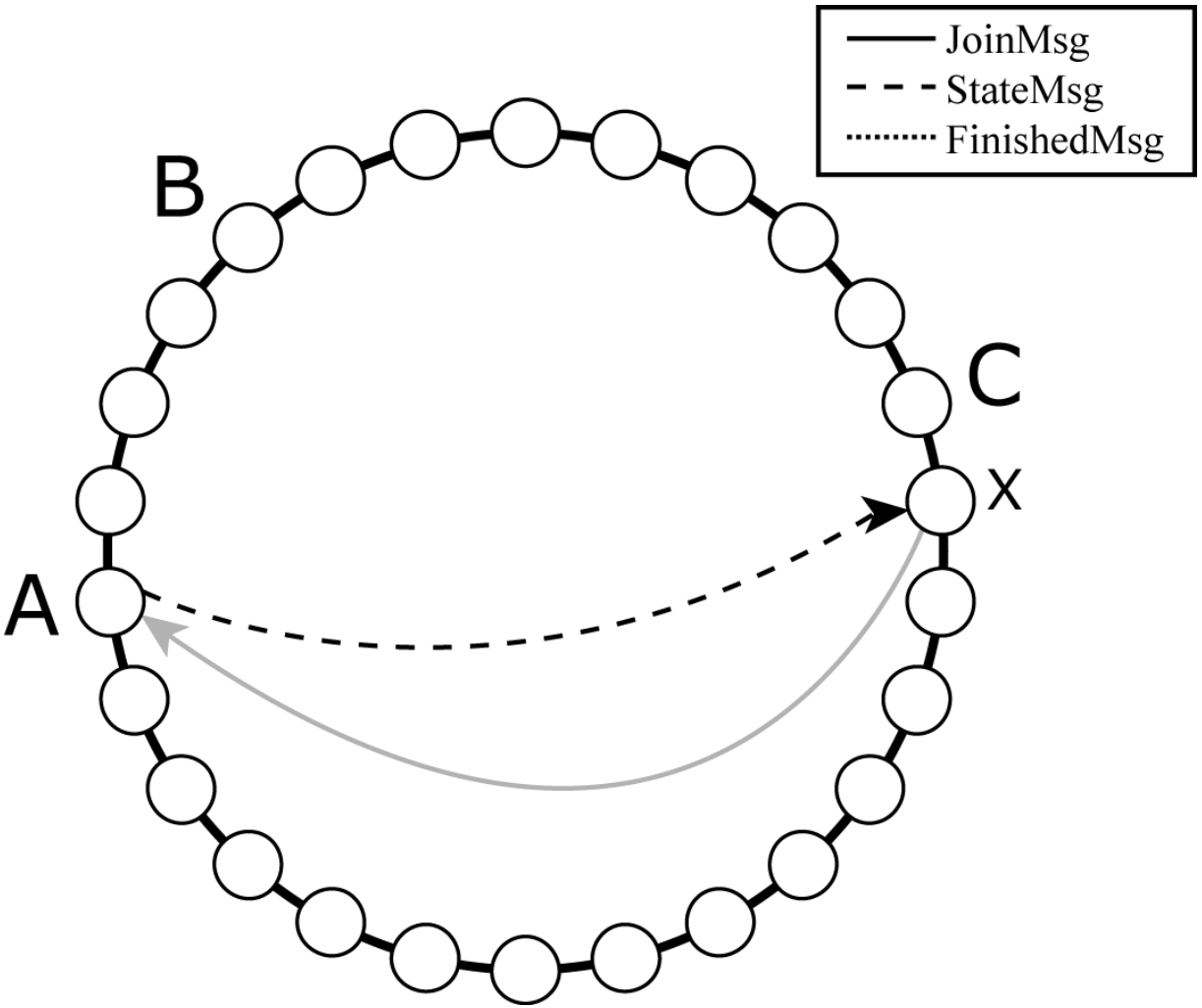
Pastry's Join - State Gathering



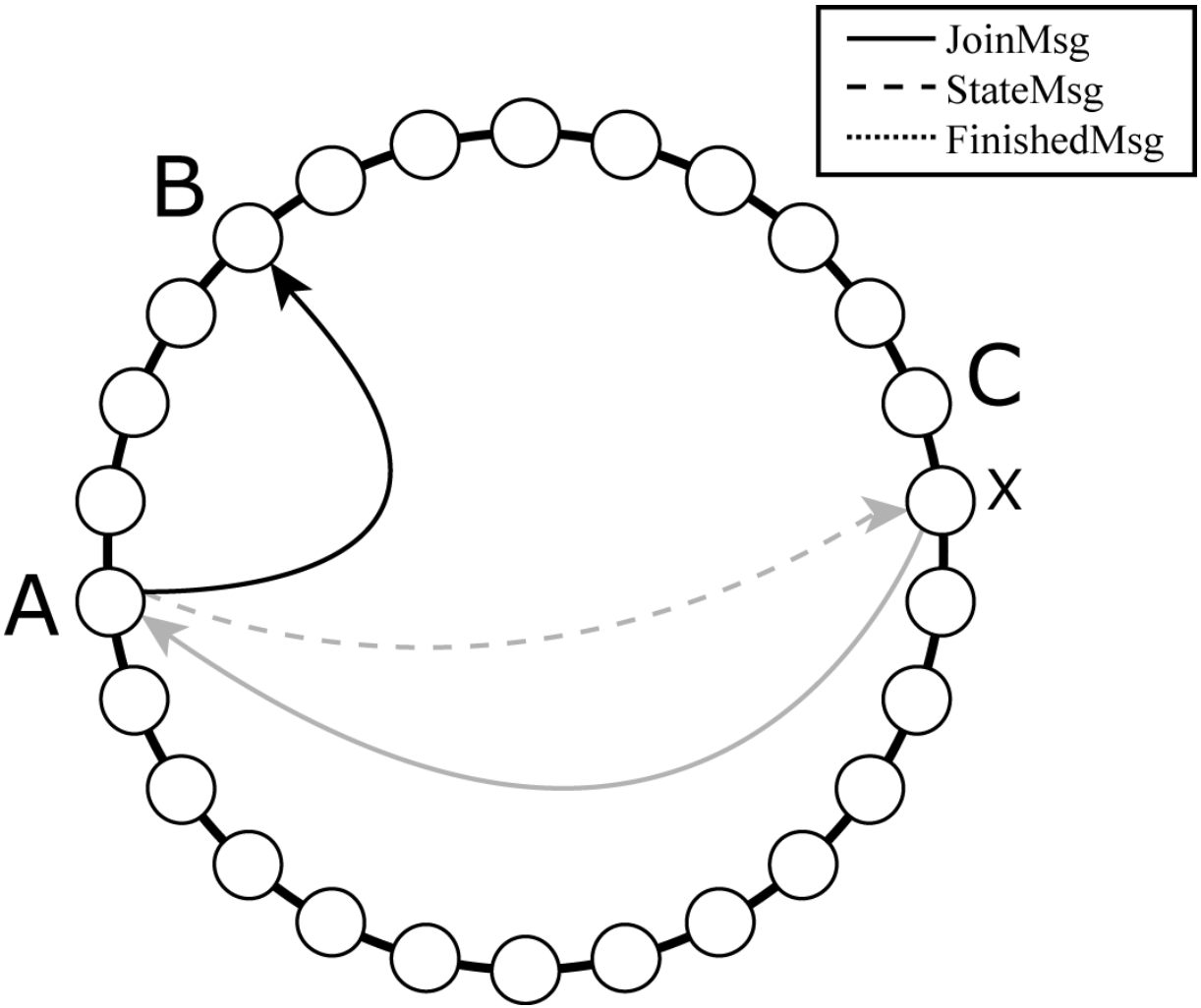
Pastry's Join - State Gathering



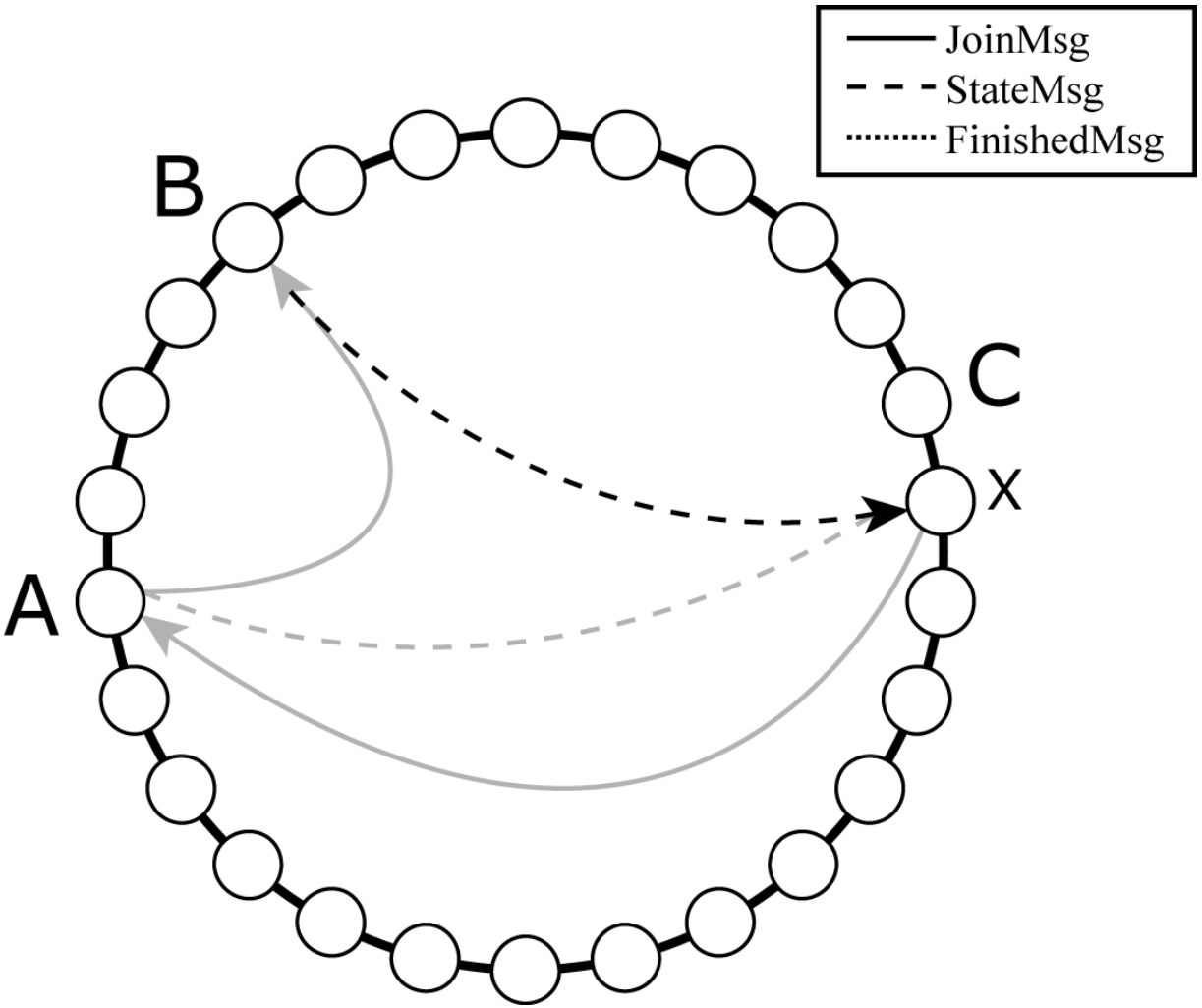
Pastry's Join - State Gathering



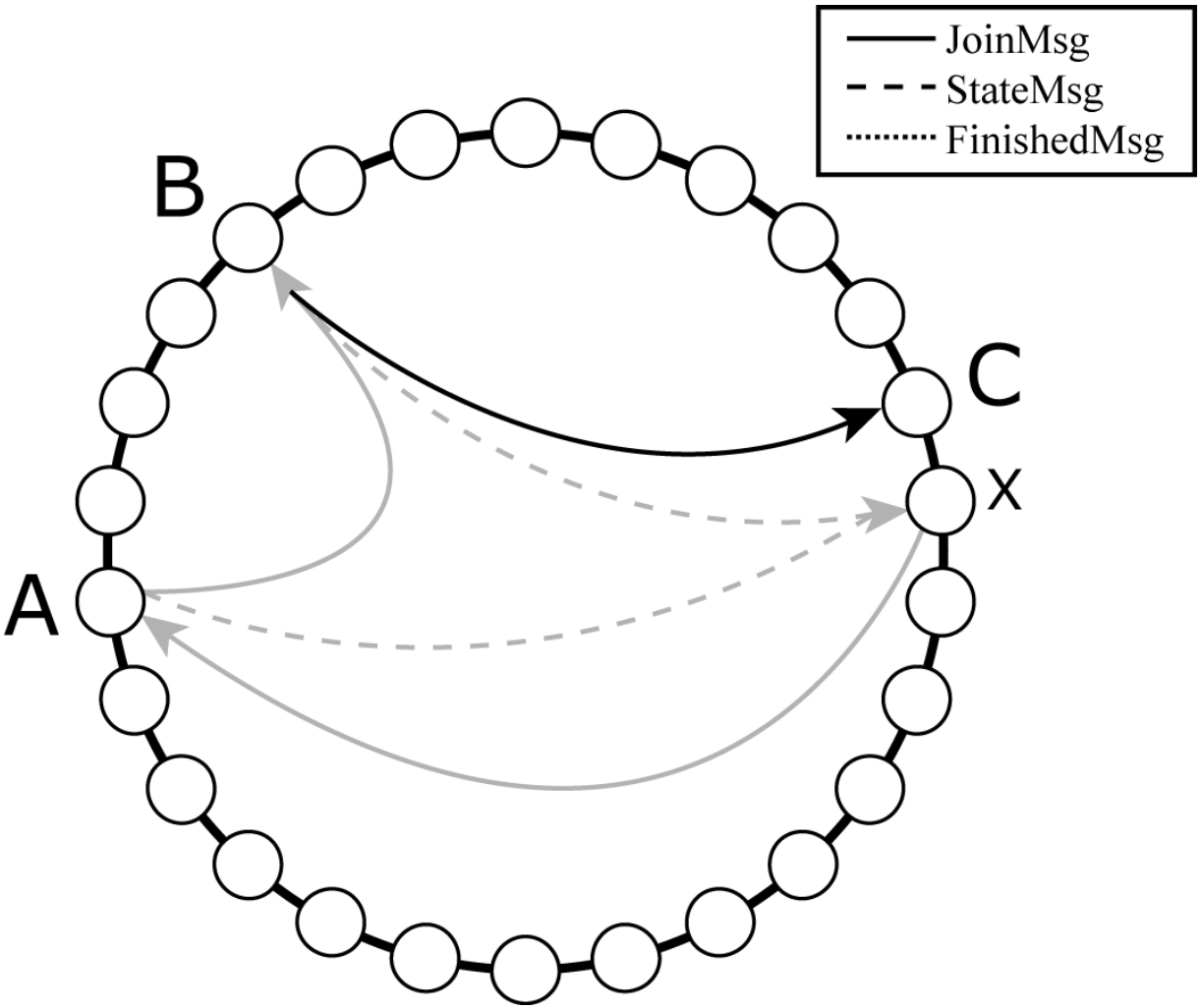
Pastry's Join - State Gathering



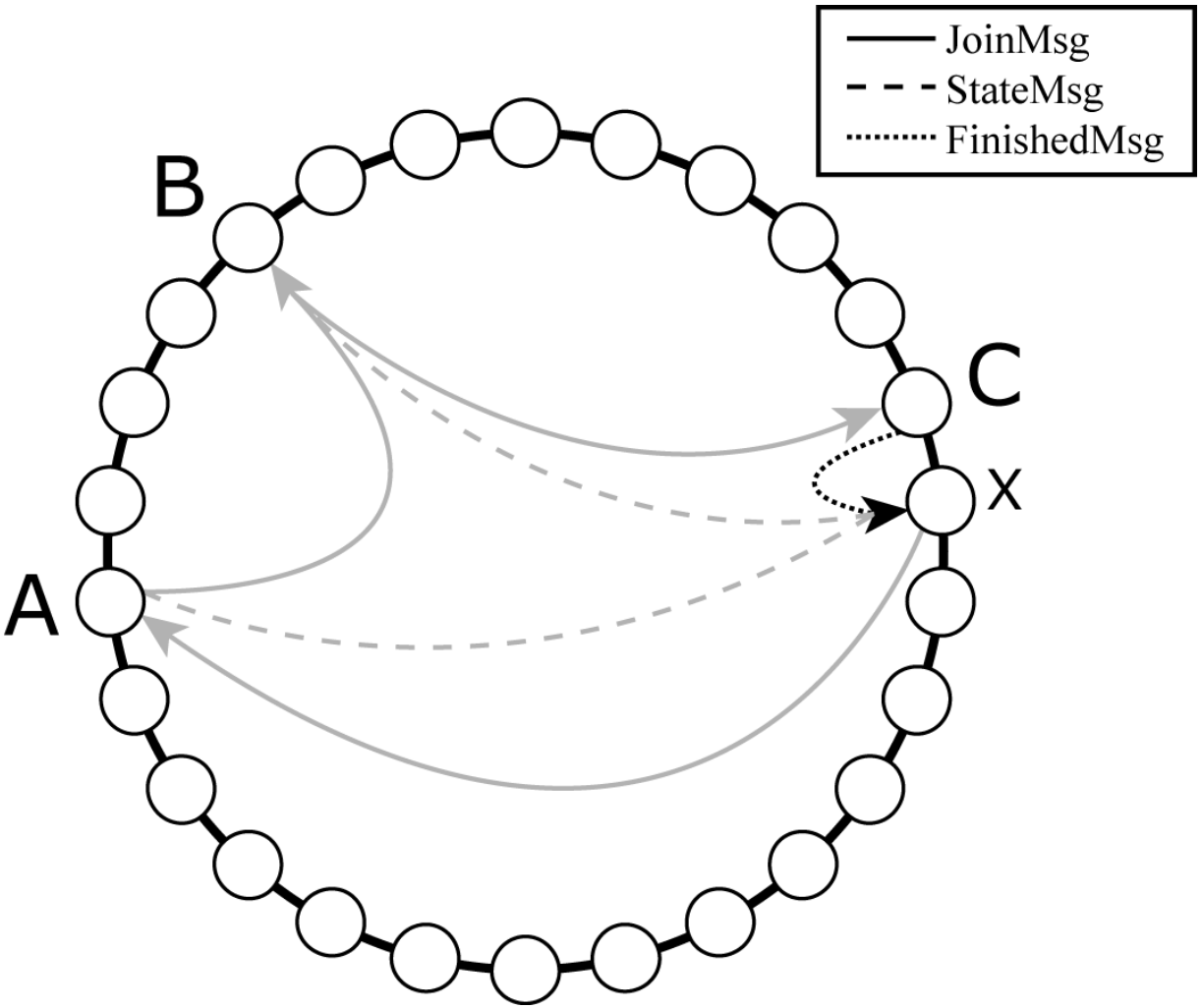
Pastry's Join - State Gathering



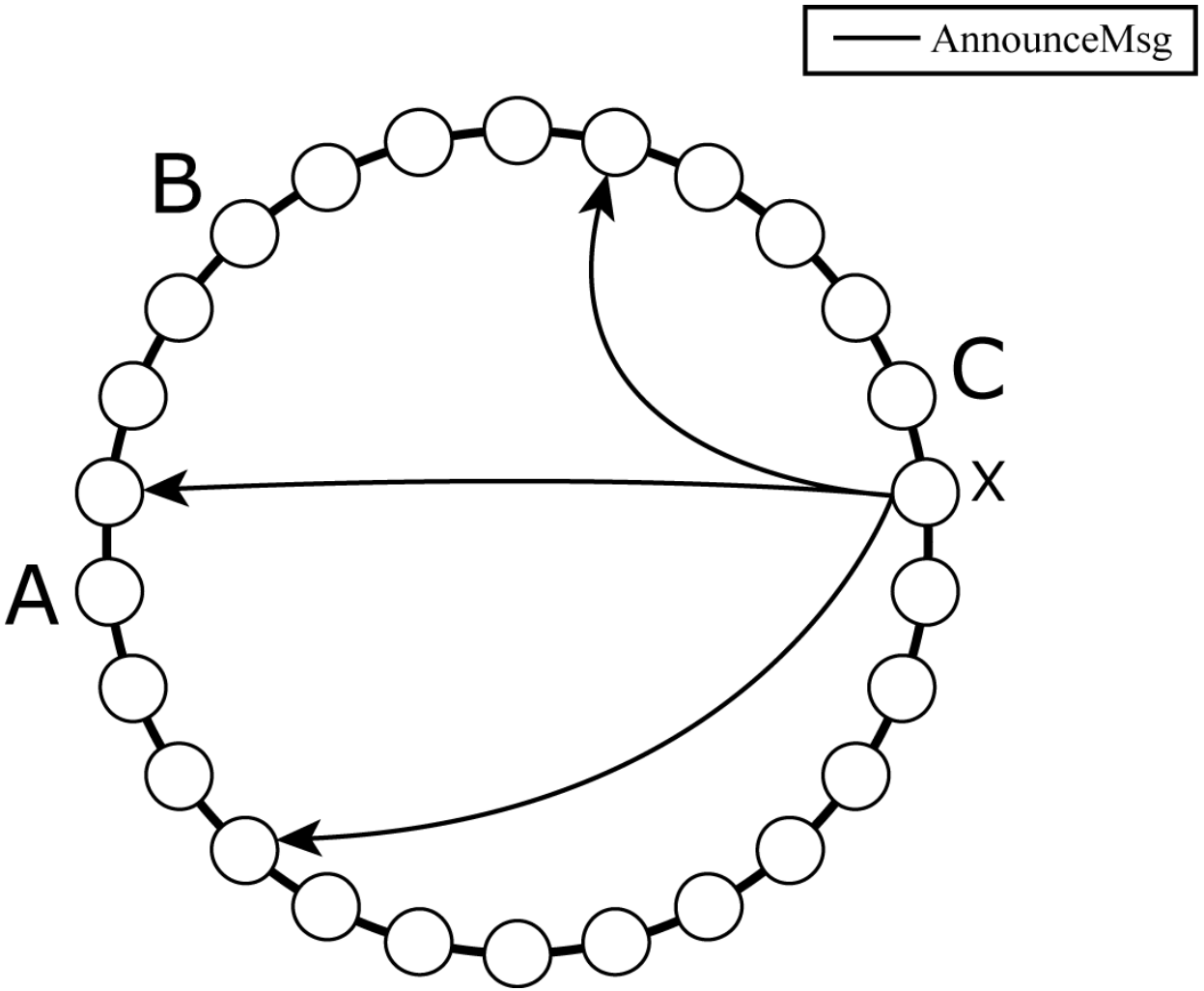
Pastry's Join - State Gathering



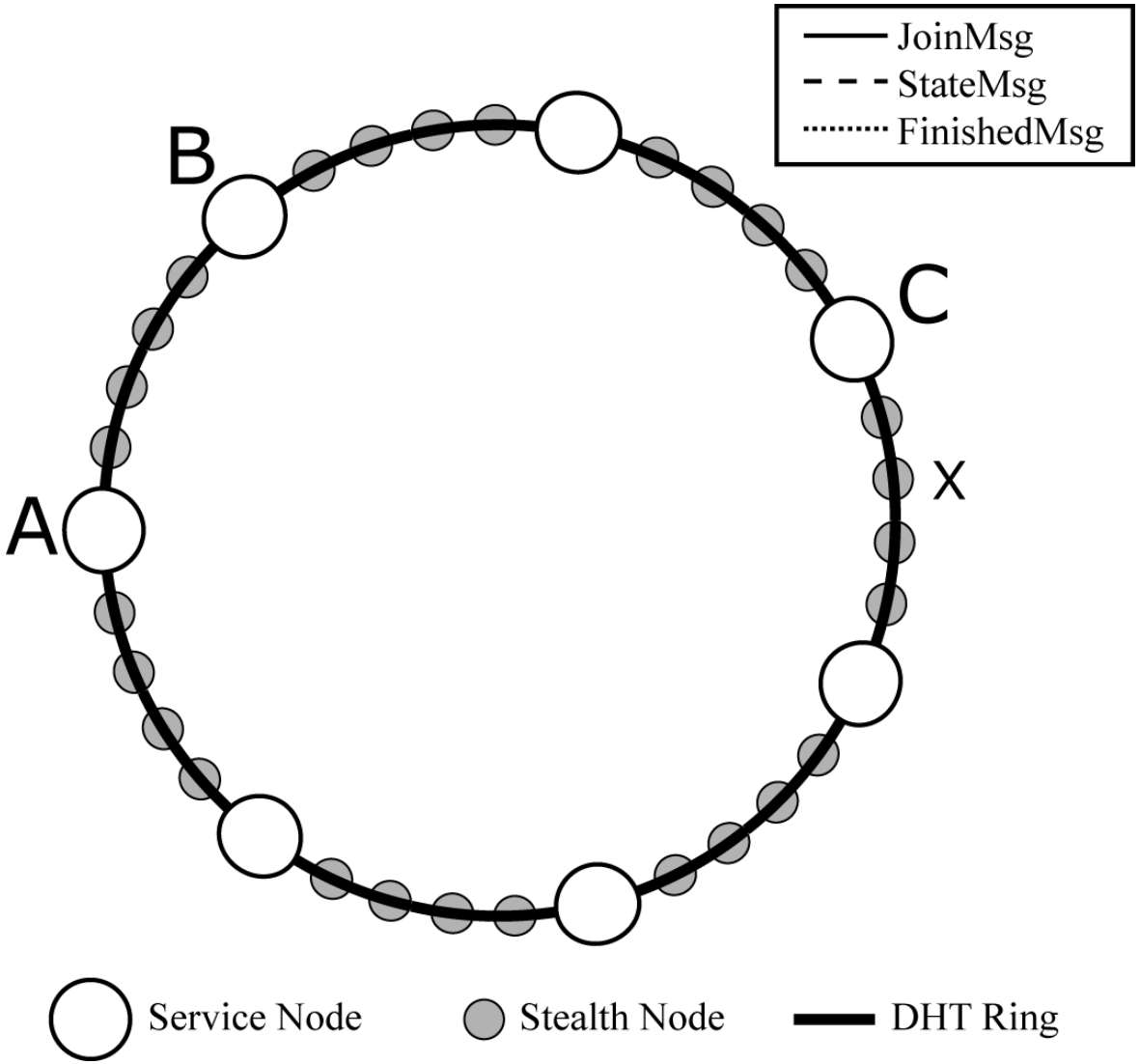
Pastry's Join - State Gathering



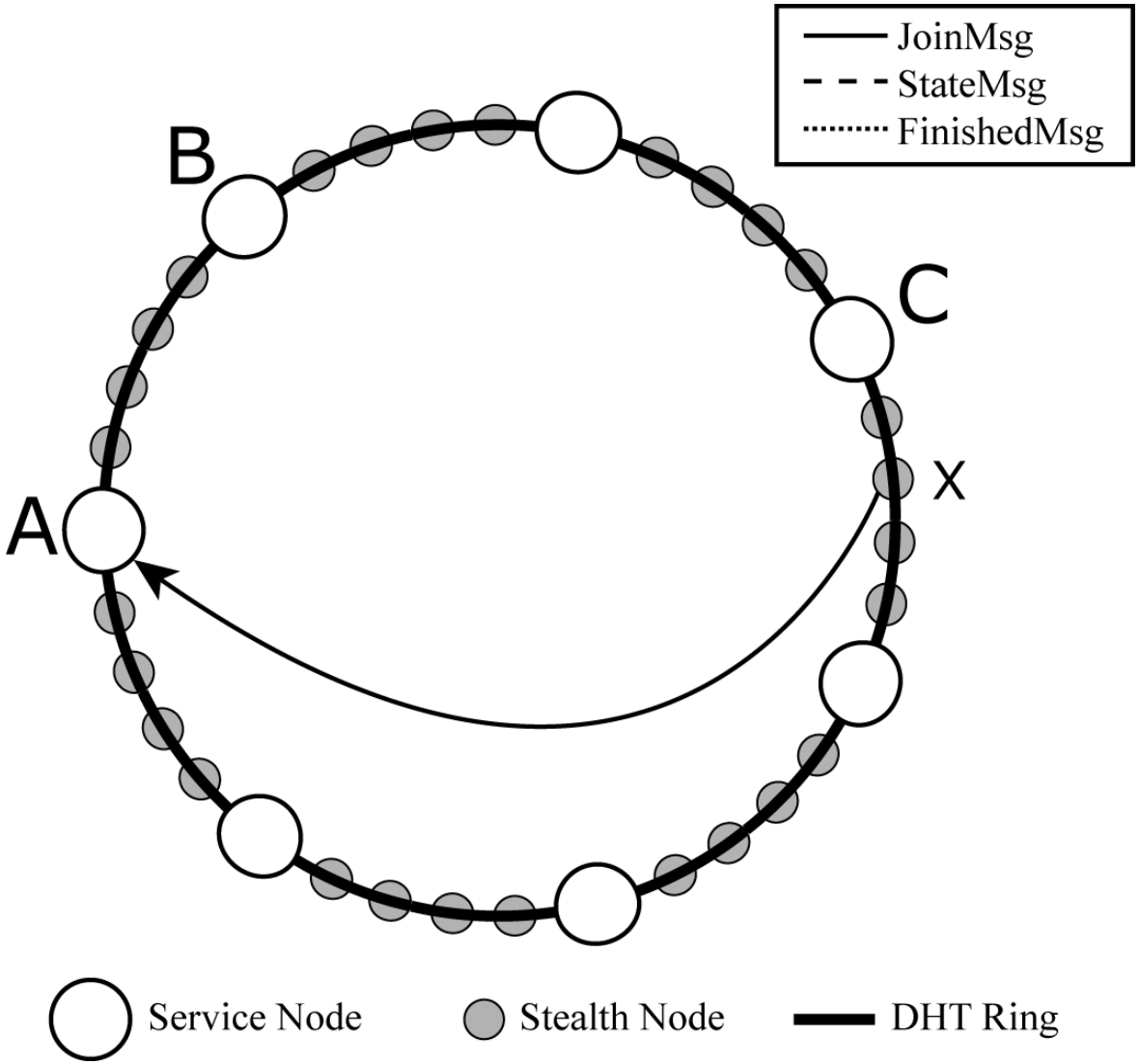
Pastry's Join - Announcement



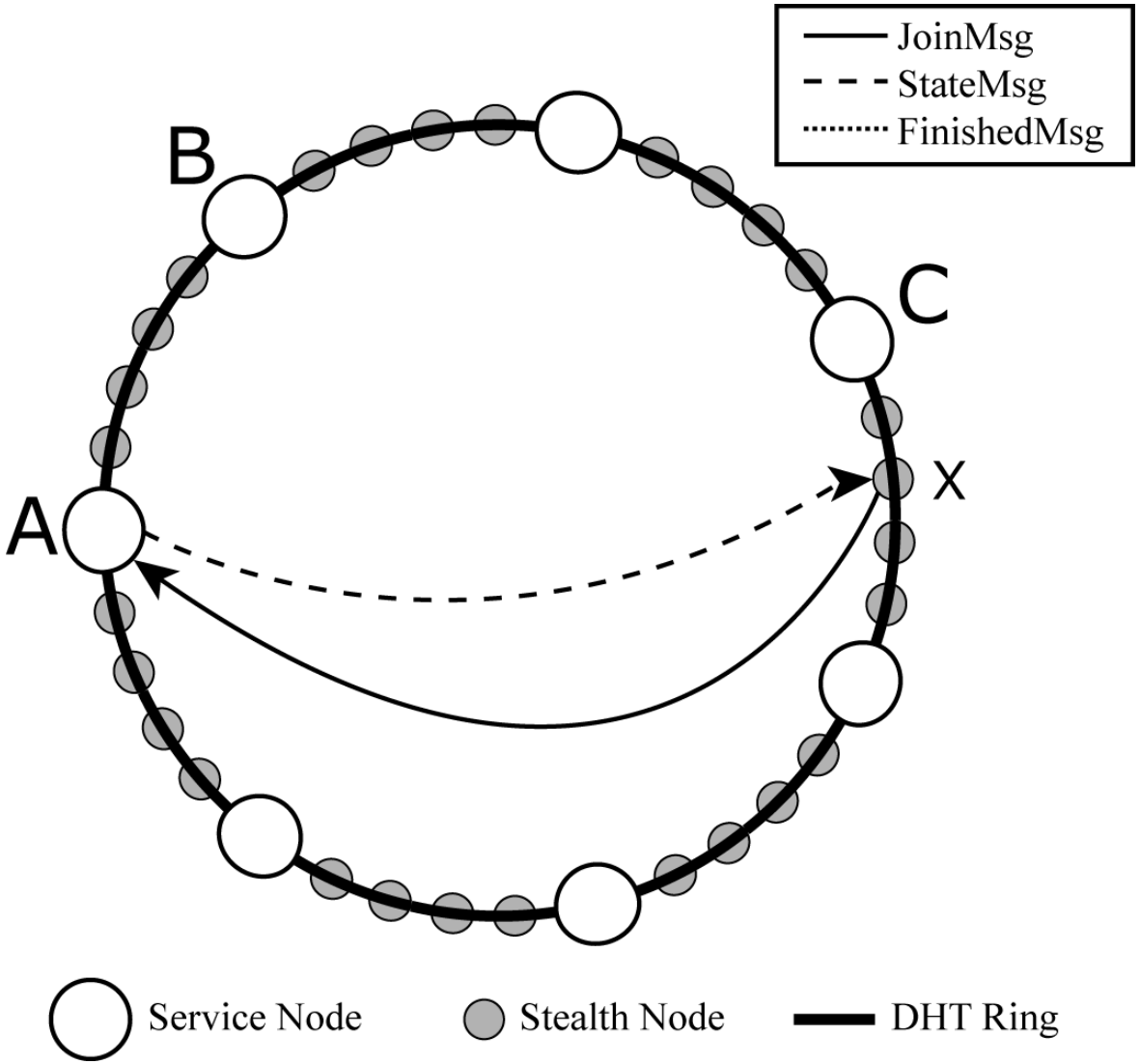
Stealth Node's Join - State Gathering



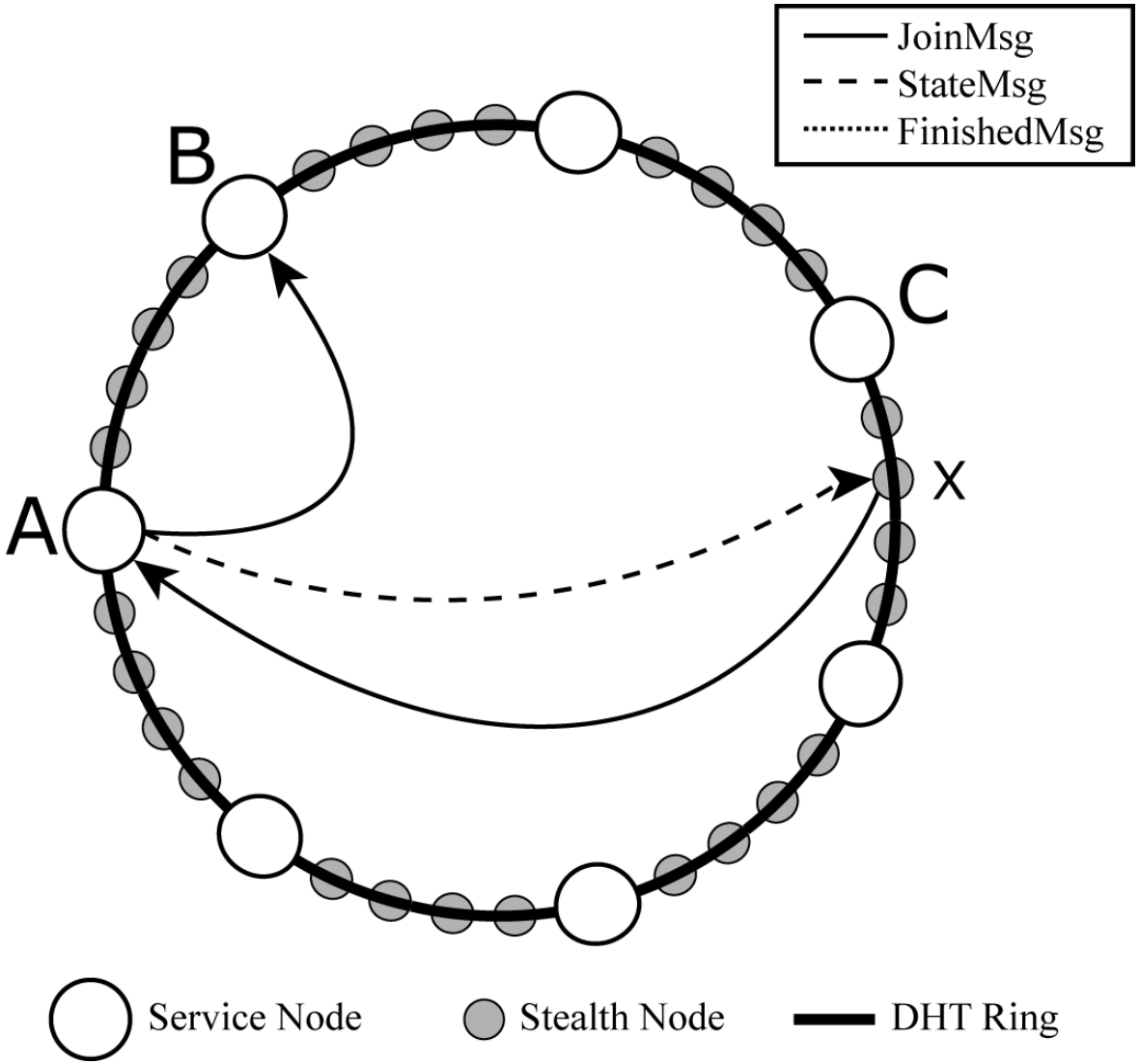
Stealth Node's Join - State Gathering



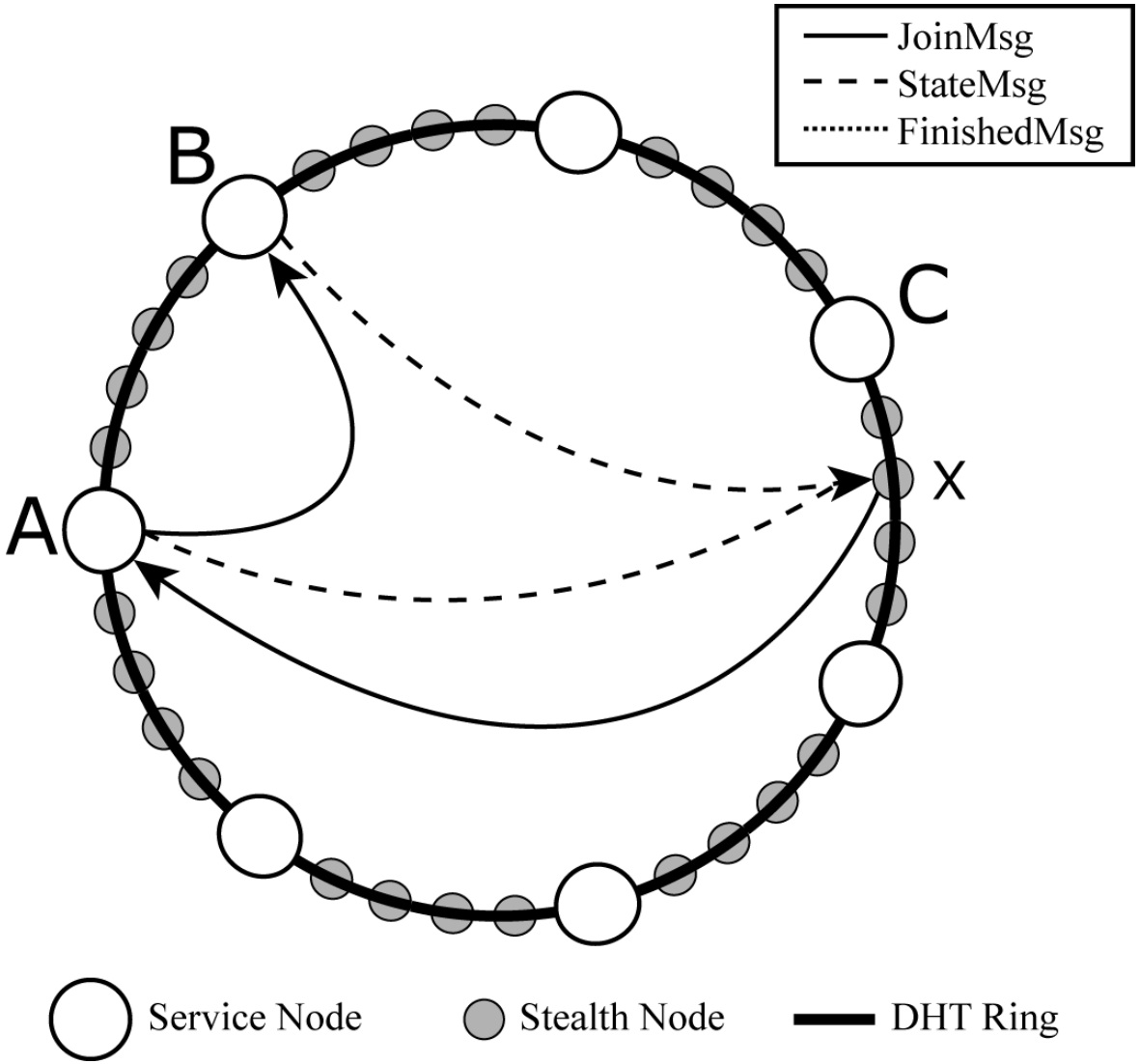
Stealth Node's Join - State Gathering



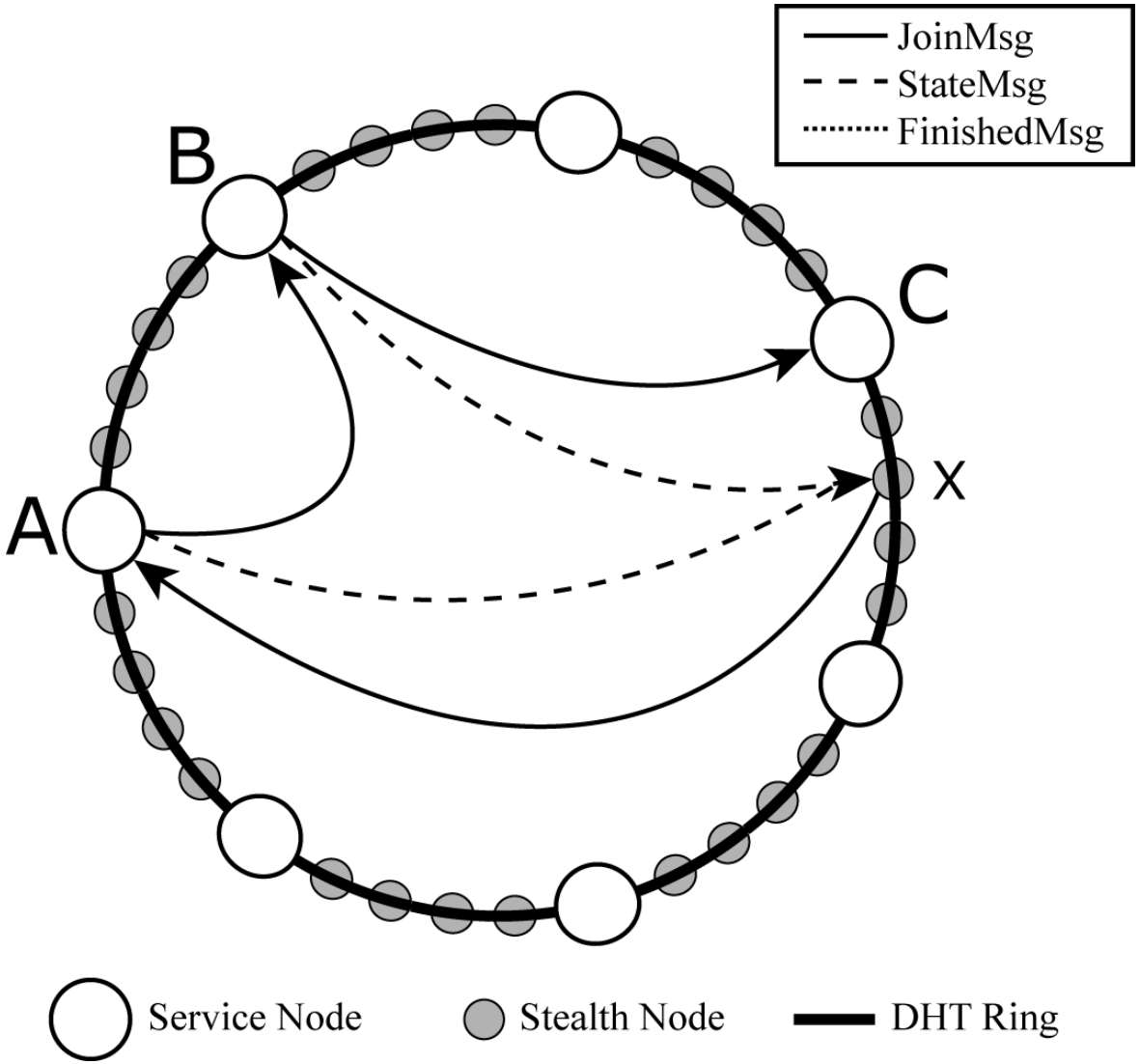
Stealth Node's Join - State Gathering



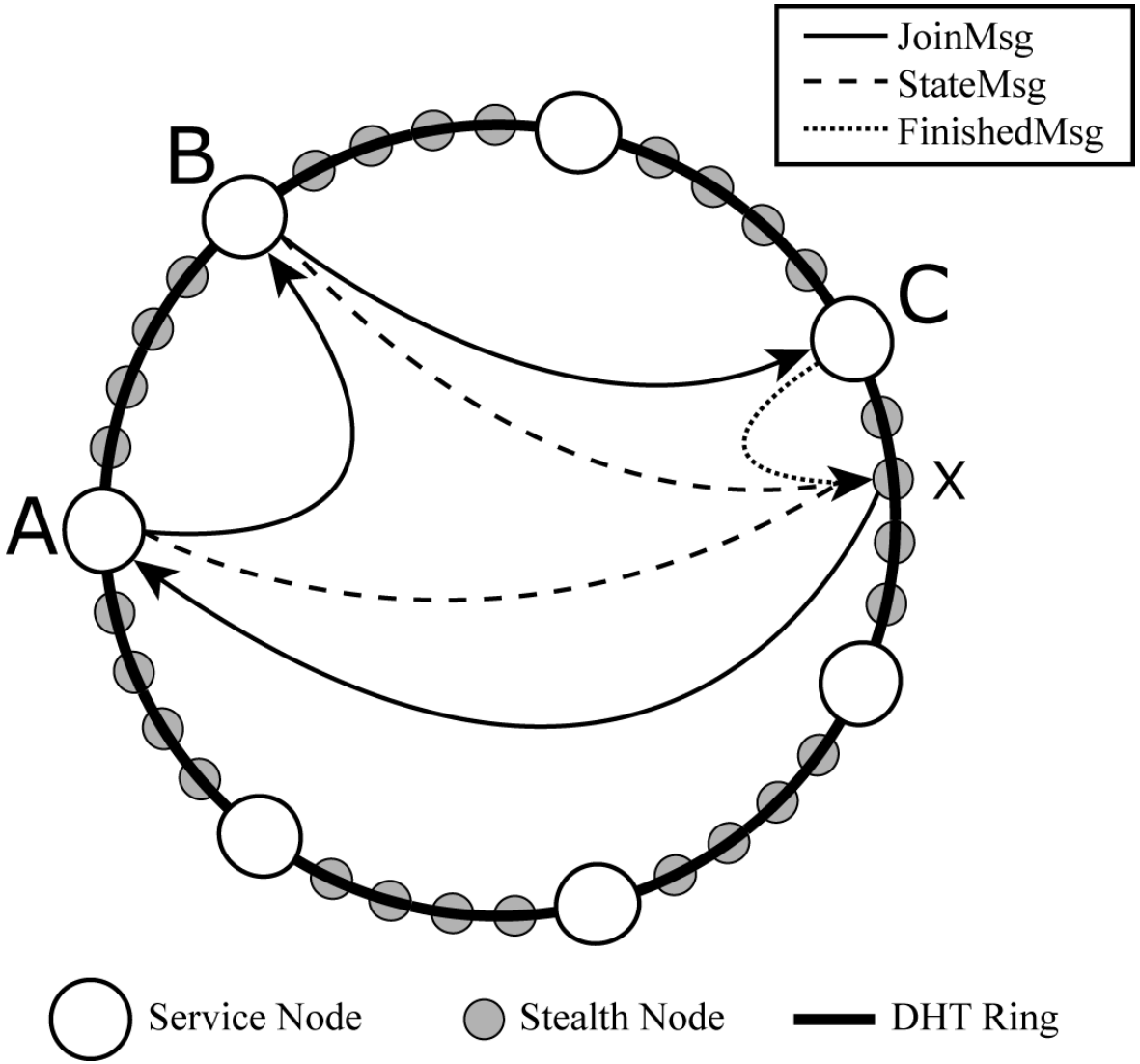
Stealth Node's Join - State Gathering



Stealth Node's Join - State Gathering



Stealth Node's Join - State Gathering



Stealth Node's Join - Announcement

and NO announcement!

How does it work? - Summary

- Service nodes (Wired nodes) join normally
- Stealth nodes (Mobile nodes) join but do not announce

- Therefore
  - Stealth nodes never appear in any node's routing tables
  - Stealth nodes still have complete routing tables, thus resistance and optimal routing (of their own messages)
  - Stealth nodes are not responsible for routing, or storing keys, etc
  - Stealth node's churn affects no one
  - Stealth nodes are effectively invisible

- However
  - Stealth nodes won't receiving routing updates

LANCASTER UNIVERSITY

How does it work? - Summary

However Stealth nodes don't receiving routing updates. (ie knowledge that new service nodes have joined)
Therefore they have an increasingly stale routing table

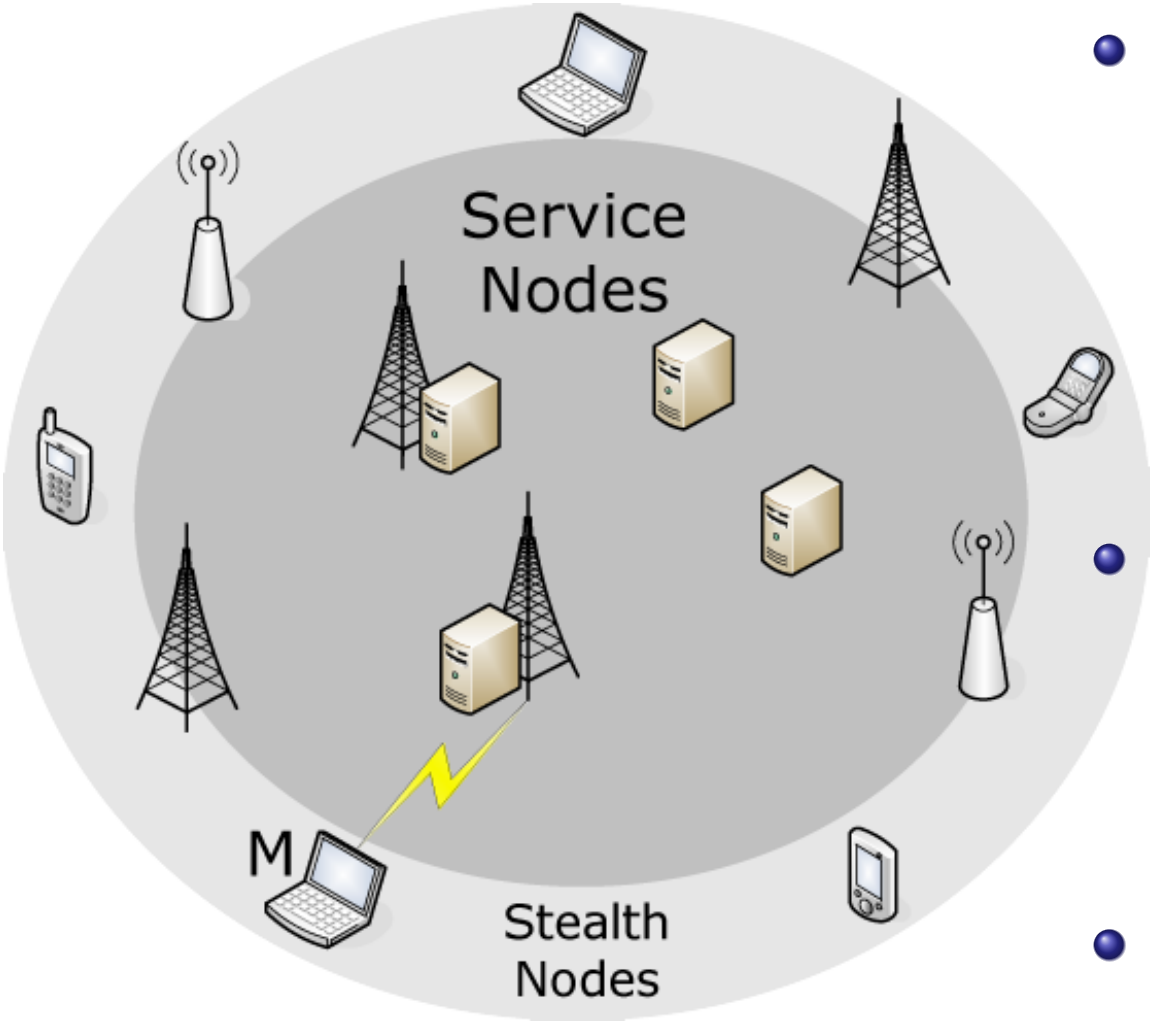
Three solutions to this:

- Polling for updates
- Piggy backing updates
- Periodically rejoining the network

Methodology

- Implementation
  - Wrote a Peer-to-Peer simulator in java
  - Implemented both Pastry and Stealth DHT (based on Pastry)
- Constructed networks of 1-1000 peers
  - 1000 Router transit-stub GT-ITM network (4% transit nodes)
  - Each stub/edge router was a wifi access point
  - Connected Stealth nodes in a random fashion to the APs
  - Connected Service nodes in a random fashion via wired links to the APs
- Simulations (Realistic Scenario)
  - Place 1 million keys in the network
  - Requested keys due to a Zipf distribution $\alpha = 1.2$
  - With and Without Mobility Churn
    - Random Waypoint Model with mean 60min "thinking" times

Methodology

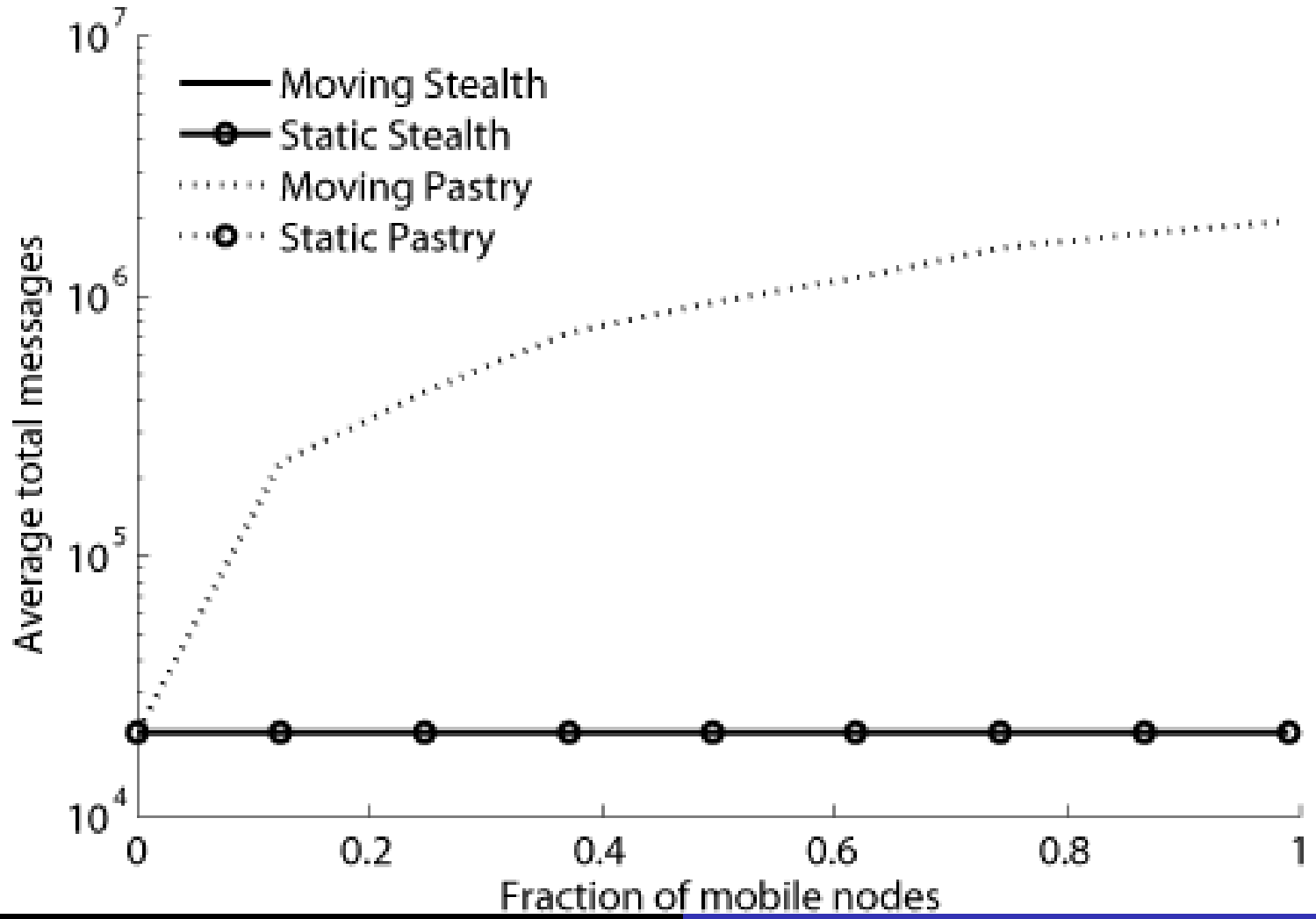


- Service Nodes
  - PCs (workstation/servers etc)
  - Connected via a wired Network
- Stealth Nodes
  - Mobile devices
  - Connected via the wifi Access Points
- Service/Stealth all in the same DHT

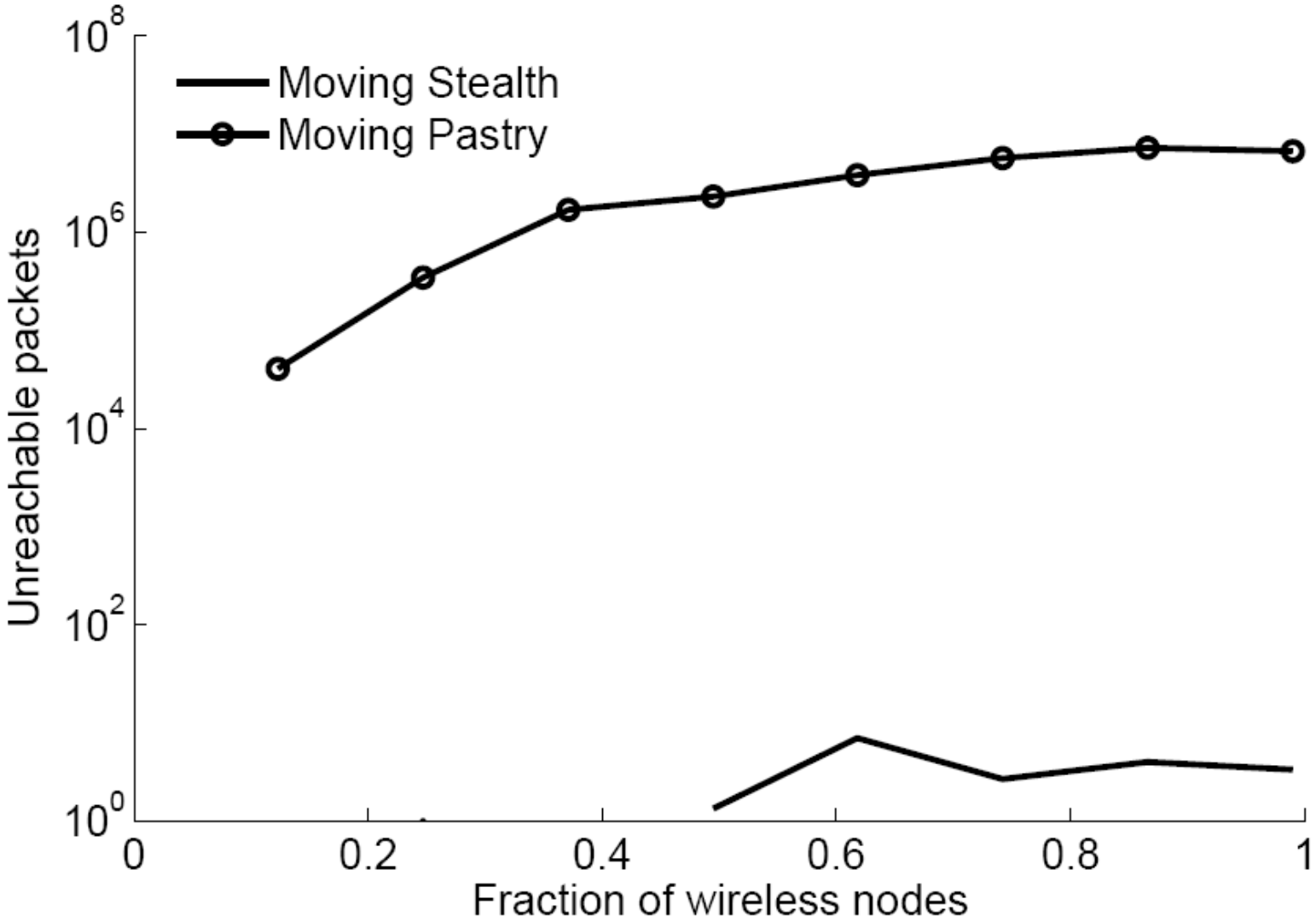
Results - Introduction

- All plots show 1000 Peer networks
  - 1% Service nodes
  - 99% Stealth nodes
- Plots on the x-axis show fractions of Stealth nodes who were wireless vs wired
- Moving {Stealth, Pastry} refers to simulations where wireless nodes moved from AP point to AP. (A new IP is obtained)
- Static {Stealth, Pastry} refer to simulations where nodes did not move

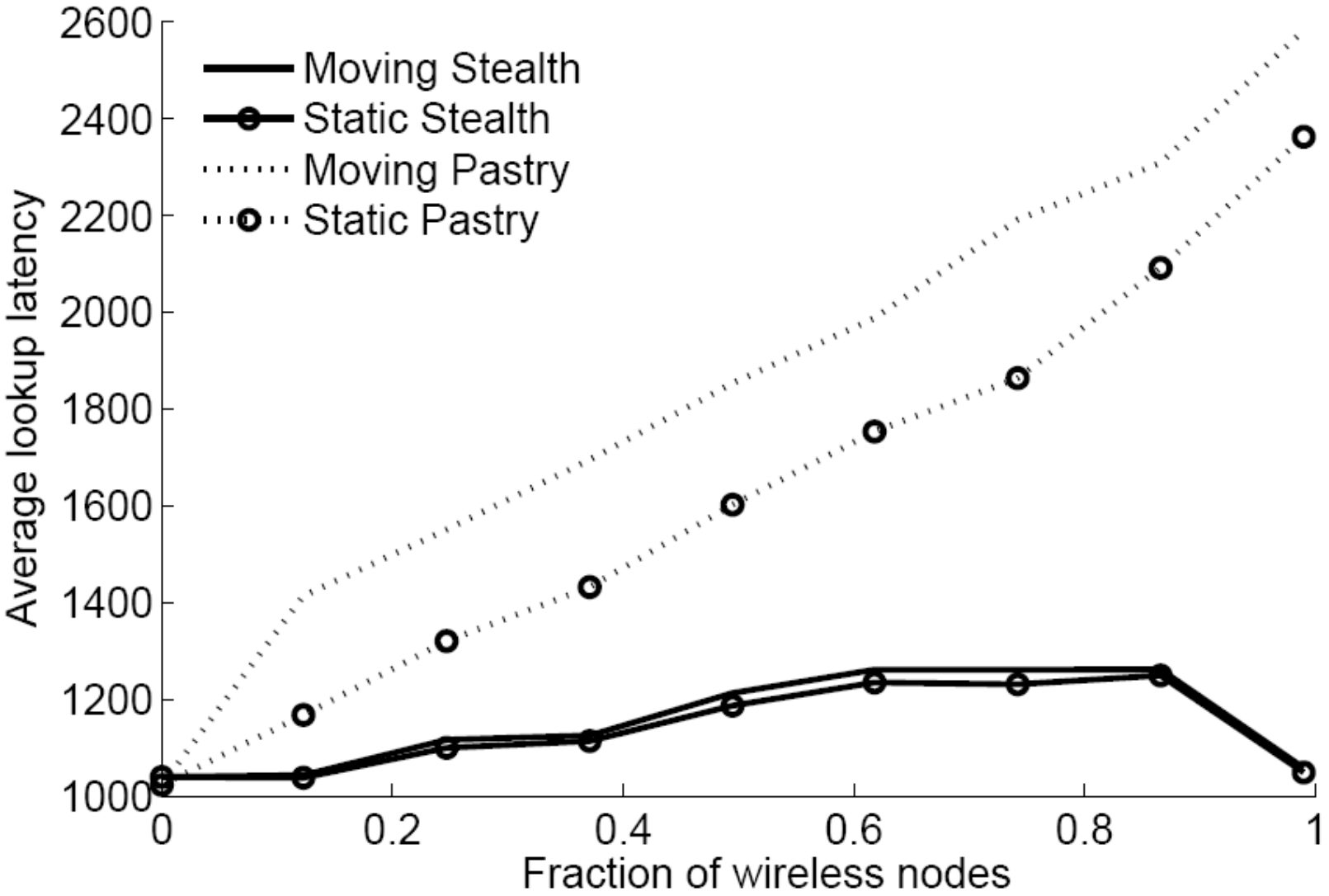
Total number of messages



Failed packets due to nodes being unreachable



Average lookup latency



Summary

Stealth DHT

- Partitions the network into two groups
- Increases DHT performance in most areas
- Returns control to the service operator
- Suitable for networks with mobile peers

Outlook

- Investigate possible applications to run on top of a Stealth DHT
  - Content Distribution Networks
  - Novel Peer-to-Peer Applications
- Automatically decide who is Stealth/Service node, and change them on the fly

LANCASTER UNIVERSITY

Thank you for listening

Questions?

Performance Analysis of Stealth Distributed Hash Table with Mobile Nodes

Andy MacQuire, **Andrew Brampton**, Idris A. Rai and Laurent Mathy
Computing Department
Lancaster University, UK

Friday 18th March / Mobile Peer-to-Peer 2006

LANCASTER UNIVERSITY

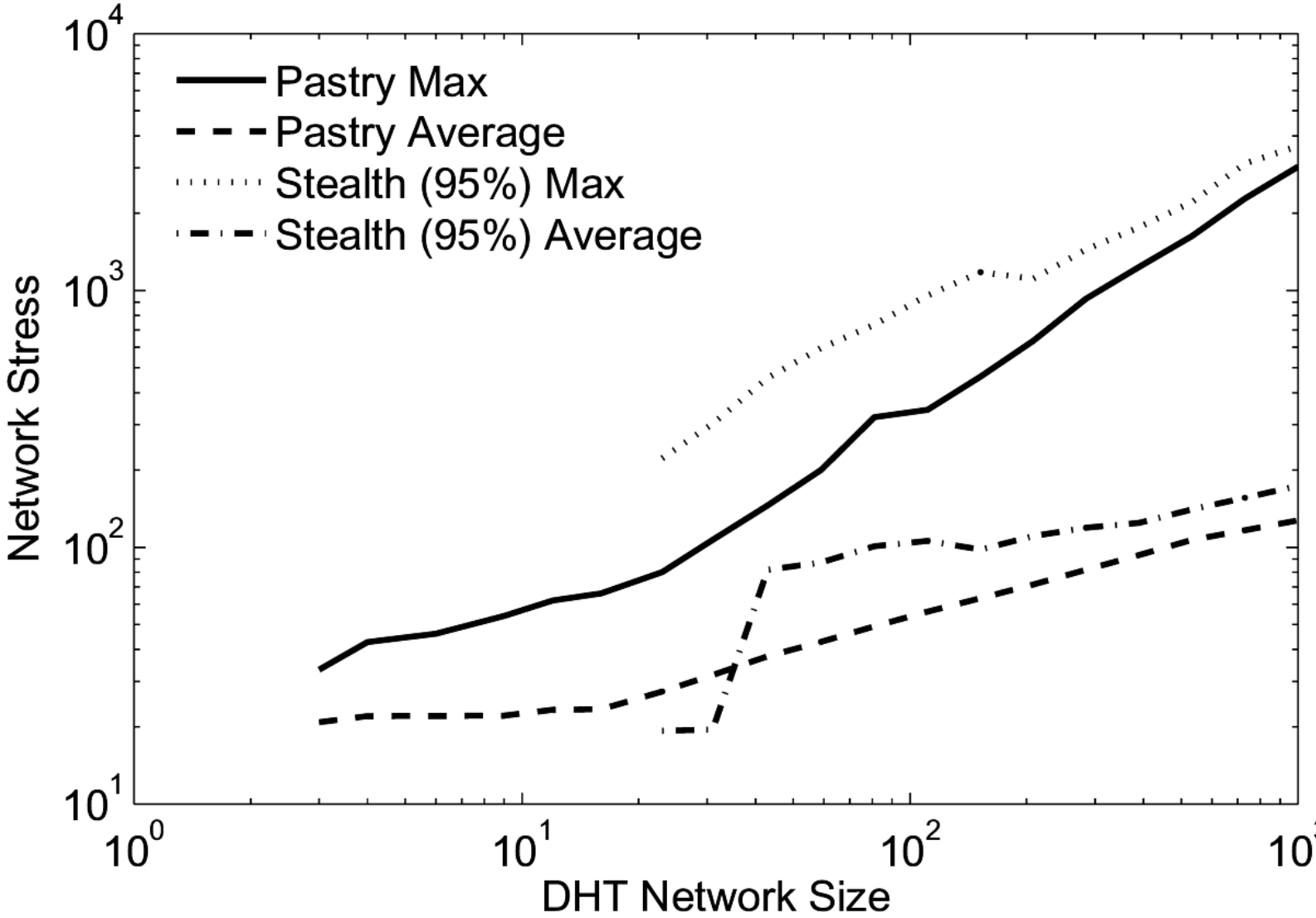
Current state of work

Stealth DHT

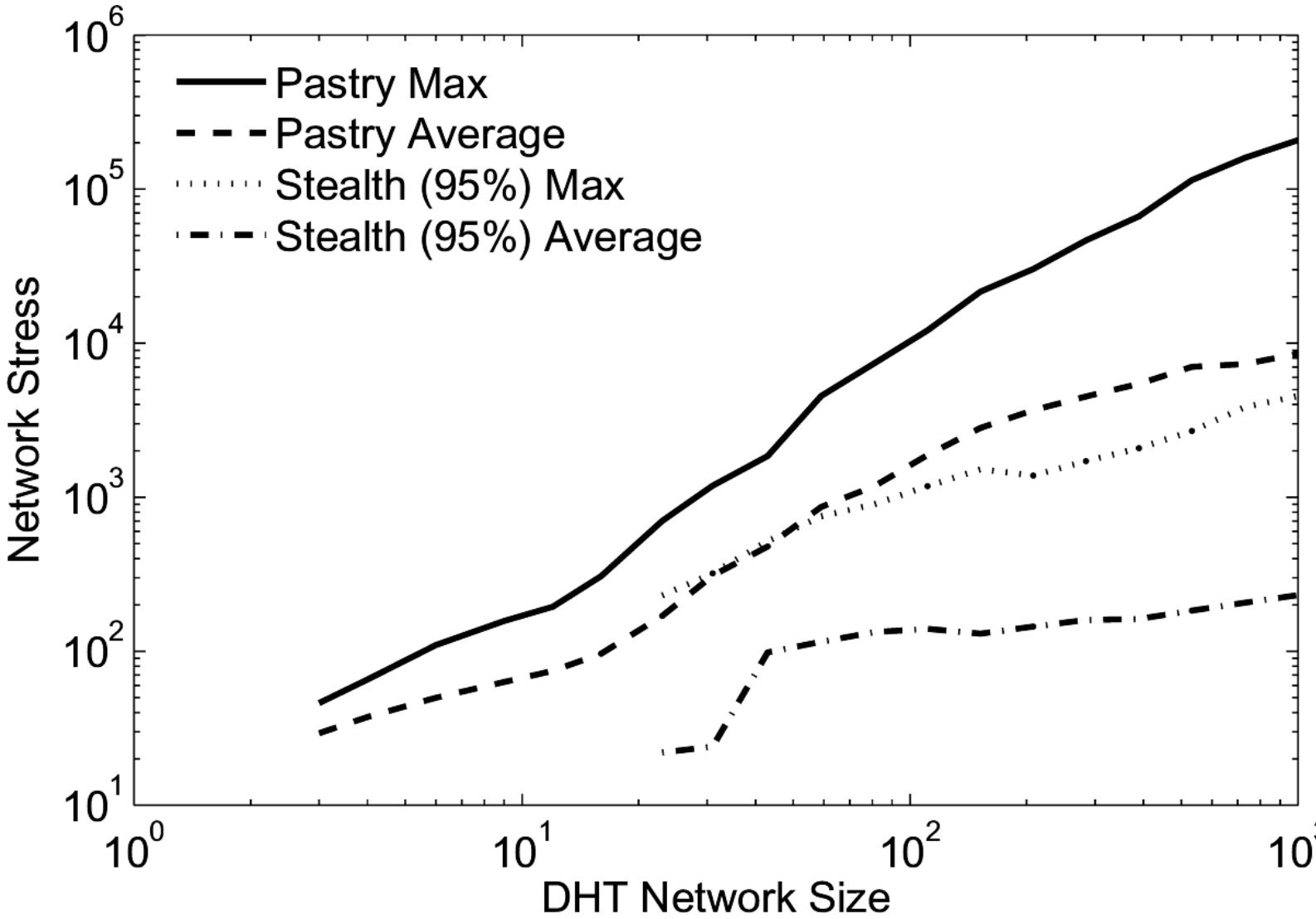
- We have a C++ implementation running on planetlab
- We have an Authentication model to enforce roles (and improve security)

LANCASTER UNIVERSITY

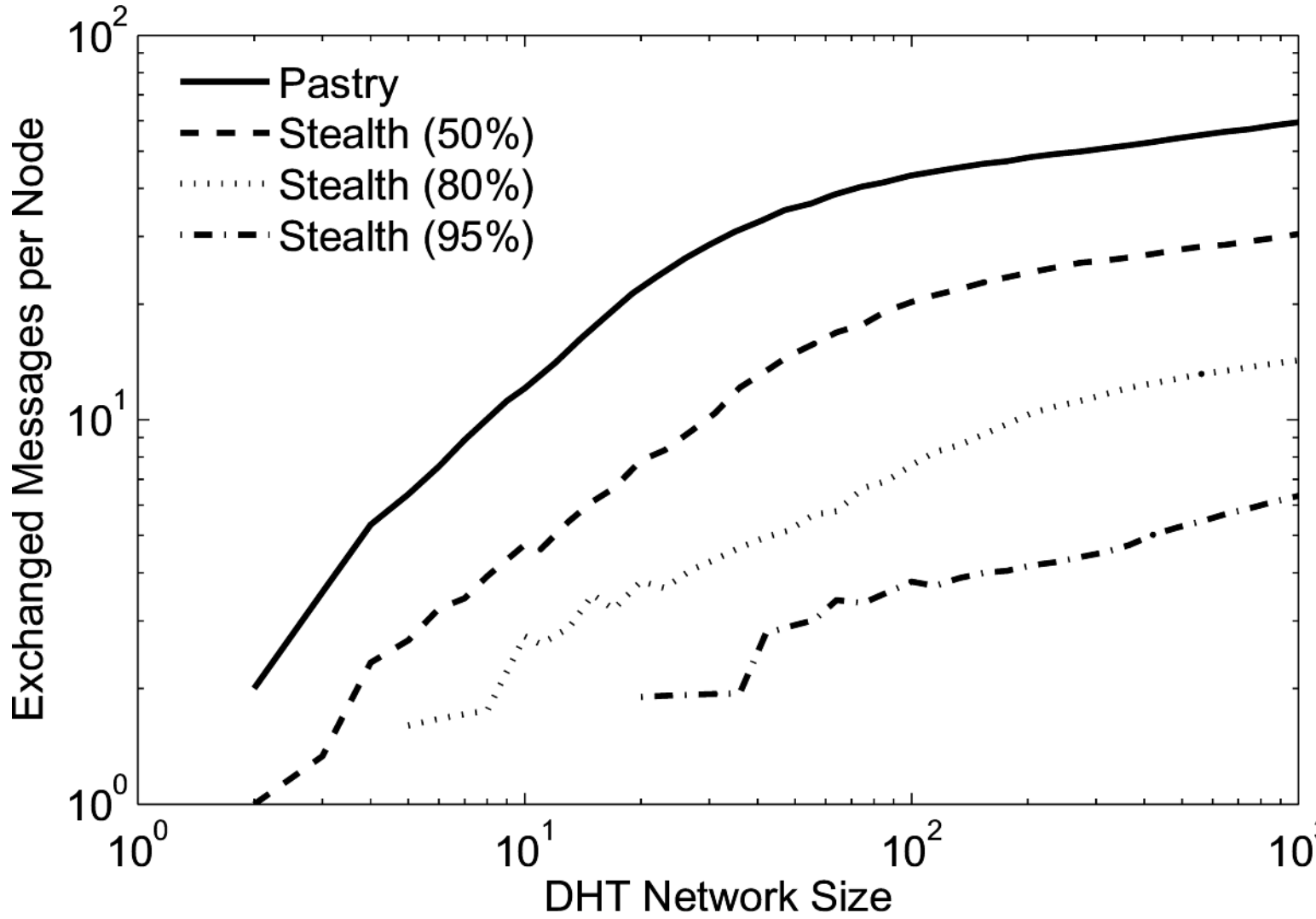
Network Stress vs DHT size



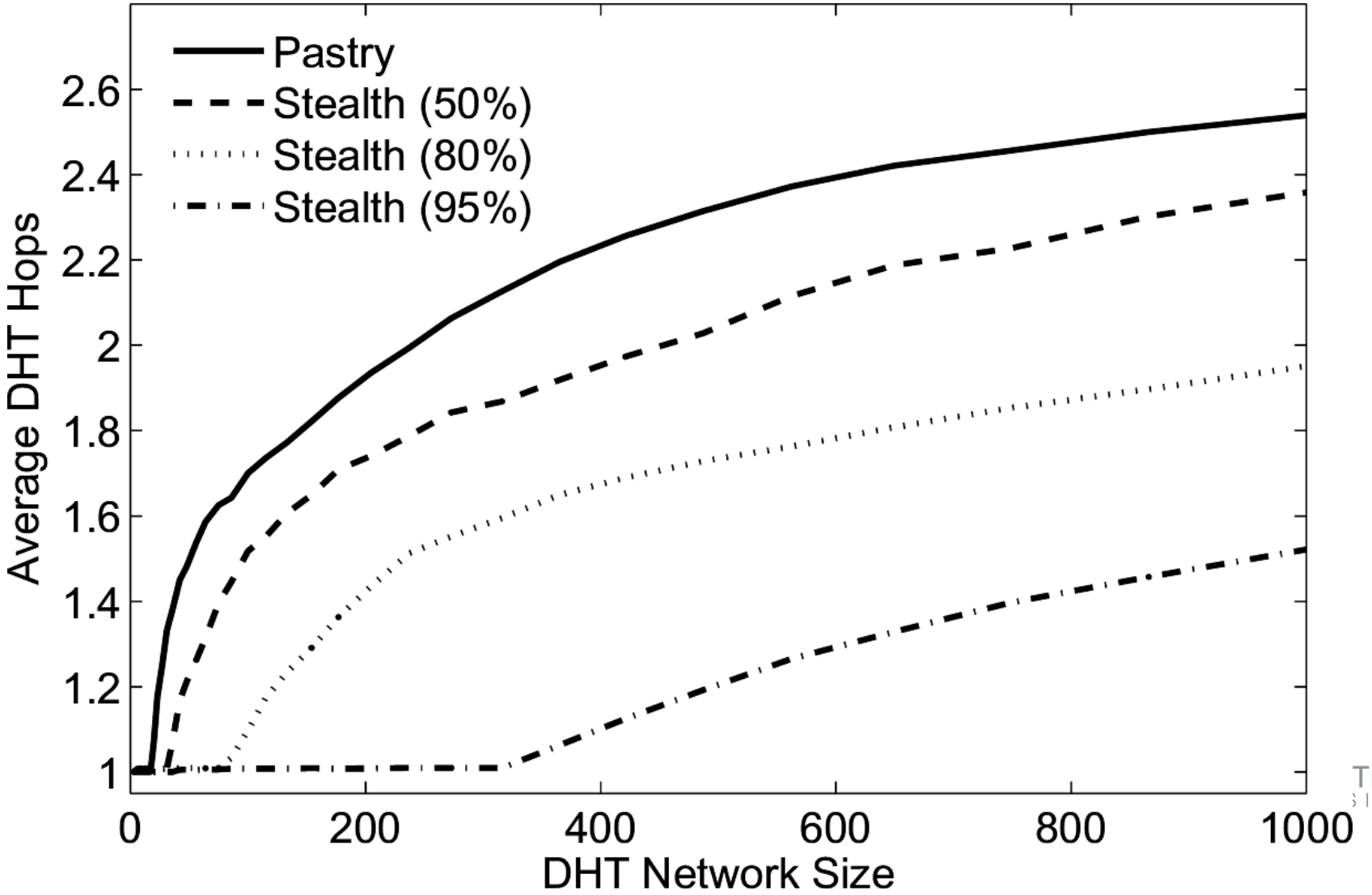
Network Stress vs DHT size (with Churn)



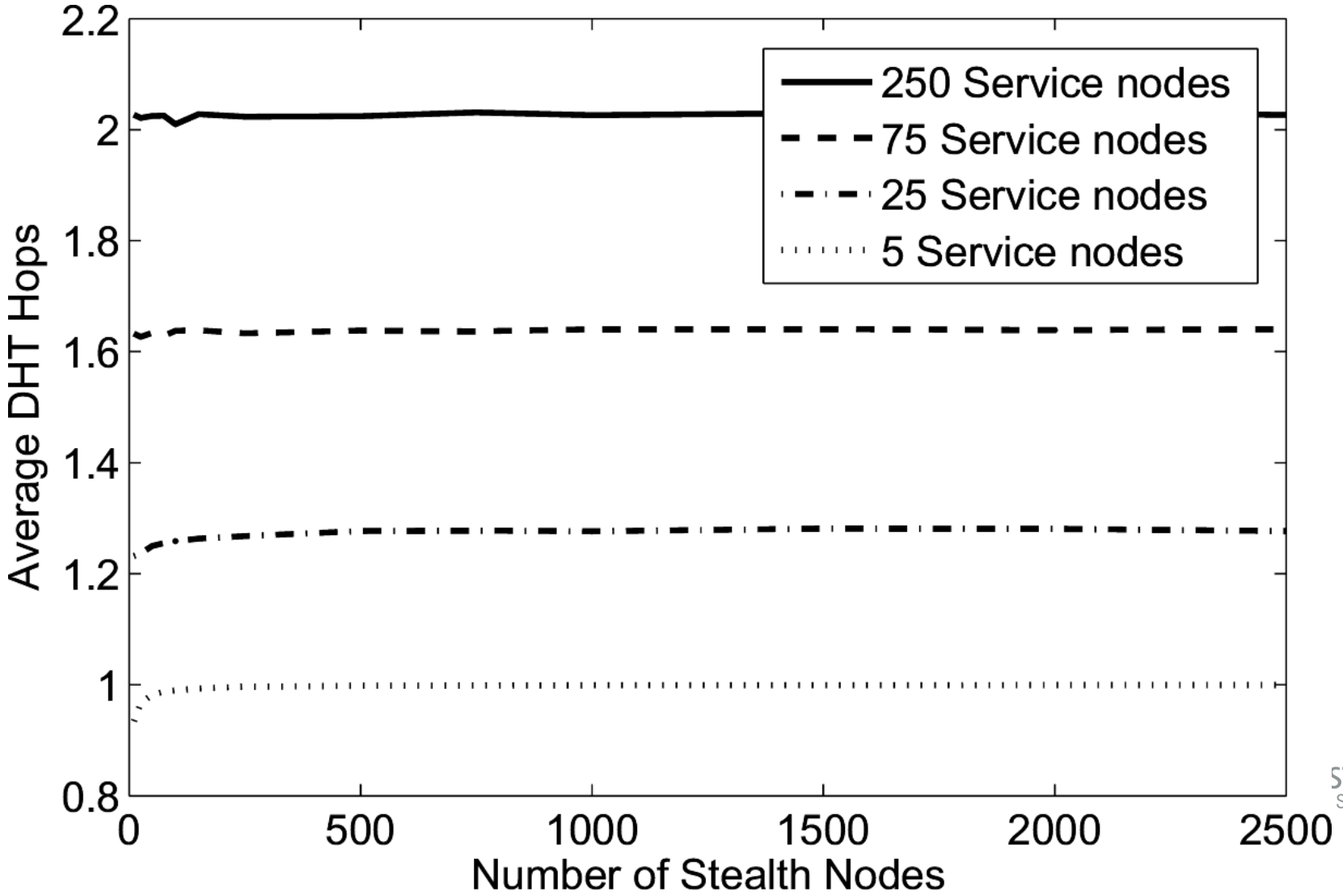
Average messages per node during join vs DHT size



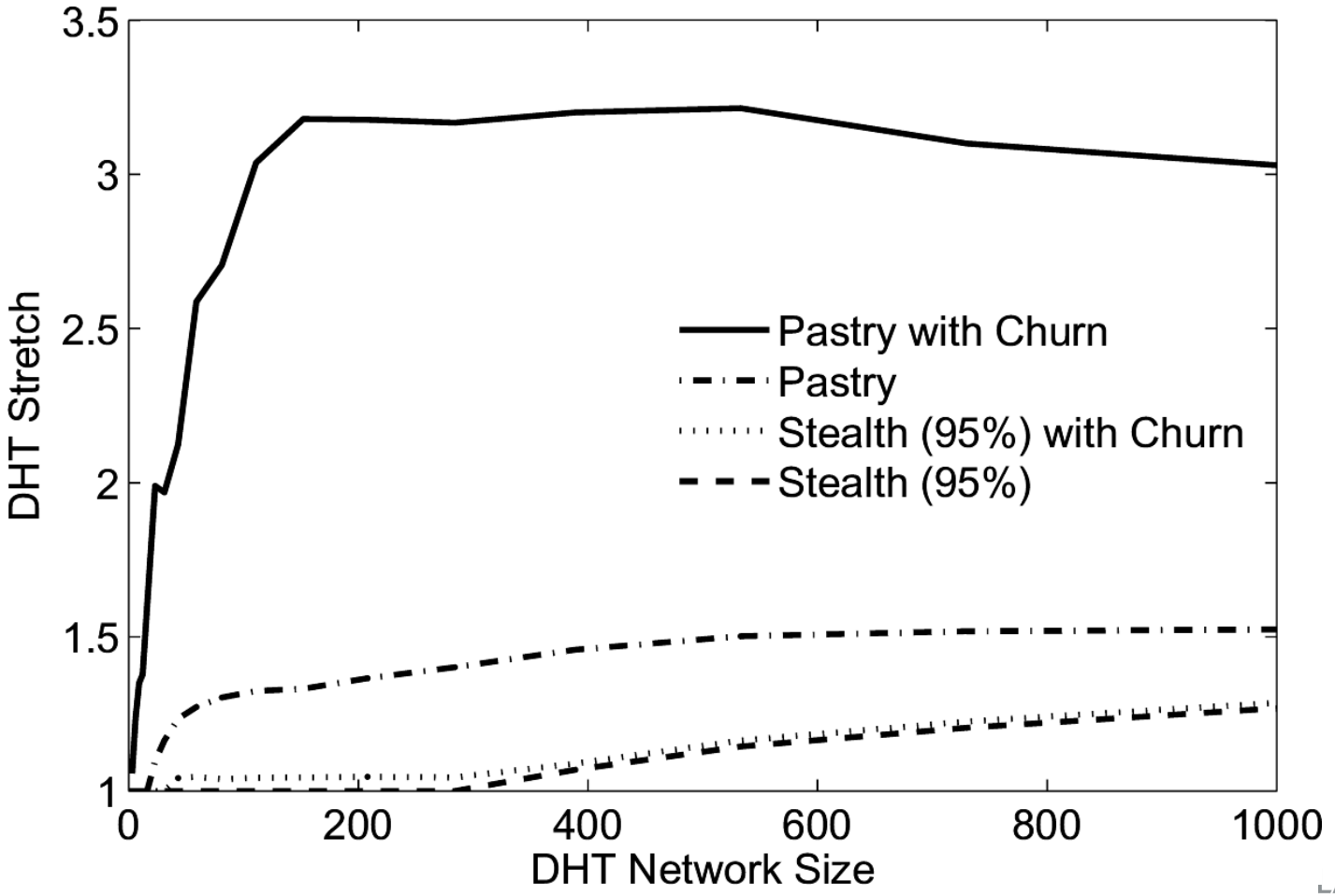
Average DHT hops vs DHT size



Average DHT hops vs # of Stealth nodes

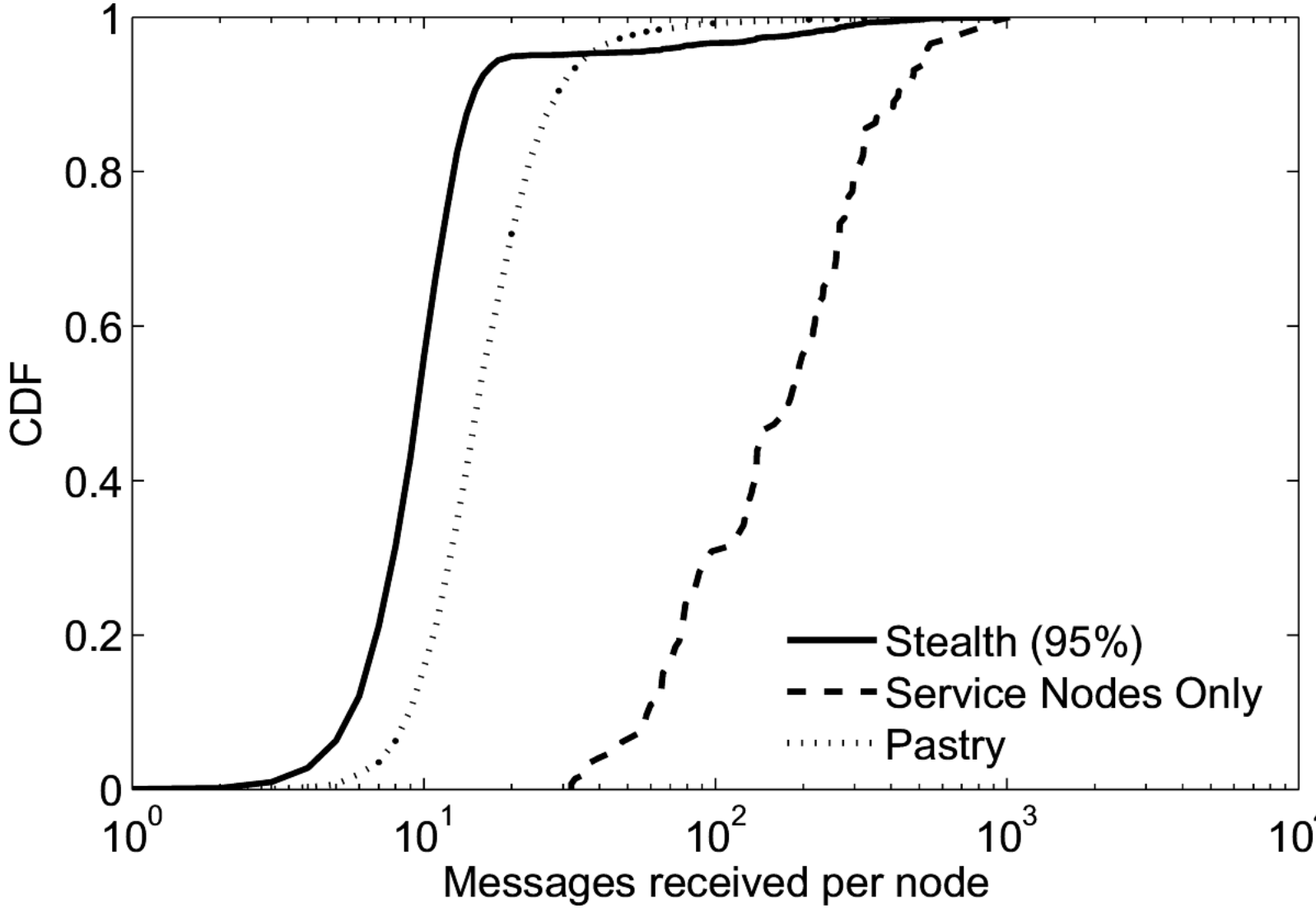


Stretch vs DHT size



Stretch = DHT End-to-end delay / Unicast End-to-end delay

Recv'ed messages per node vs DHT size



Recv'ed messages per node vs DHT size (with Churn)