Kademlia on the Open Internet

How to Achieve Sub-Second Lookups in a Multimillion-Node DHT Overlay

Raul Jimenez

ROYAL INSTITUTE
OF TECHNOLOGY

Licentiate Thesis in Communication Systems

Outline

- Context
- Introduction
- Background
- Main results
- Conclusion

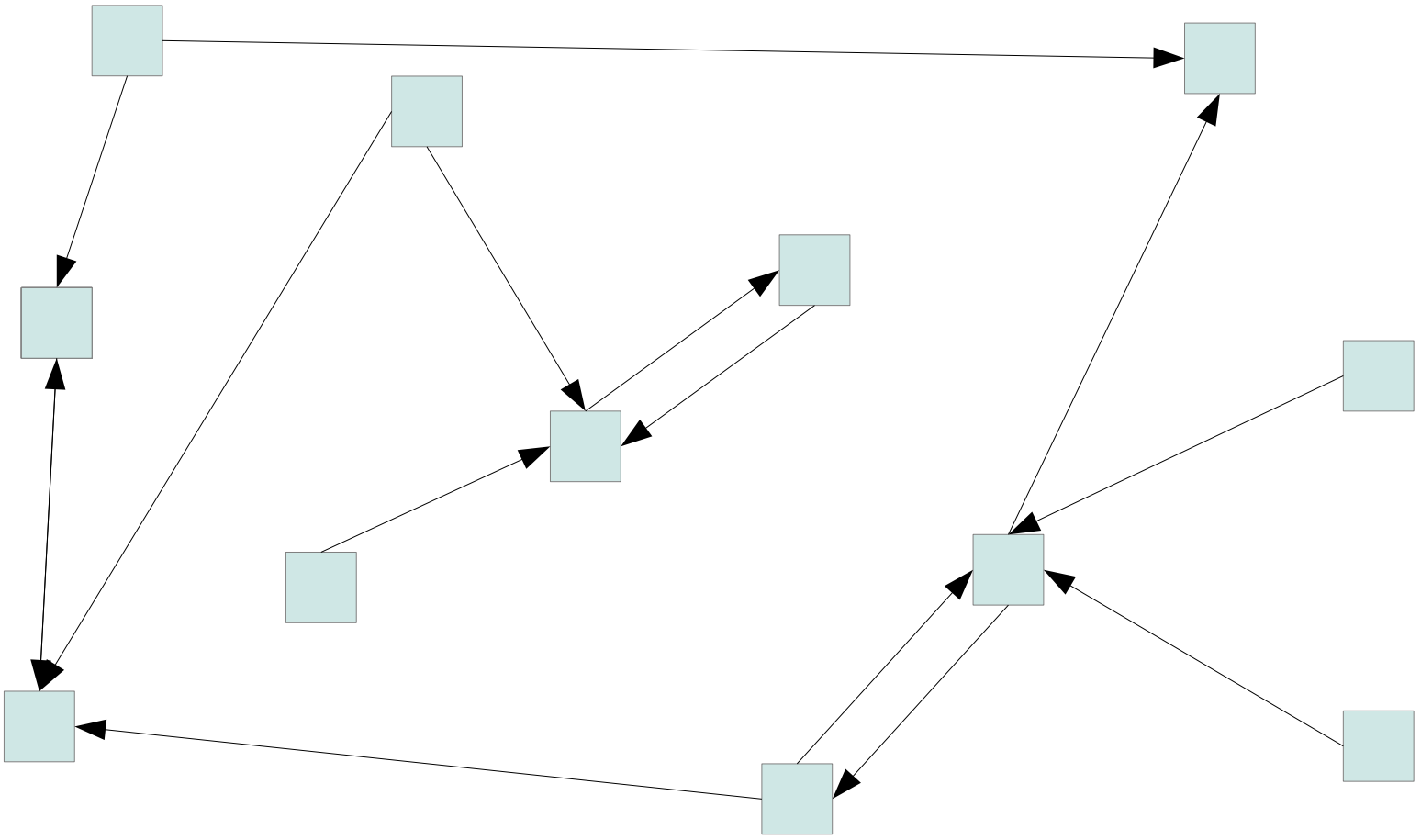
<context>


P2P
NEXT


AG Projects
IRT
RTV SLO MMC
LANCASTER UNIVERSITY
BBC
SAPIENZA UNIVERSITÀ DI ROMA
DACC SYSTEMS
kendra initiative
oversi
content. storage. delivery.
EBU·UER
Universitatea Politehnica Bucureşti
TU Delft
KTH
Pioneer
ALPEN-ADRIA UNIVERSITÄT KLAGENFURT
VTT

- Content distribution platform
- Fully decentralized
- Streaming support
- Web integration

P2P content distribution



File Edit View History Bookmarks Tools Help

File:Play fight of polar b...

en.wikipedia.org/wiki/File:Play_fight_of_polar_bears_edit_1.avi.OGG

Google

Log in / create account

WIKIPEDIA
The Free Encyclopedia

File Discussion | Read Edit View history | Search

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia

Toolbox

File:Play fight of polar bears edit 1.avi.OGG

From Wikipedia, the free encyclopedia

File | File history | File usage | Global file usage

More...

No higher resolution available.

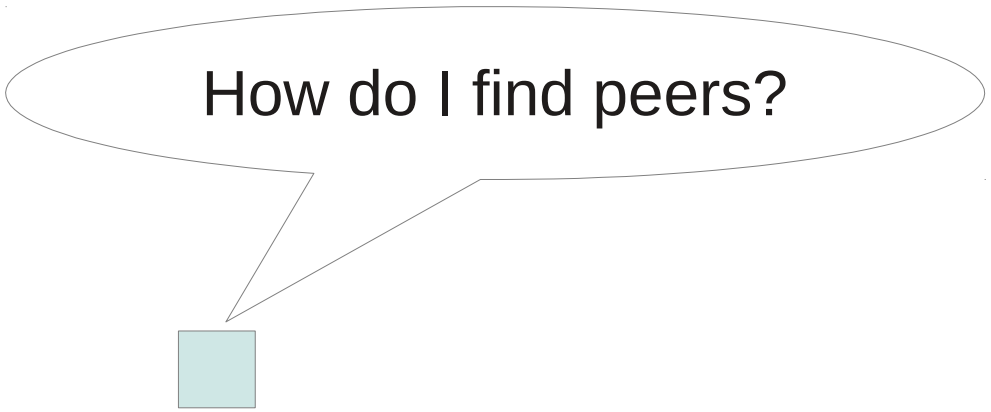
http://swarmplayer.p2p-next.org/

0:0

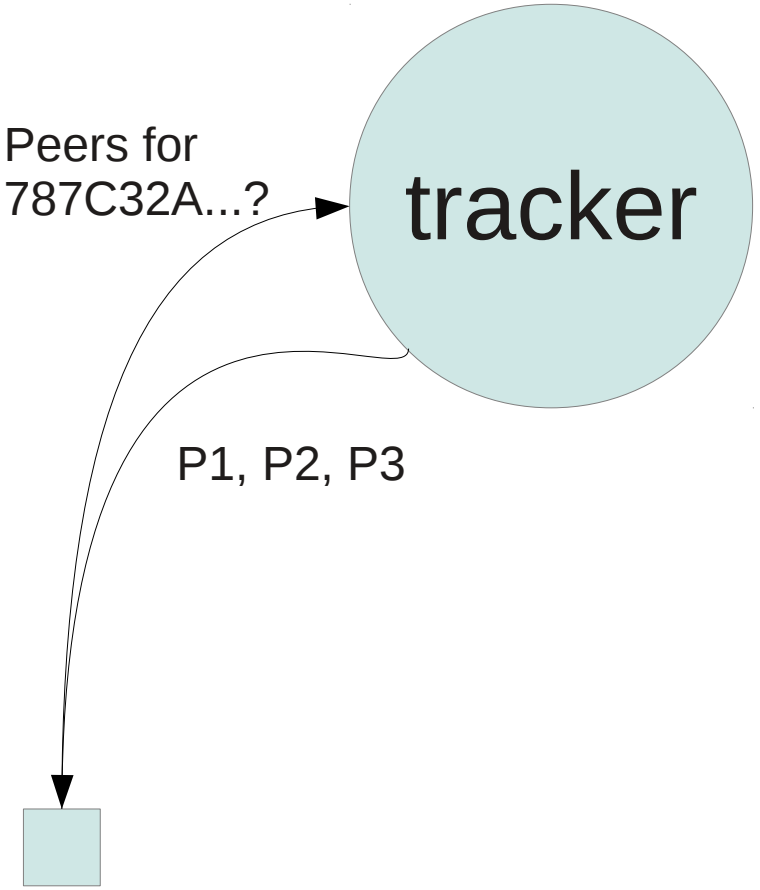
zotero

</context>

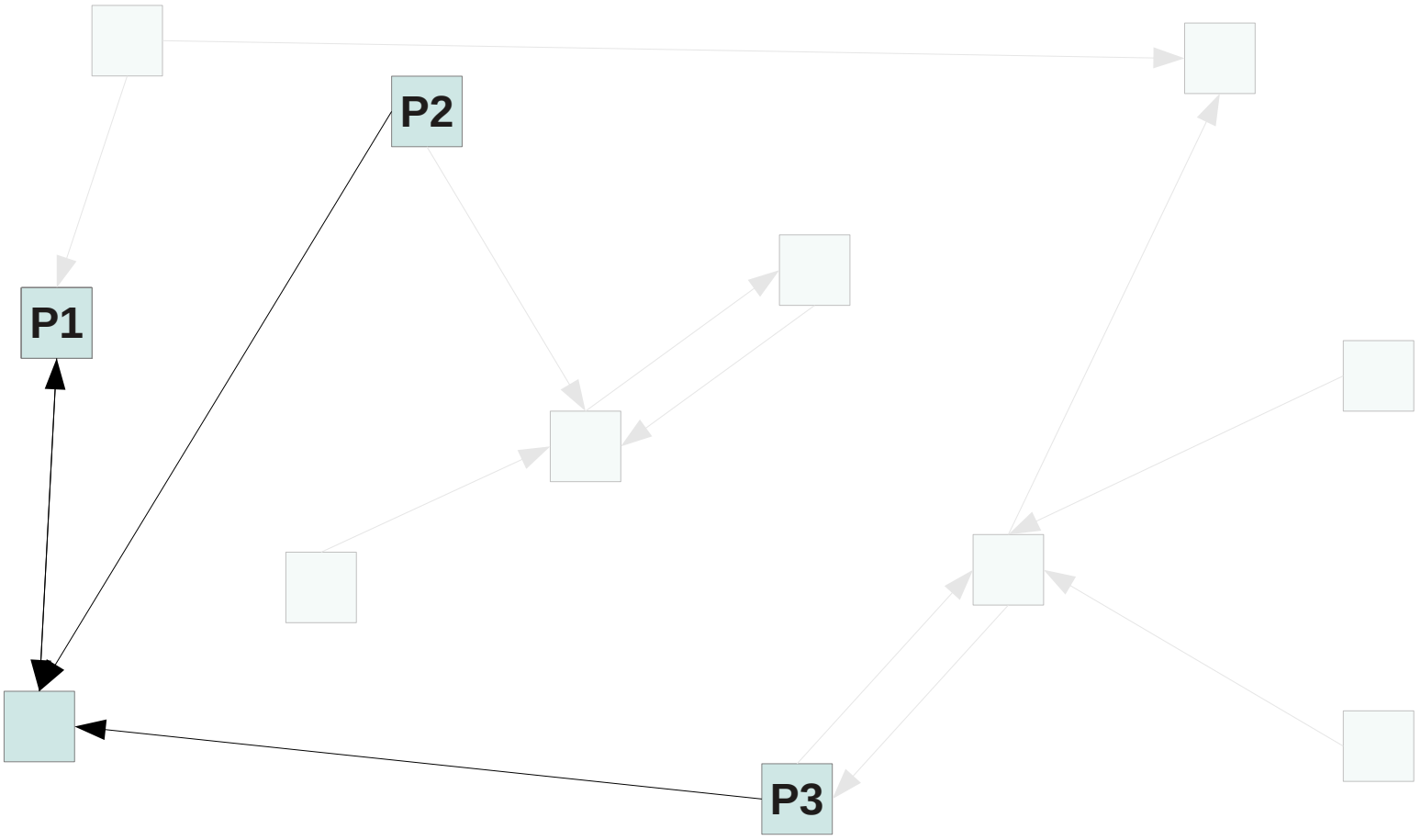
<introduction>


How do I find peers?

Peer discovery (aka *tracker*)



Peer-to-peer data transfer



Tracker

- Server
  - Centralized
  - Simplicity
  - Statistics
  - Scalability
  - Single point of failure

Tracker

- Server
  - Centralized
  - Simplicity
  - Statistics
  - Scalability
  - Single point of failure
- Distributed hash table
  - Decentralized
  - Scalability
  - No single point of failure
  - Churn
  - **High latency**

Tracker

- Distributed hash table
  - Decentralized
  - Scalability
  - No single point of failure
  - Churn
  - **High latency**

</introduction>

<background>

Distributed Hash Table (DHT)

Distributed Hash Table (DHT)

- Key, value store/retrieve service

KEY	→	**VALUE**
Tel nr.	→	Name
Person ID	→	contact info
Product ID	→	stock info
Content ID	→	**list of peers**

Distributed Hash Table (DHT)

- Key, value store/retrieve service
- Same as “normal” hash tables but:
  - Values are stored on nodes
  - Nodes are connected to other nodes, forming an overlay
  - Lookup mechanism to locate the nodes responsible for a given key
- DHT designs:
  - Chord, Tapestry, CAN, Pastry, **Kademlia**, ...

Kademlia

Kademlia

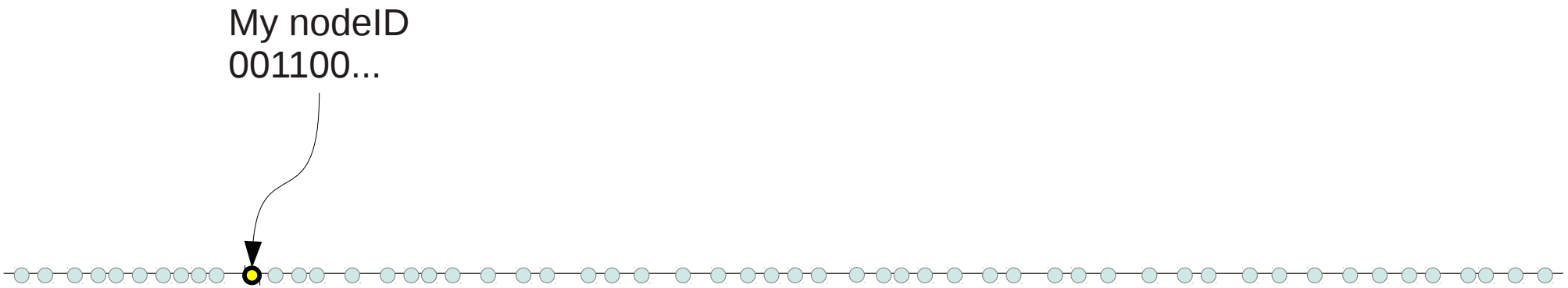
- XOR metric
- Iterative lookup

- Kademlia-based overlays on the Internet
  - Mainline DHT
  - KAD
  - Azureus DHT

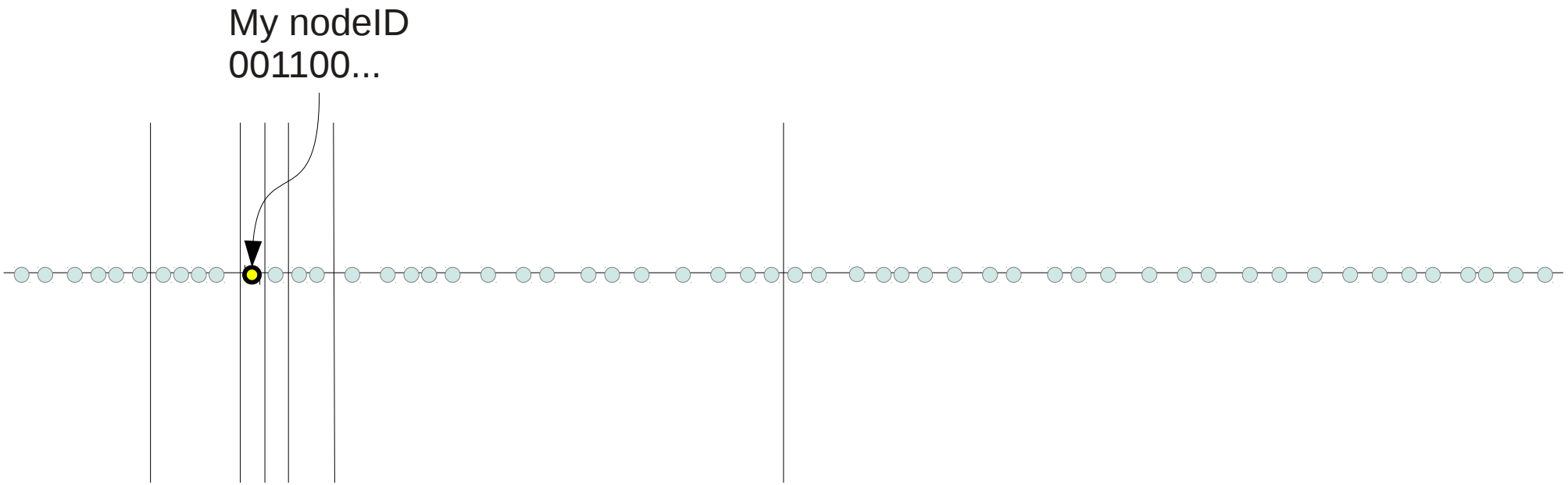
} > 1 million nodes each!

Kademlia's routing tables

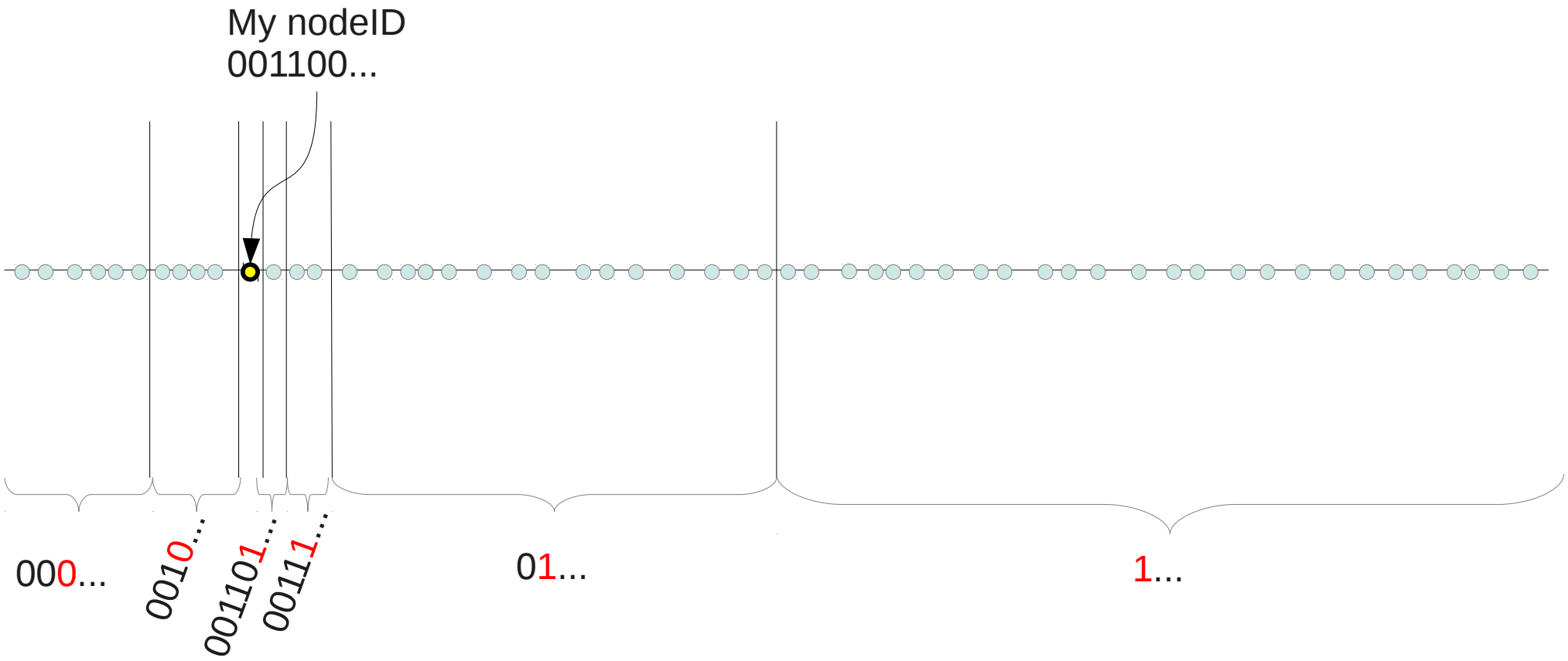
Kademlia's routing tables



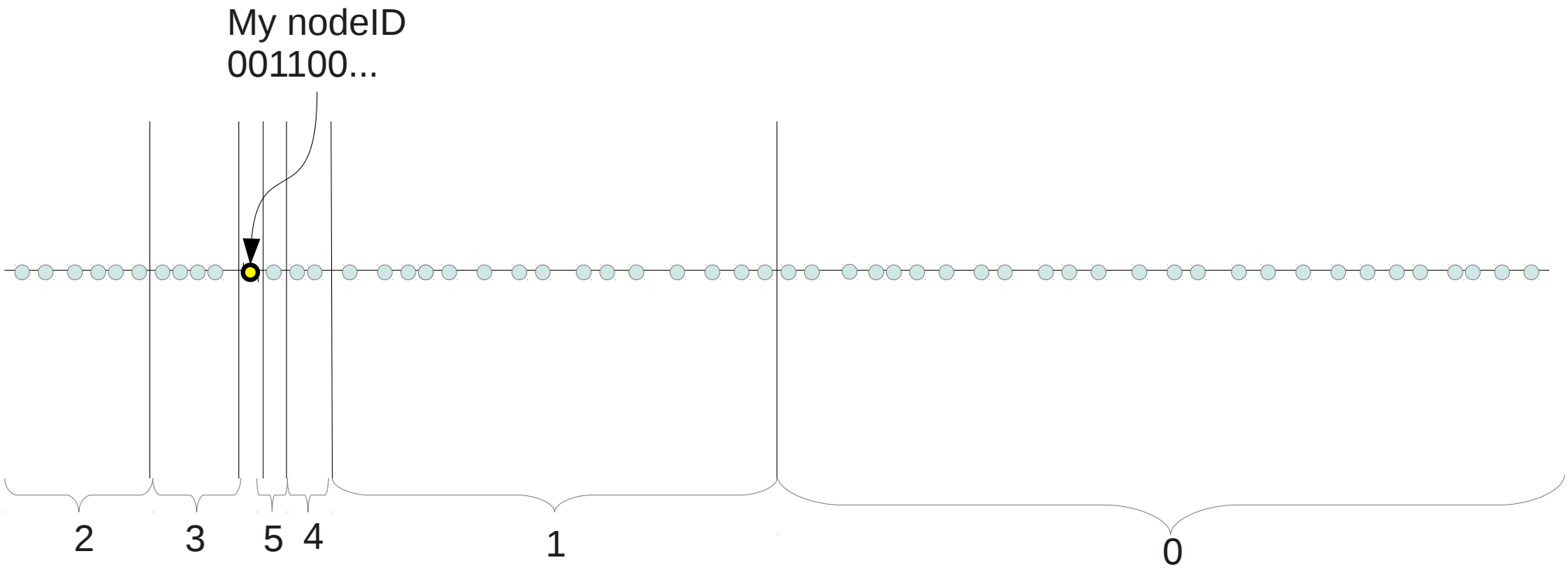
Kademlia's routing tables



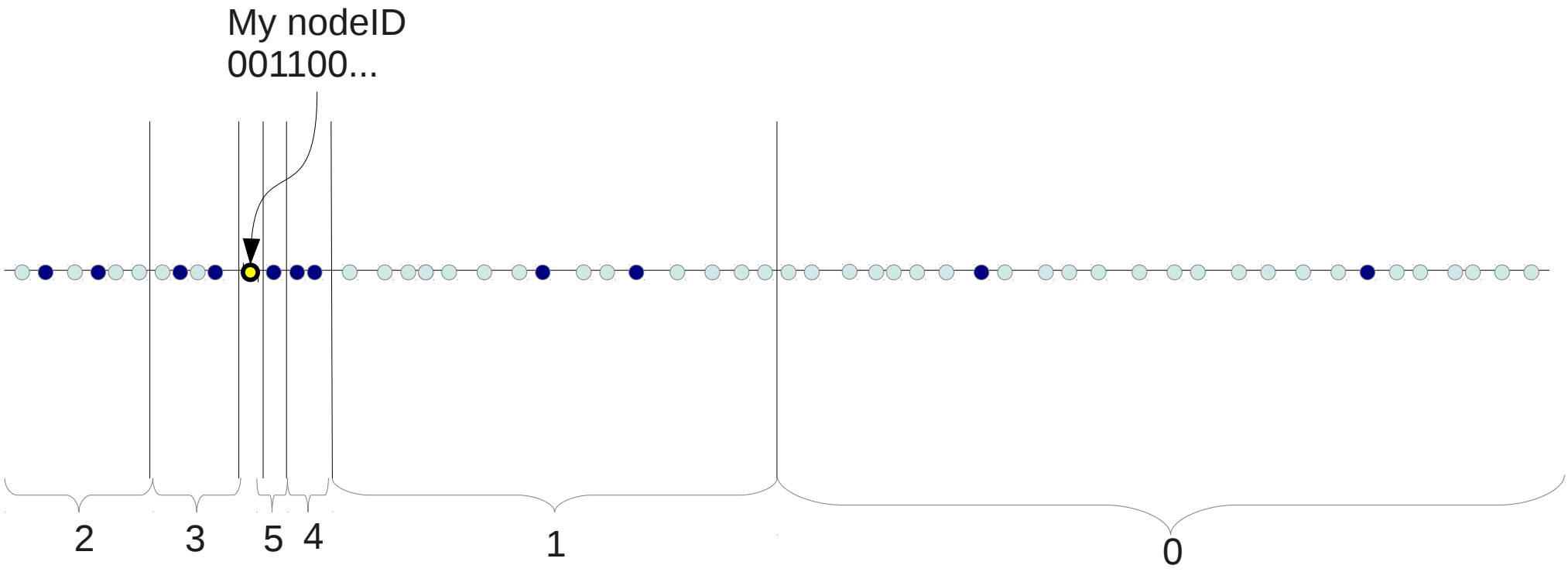
Kademlia's routing tables



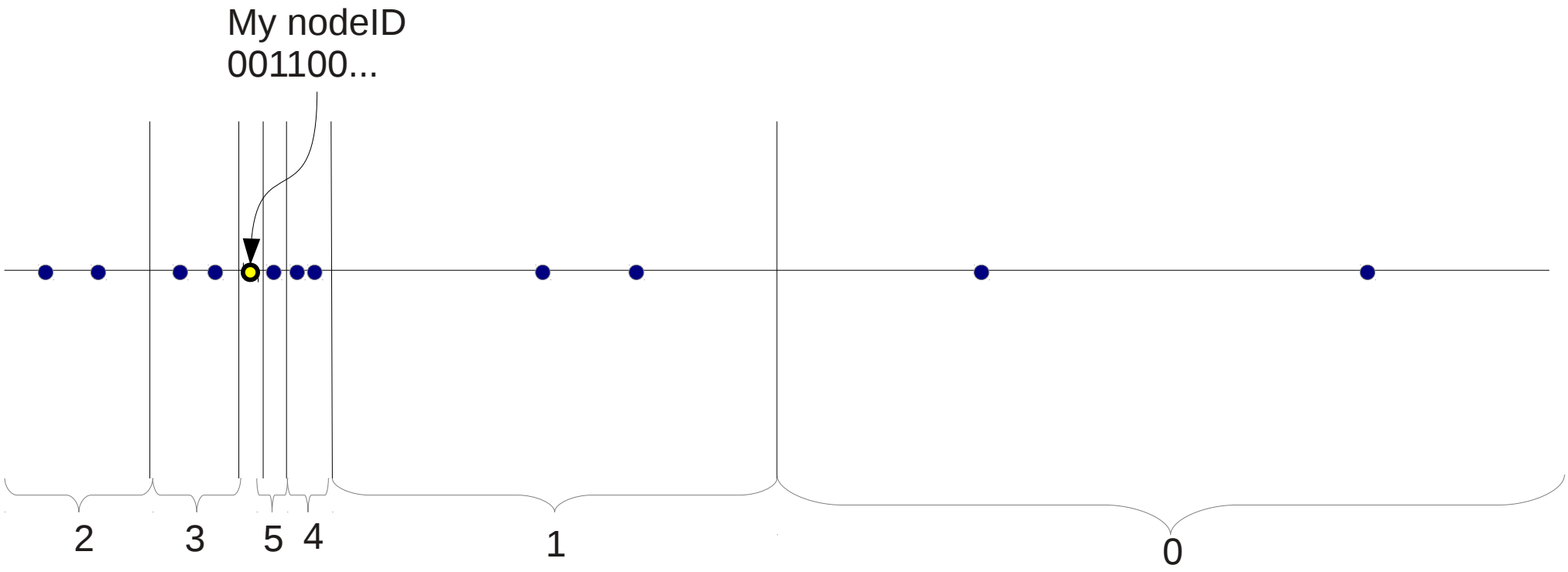
Kademlia's routing tables



Kademlia's routing tables

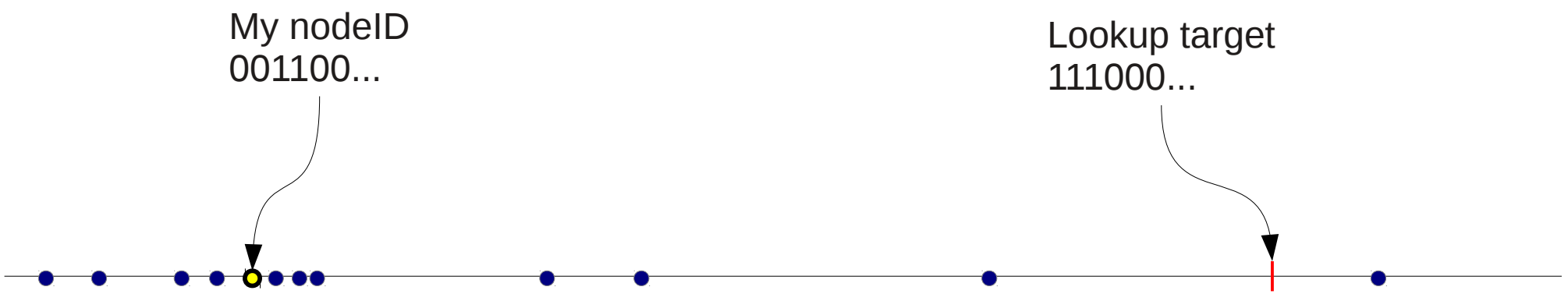


Kademlia's routing tables

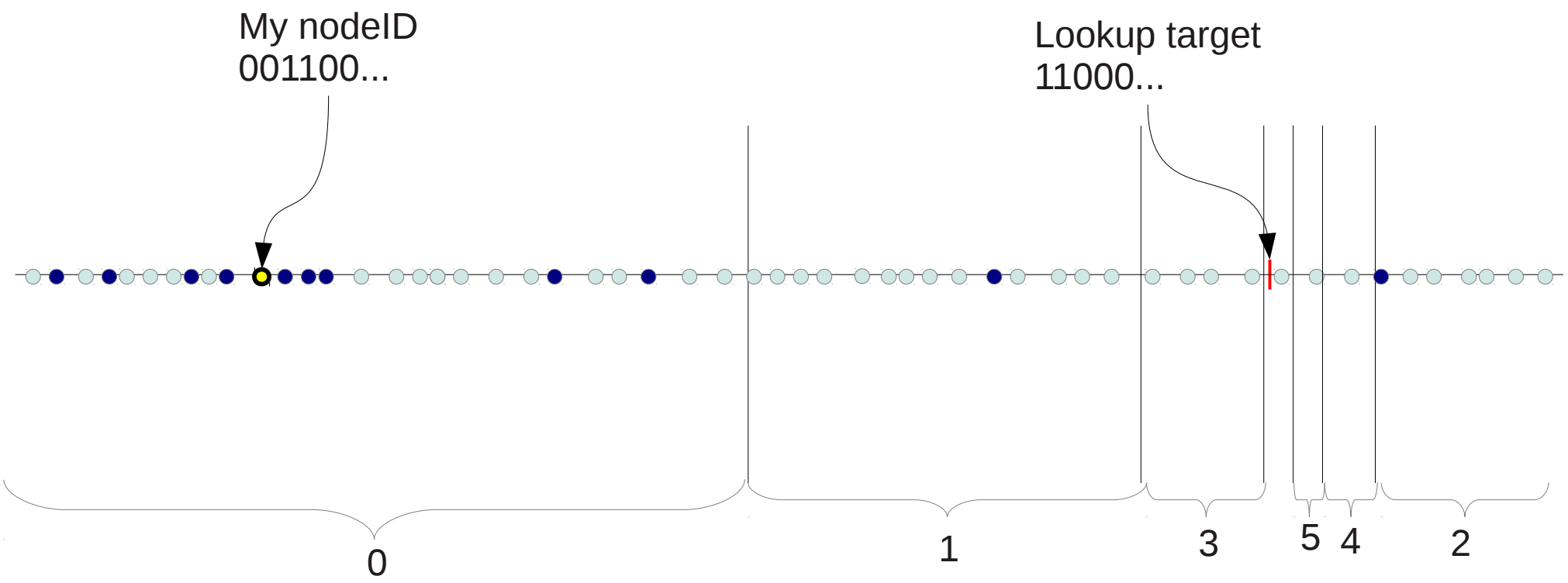


Kademlia's lookup

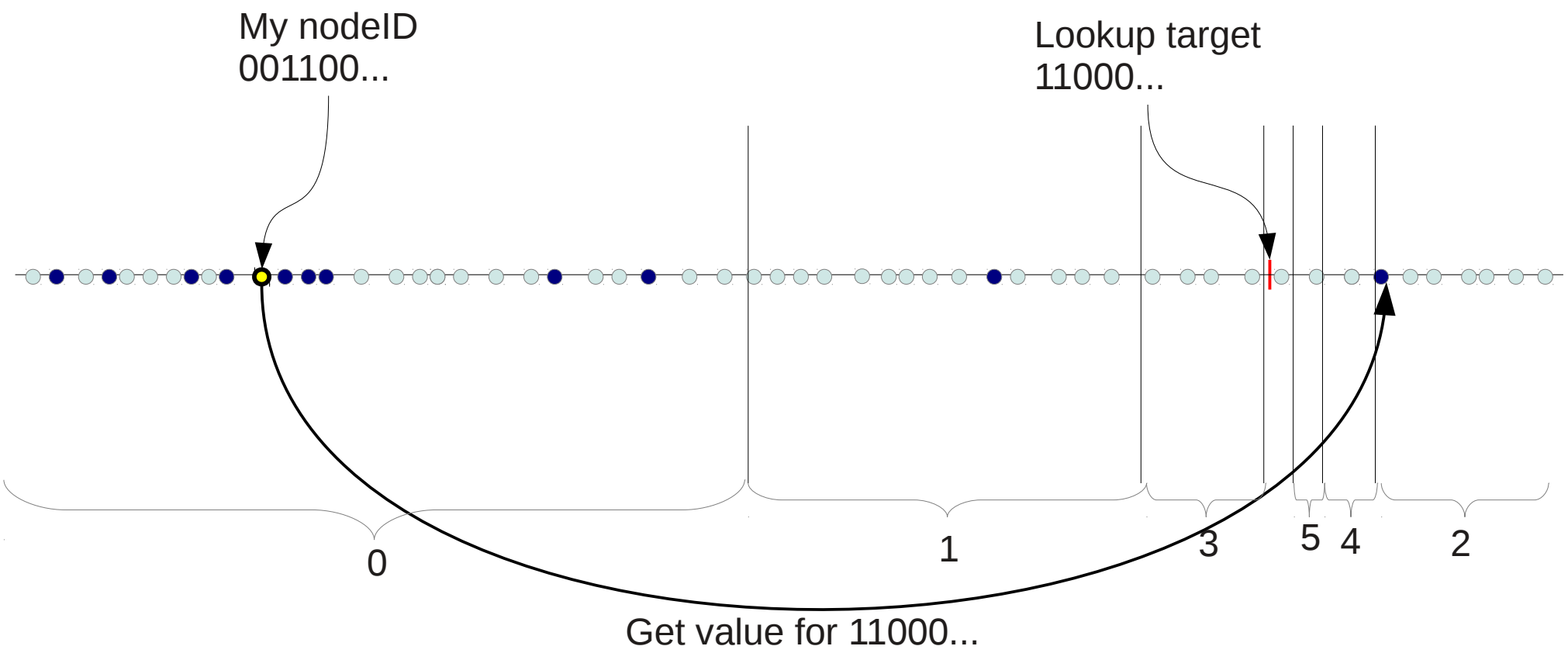
Kademlia's lookup



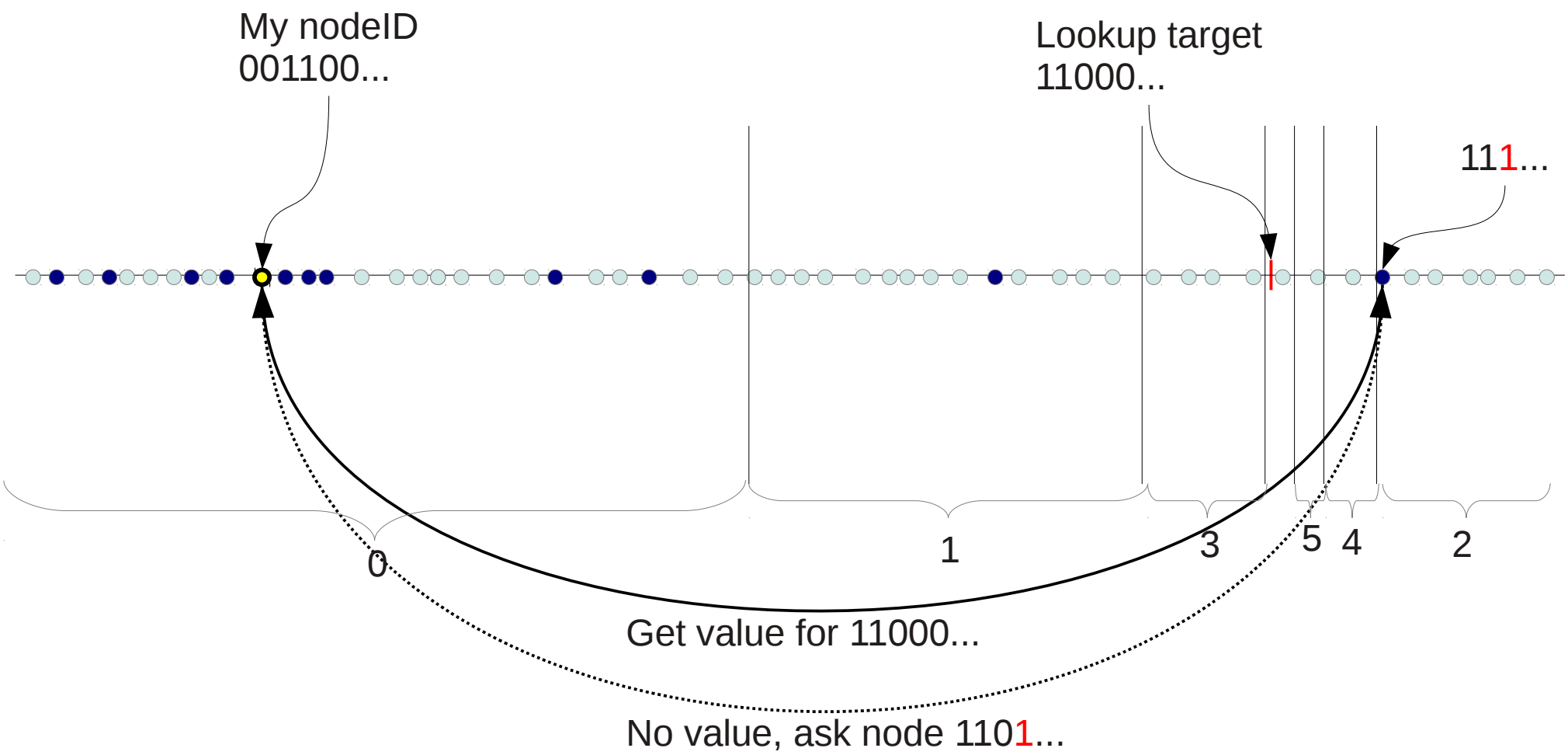
Kademlia's lookup



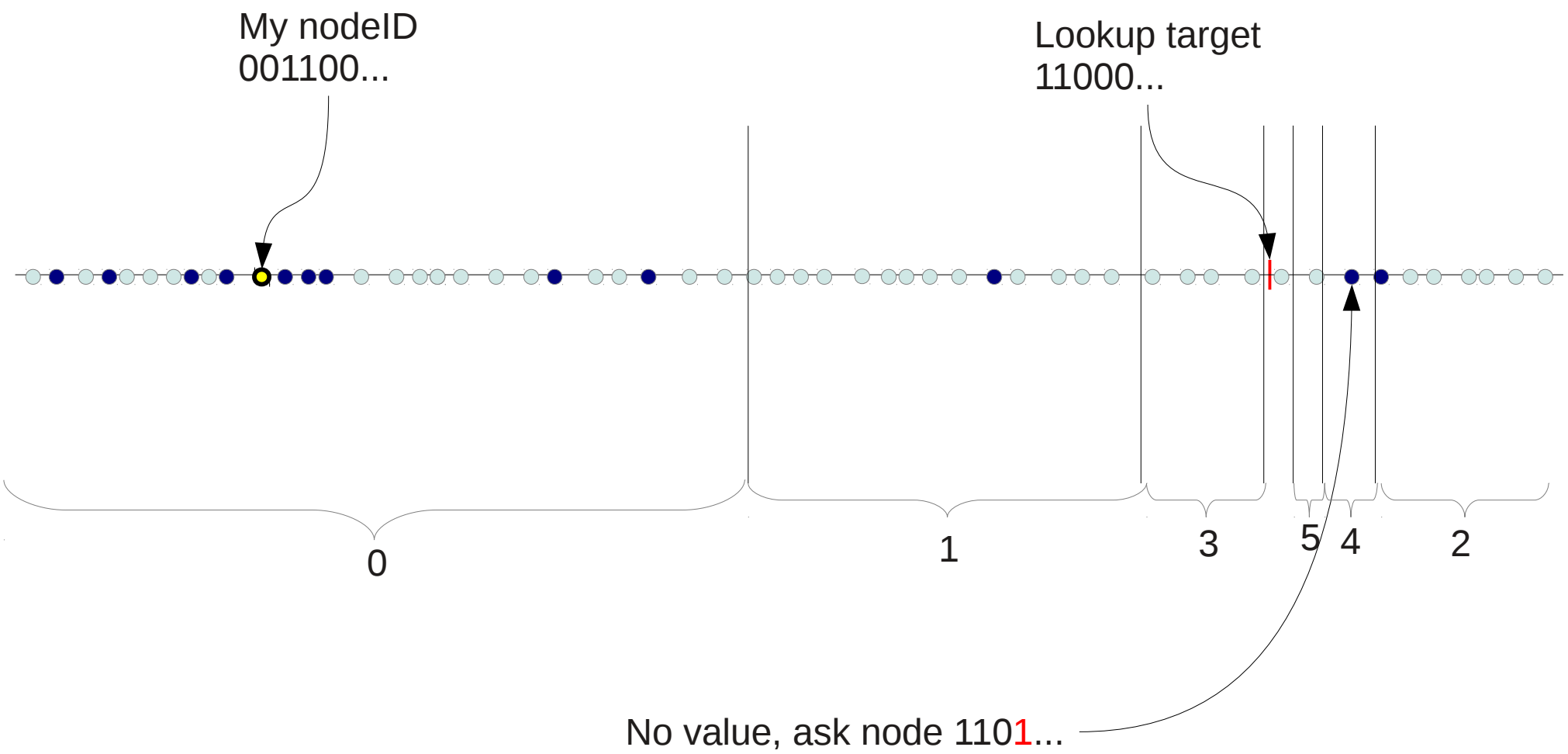
Kademlia's lookup



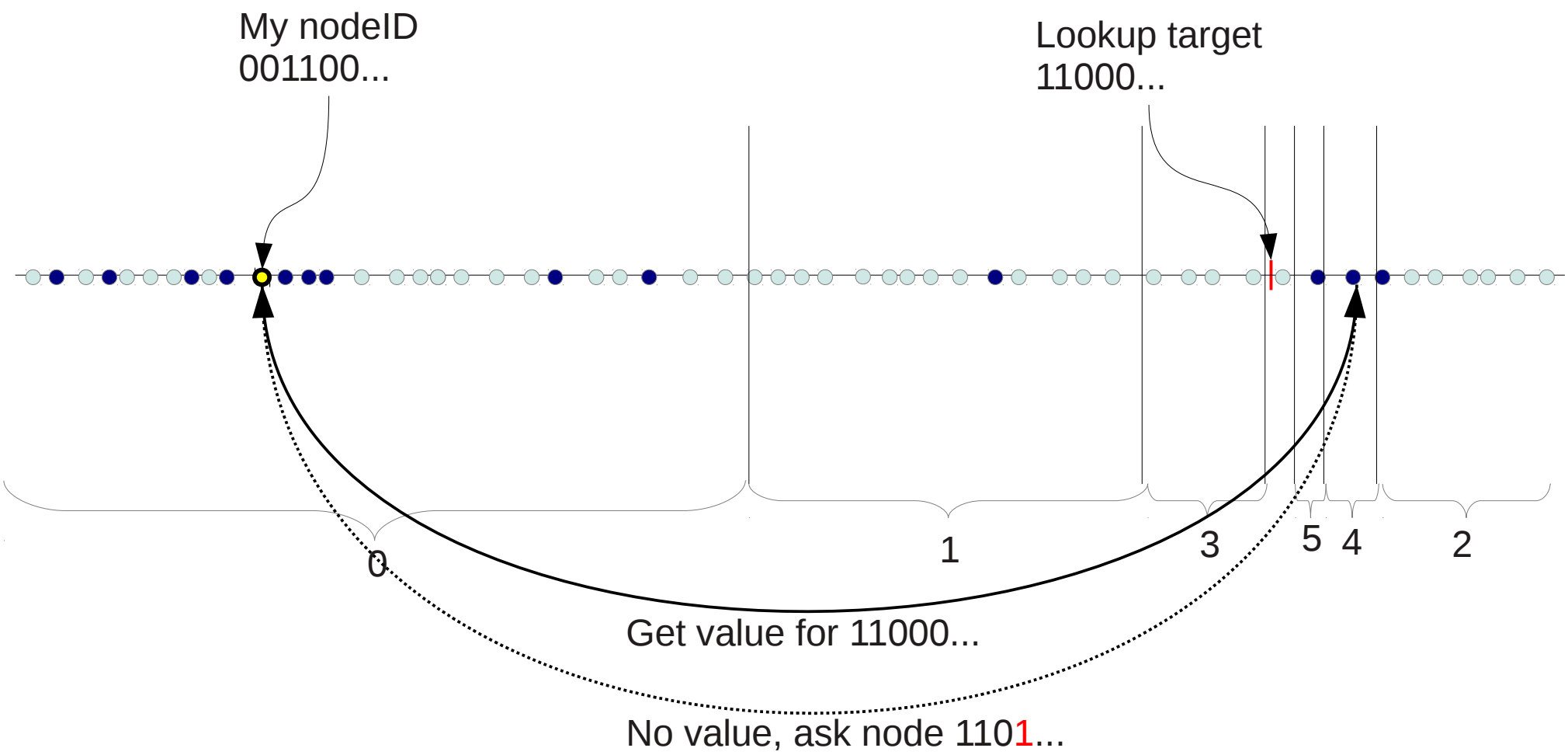
Kademlia's lookup



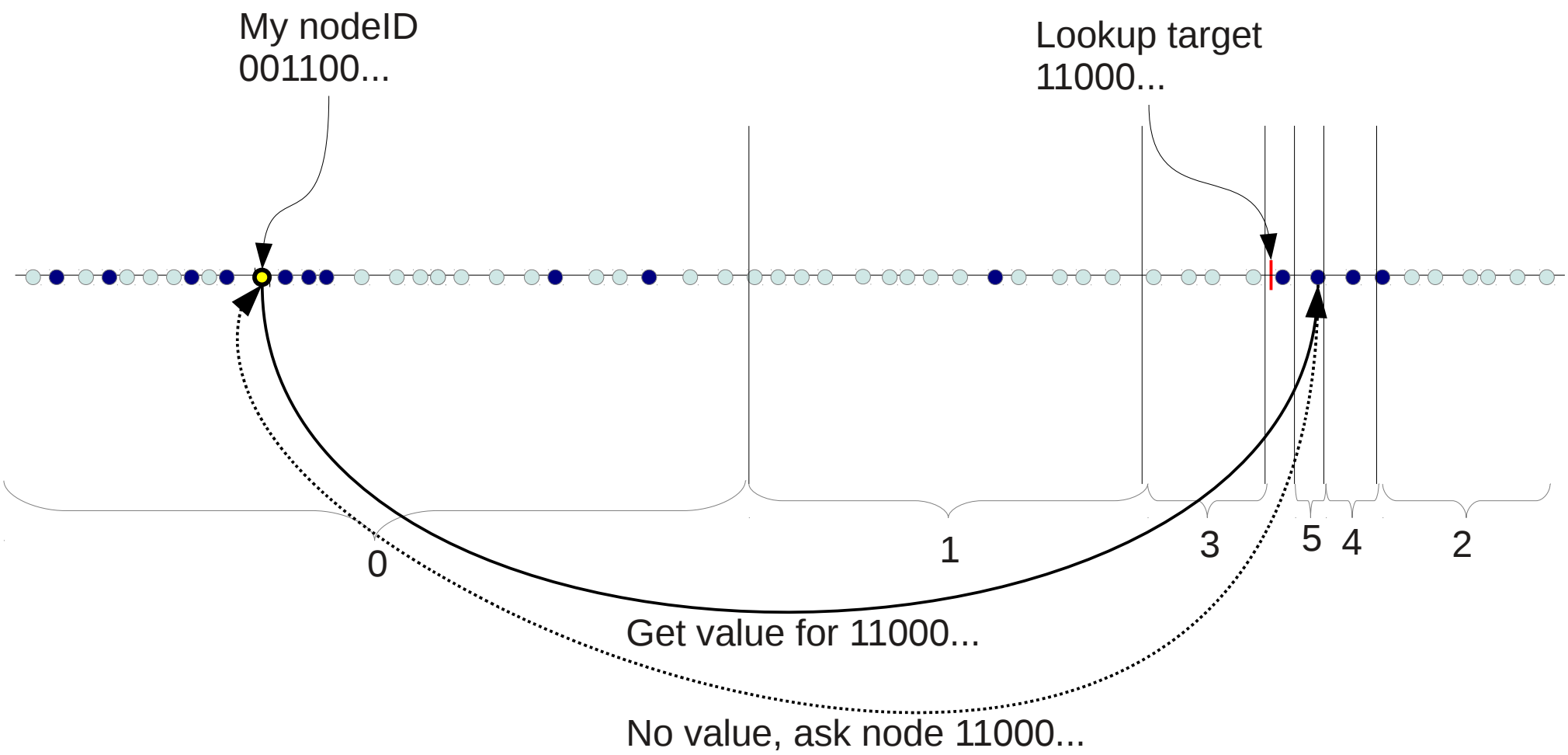
Kademlia's lookup



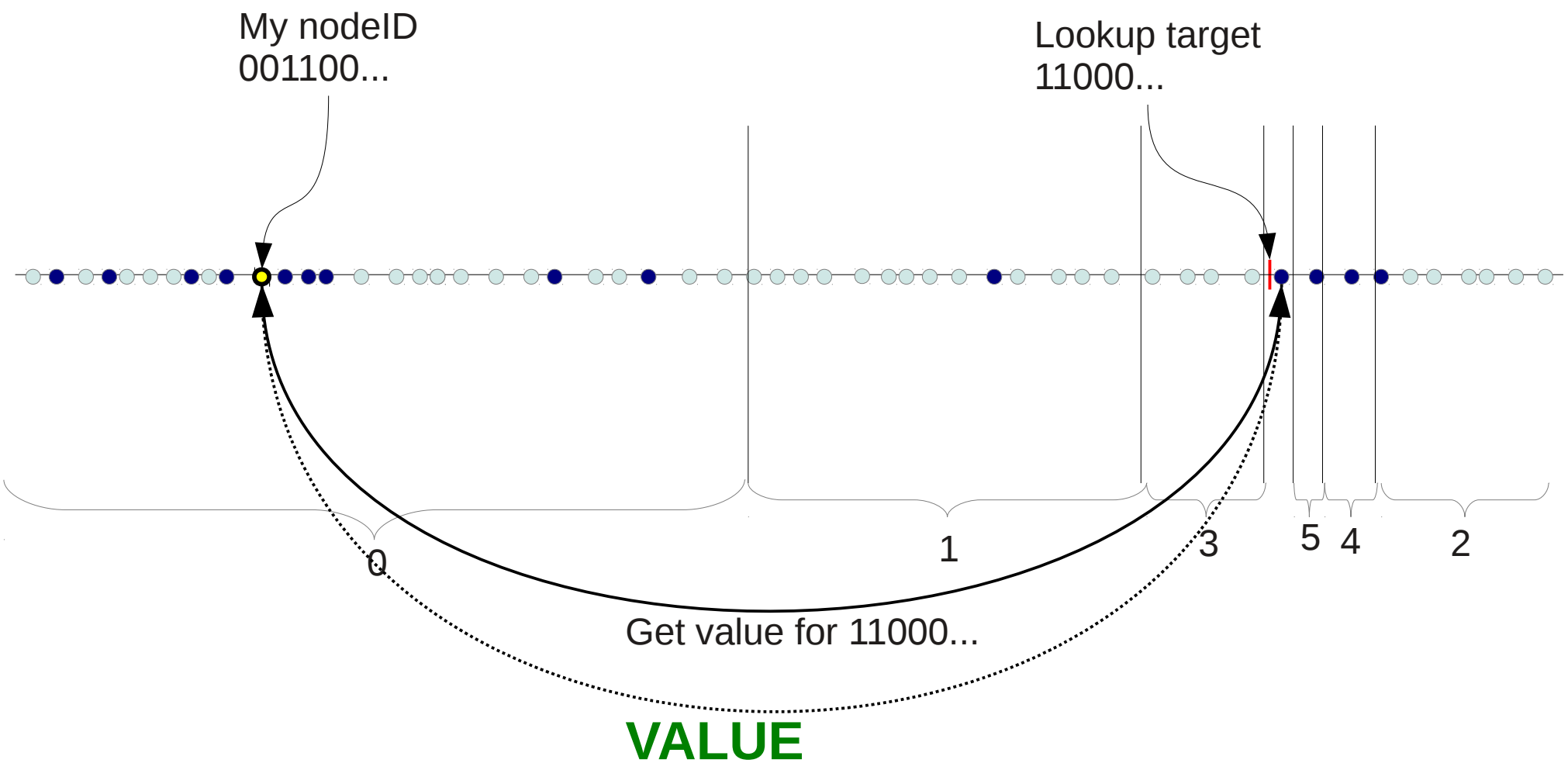
Kademlia's lookup



Kademlia's lookup



Kademlia's lookup



Kademlia's performance

- In the lab
  - [Li-05]
    - Median: 450 ms
  - [Kaune-08]
    - Median: 250 ms
  - [Rhea-05]
    - Median < 200 ms
    - 99 p < 600
  - [Dabek-04]
    - 100–300 ms
- On the Internet (1M+)
  - [Crosby-07]
    - MDHT median: ~1 min
    - ADHT median: ~2 min
  - [Falkner-07]
    - ADHT median: 13 s
  - [Stutzbach-06]
    - KAD median: 2 s
  - [Steiner-09]
    - KAD median: 1.5 s

</background>

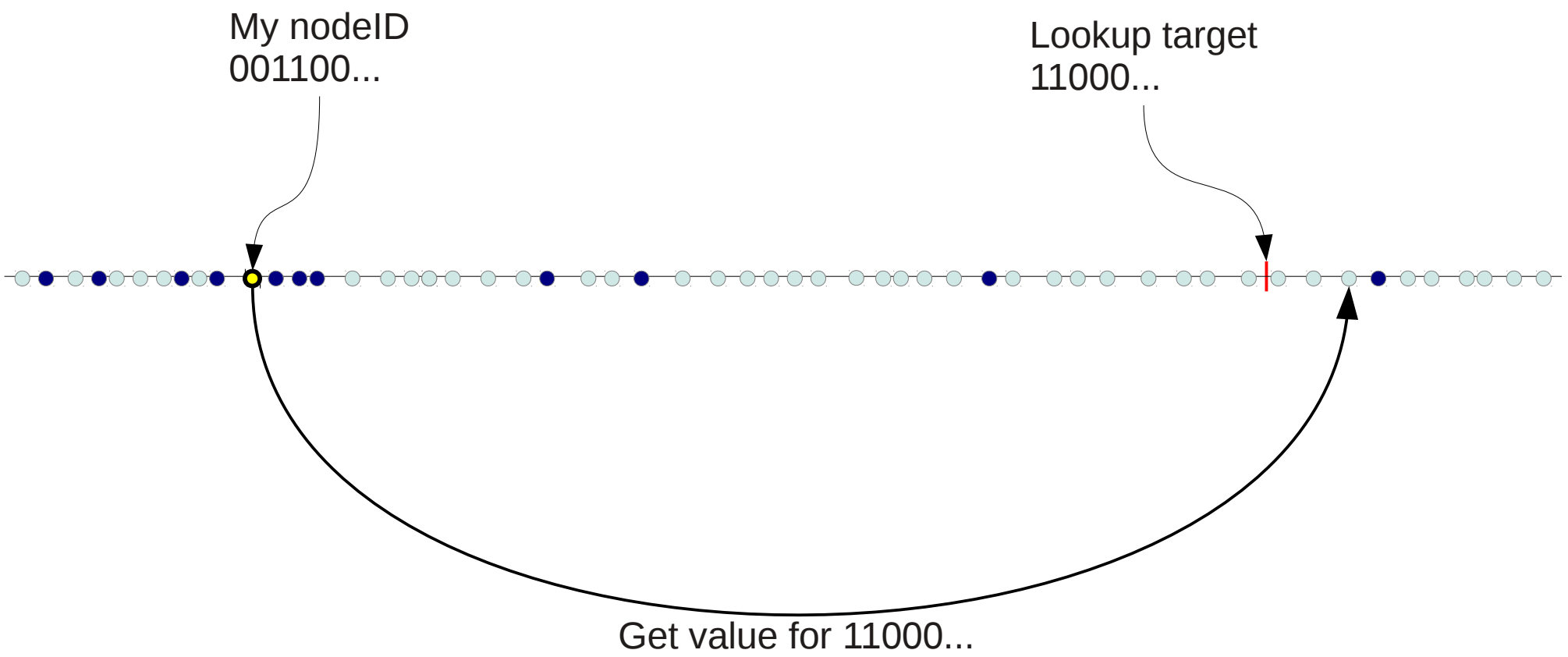
<main results>

Main results

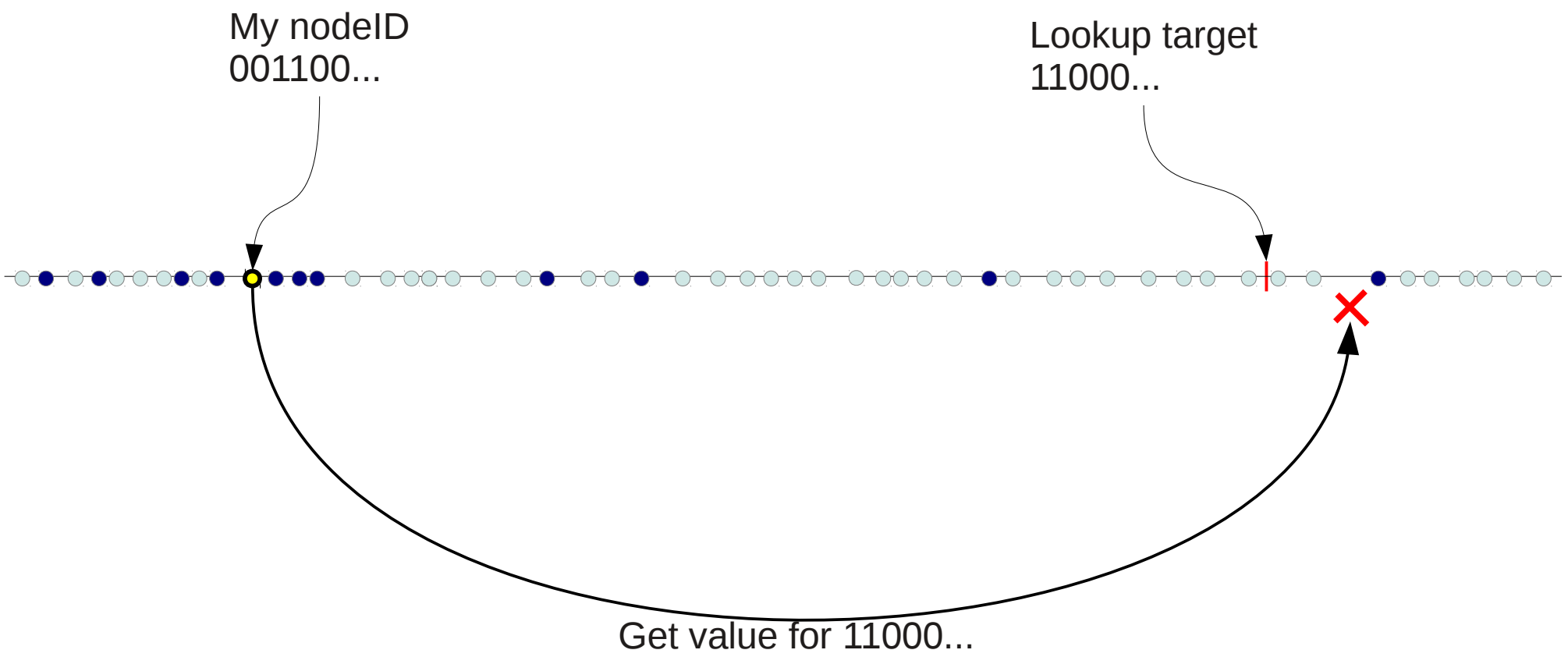
- Connectivity properties
- Sub-second lookups
- Scalability & Locality

Connectivity Properties of Mainline DHT Nodes

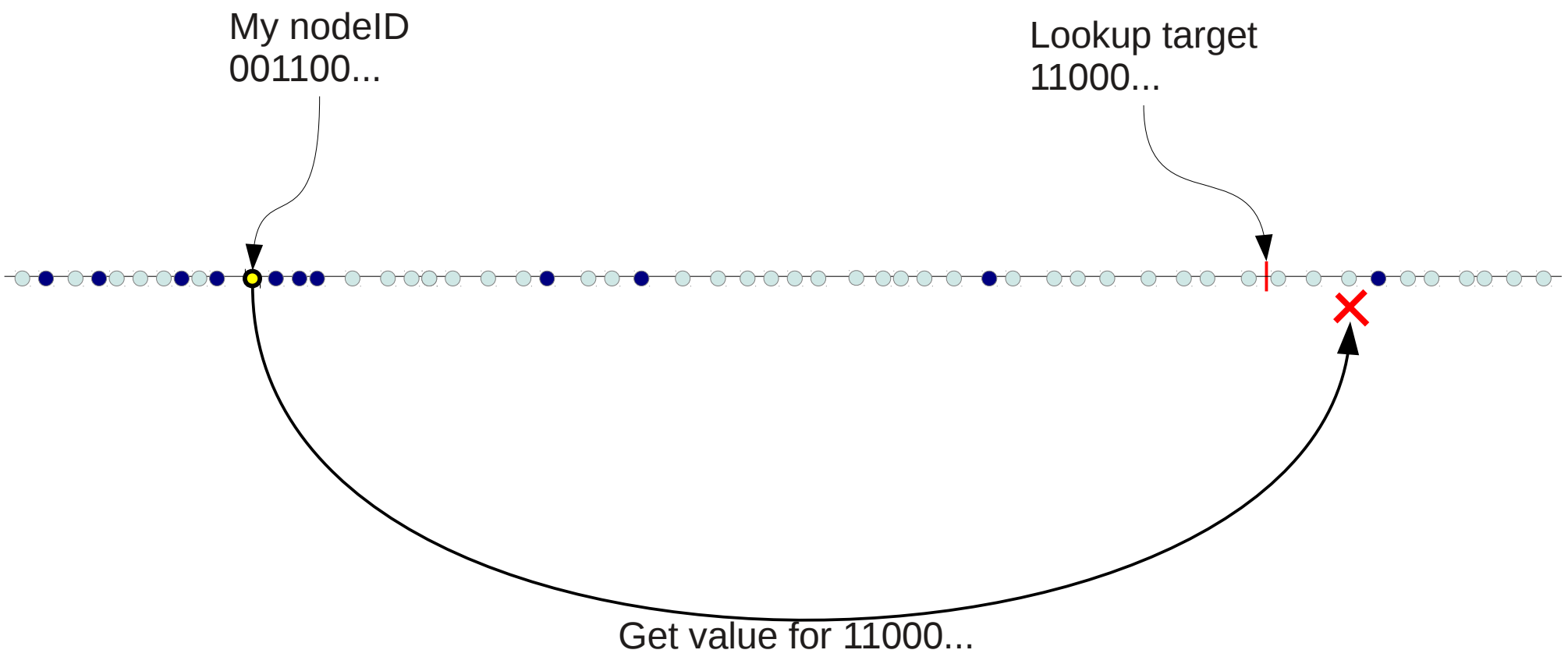
Why Mainline DHT is so slow?



Why Mainline DHT is so slow?

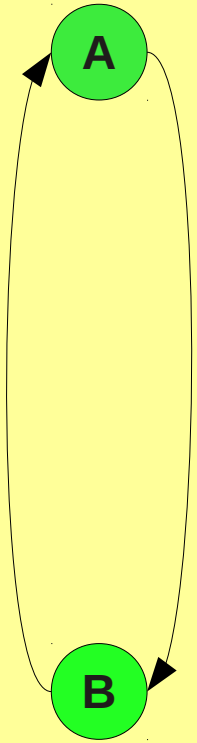


Why Mainline DHT is so slow?

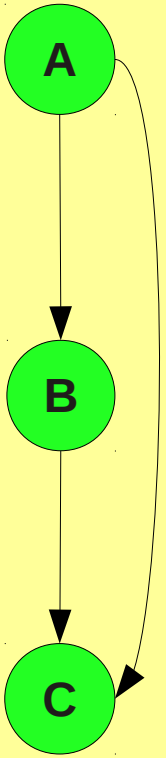


Connectivity properties

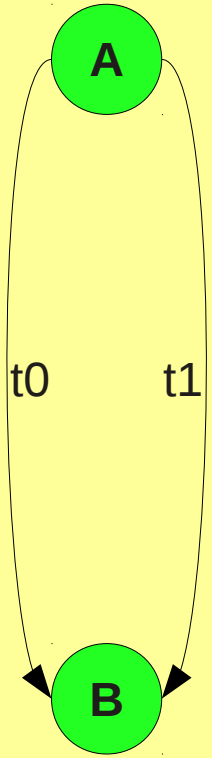
Reciprocity



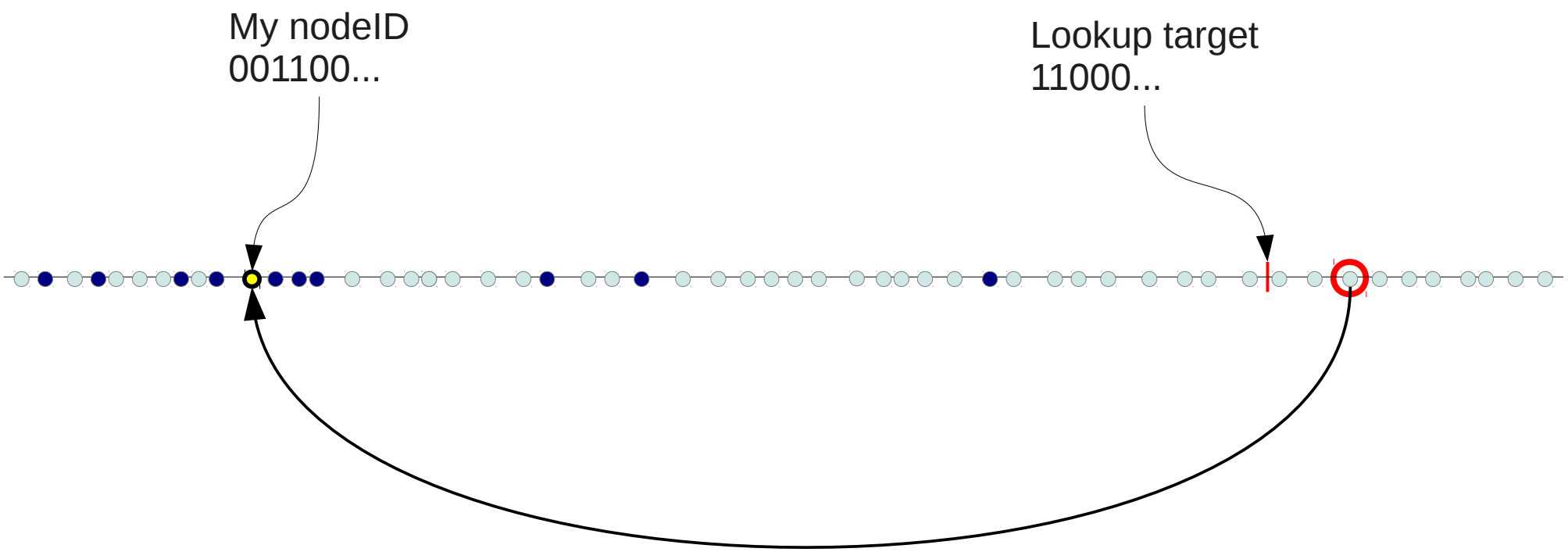
Transitivity



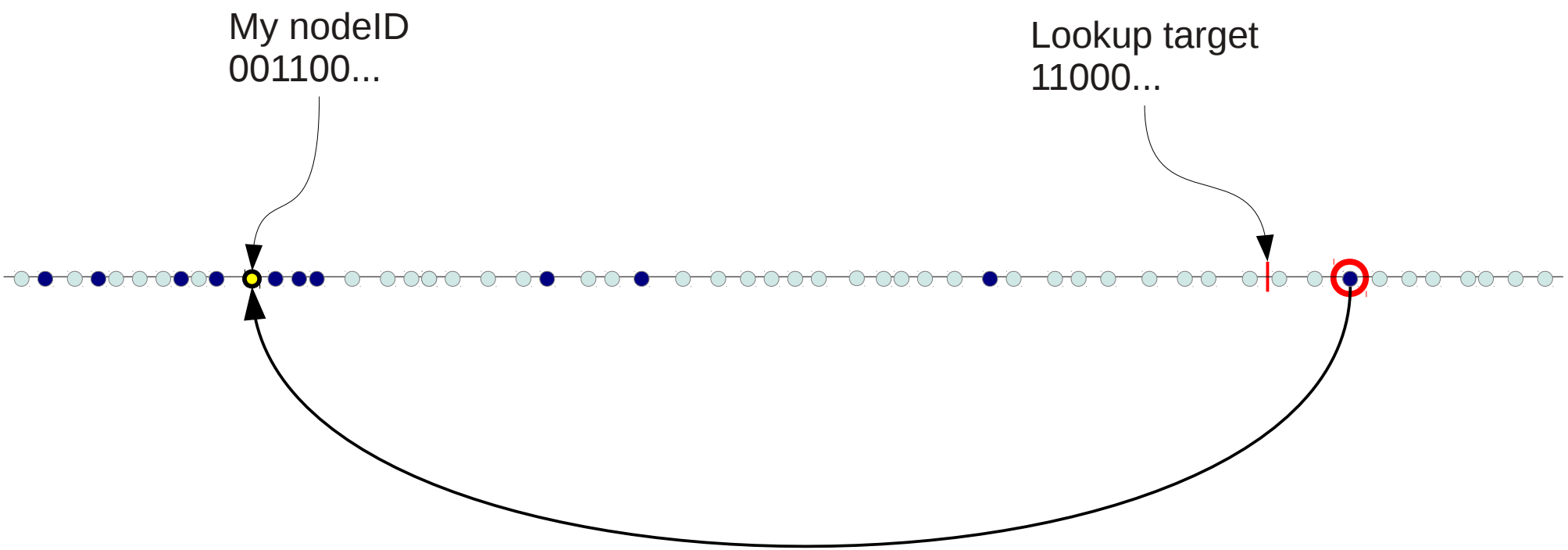
Persistence



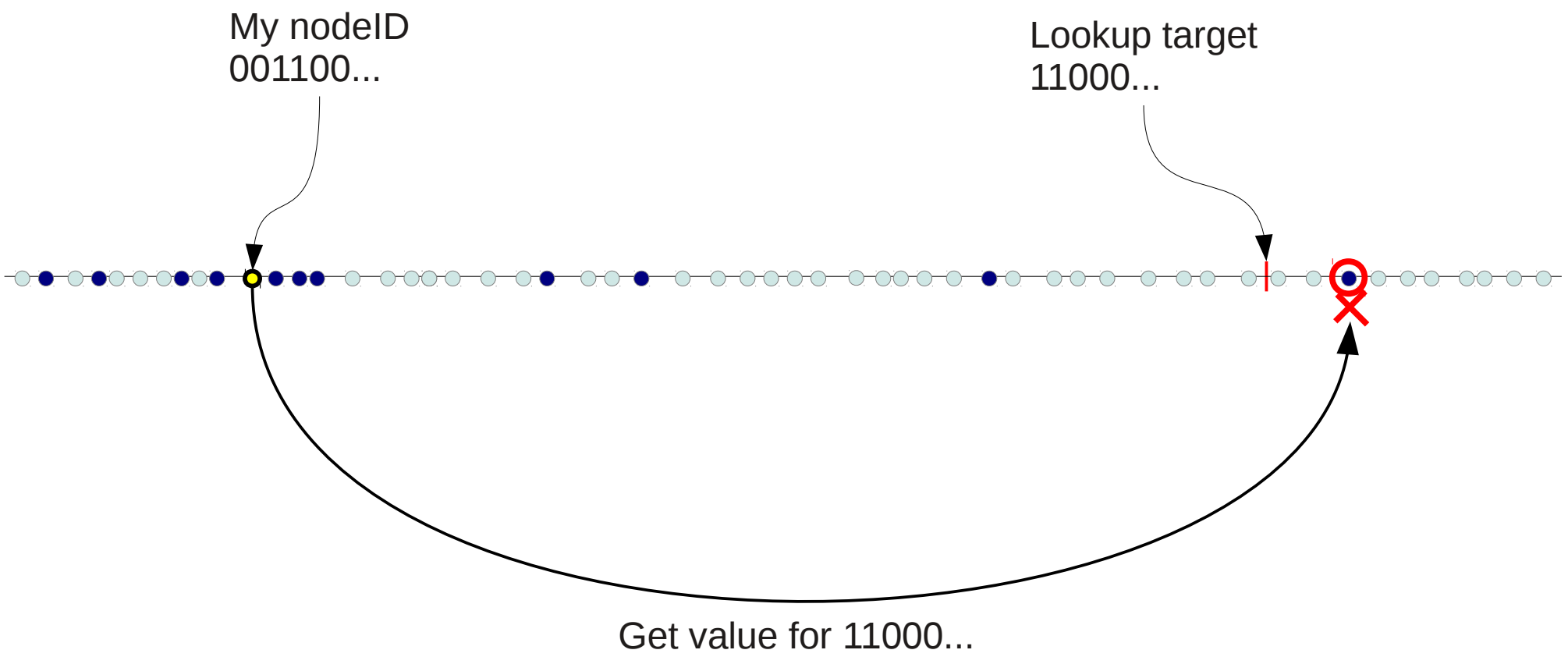
Reciprocity / Persistence



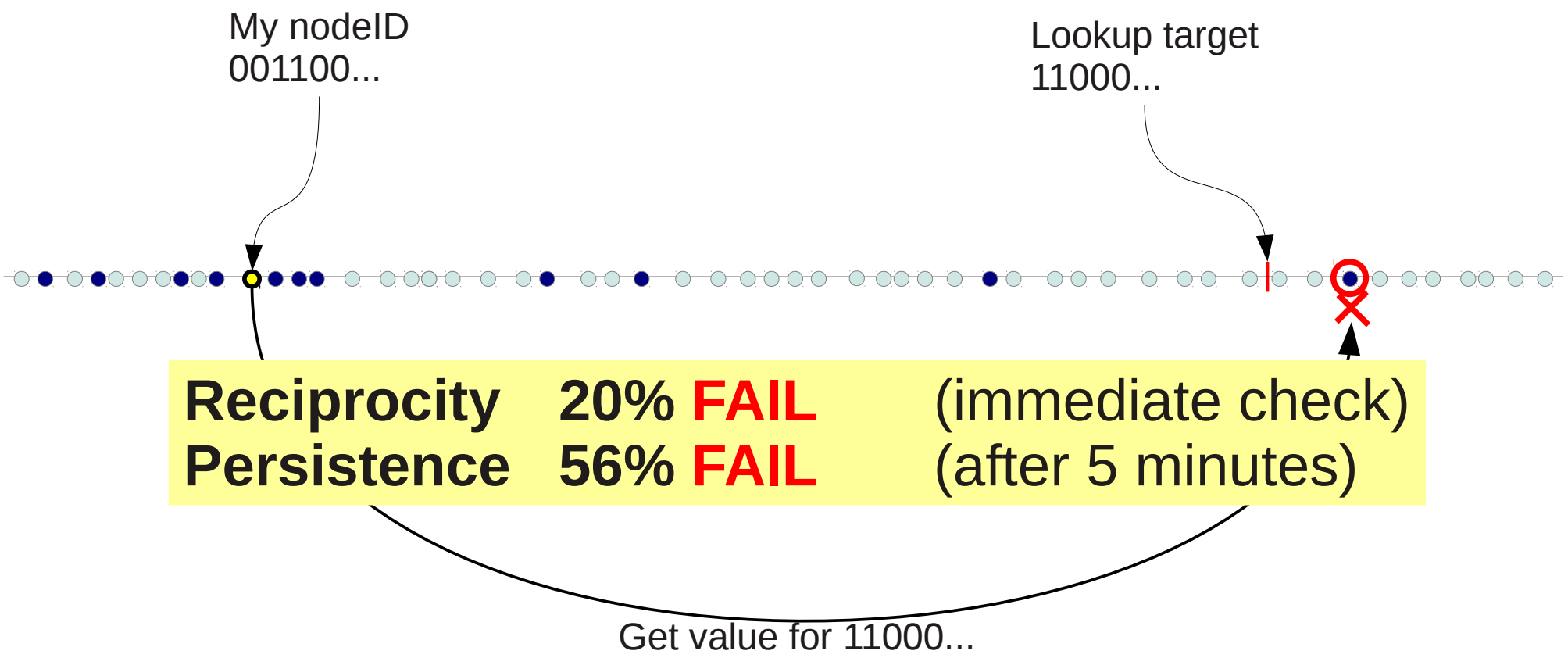
Reciprocity / Persistence



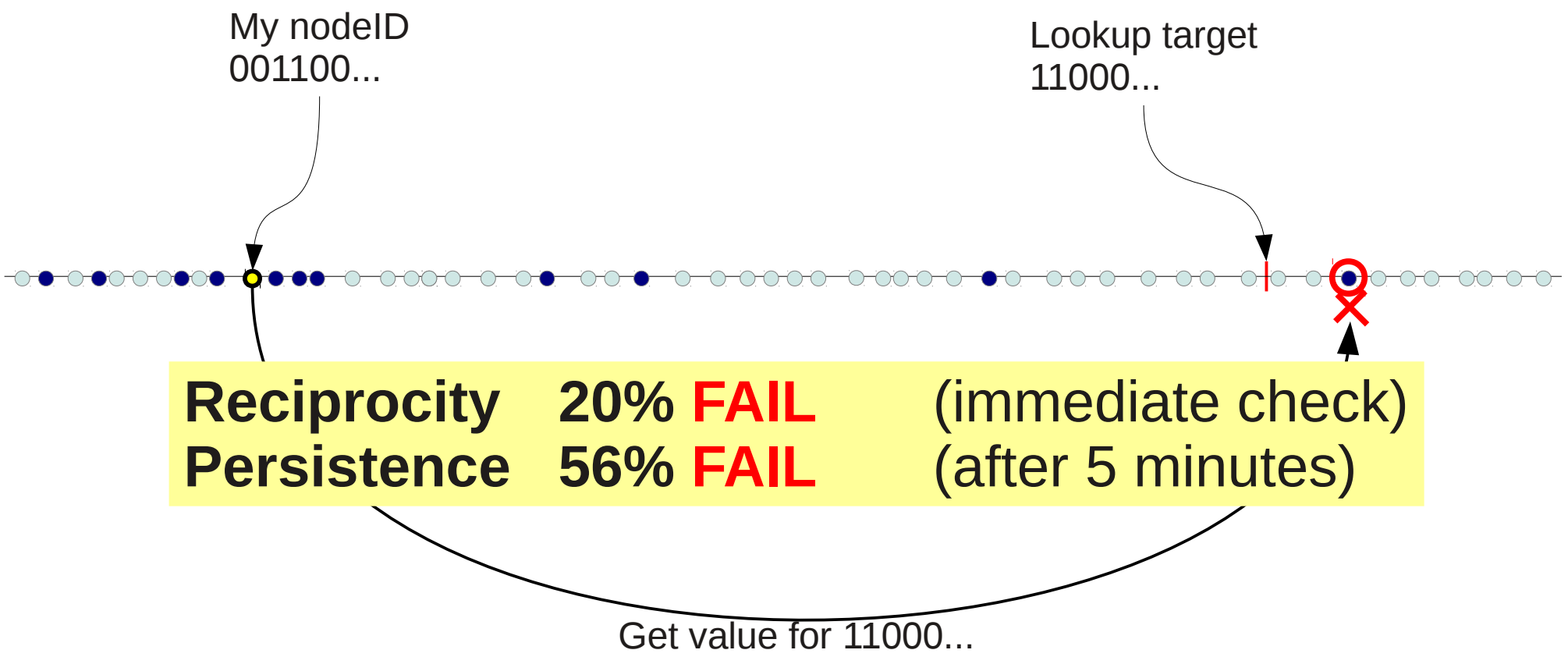
Reciprocity / Persistence



Reciprocity / Persistence



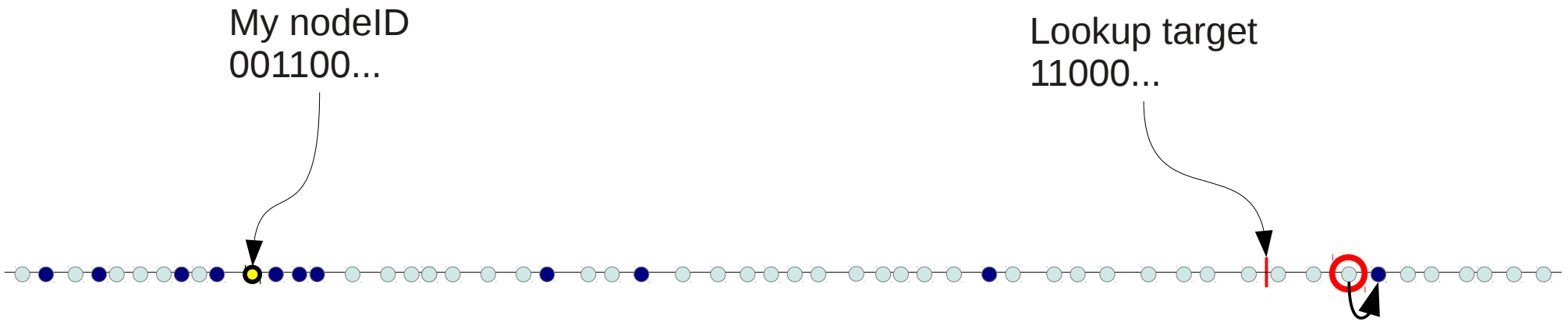
Reciprocity / Persistence



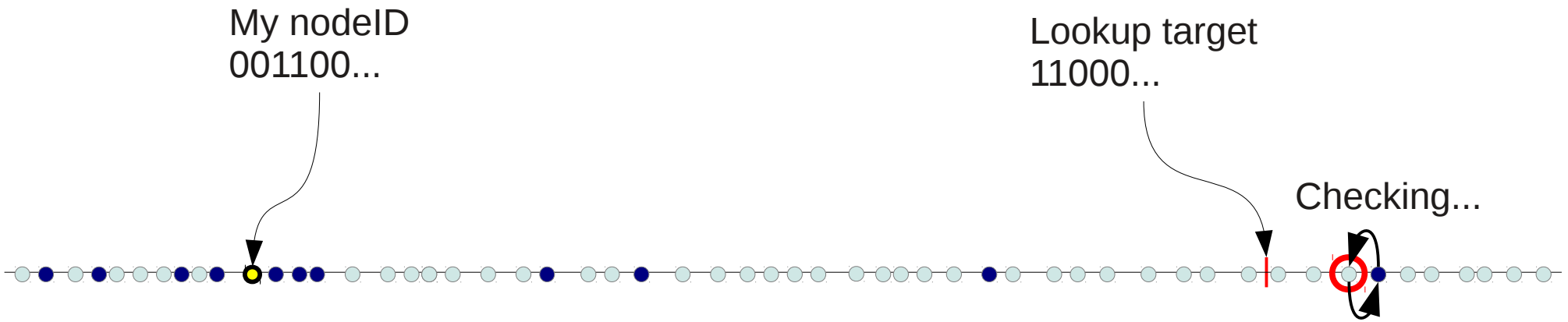
Solution:

~~**Opportunistic**~~ → **connectivity check** → **quarantine**

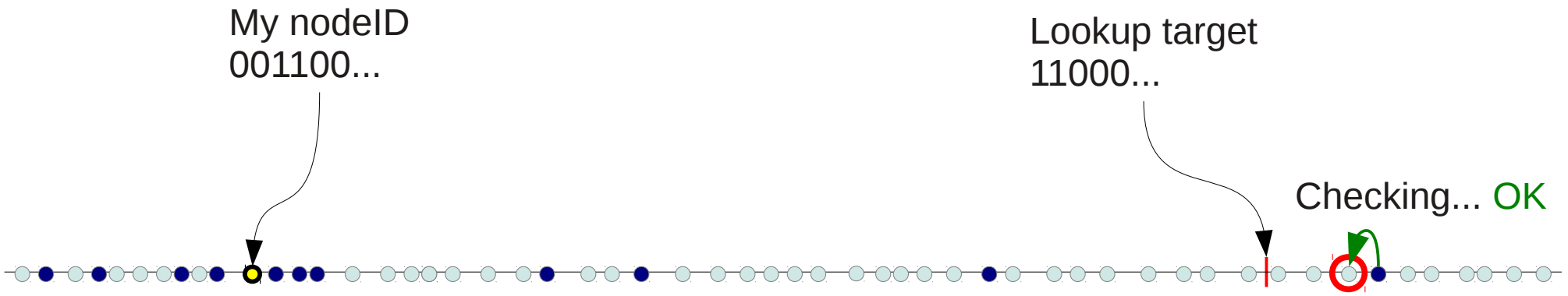
Transitivity



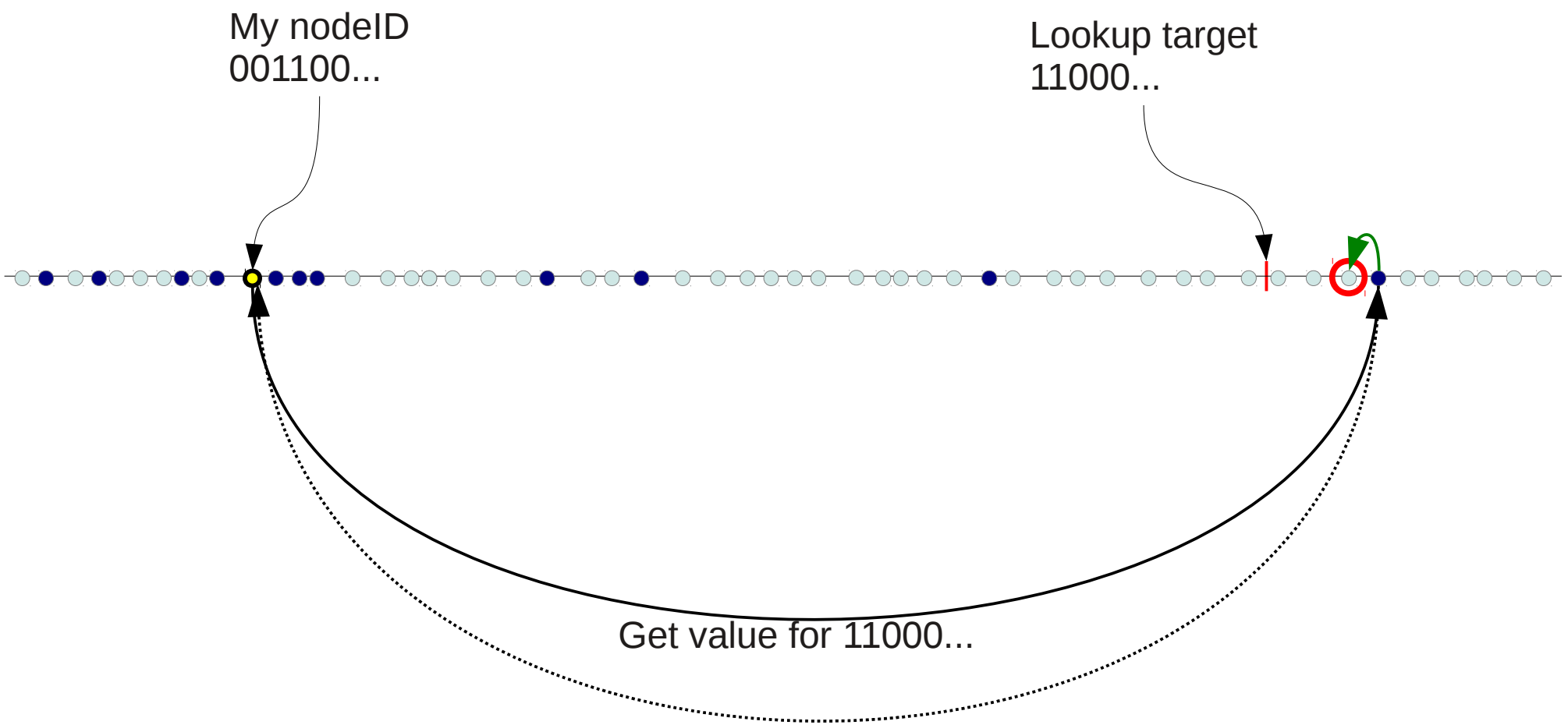
Transitivity



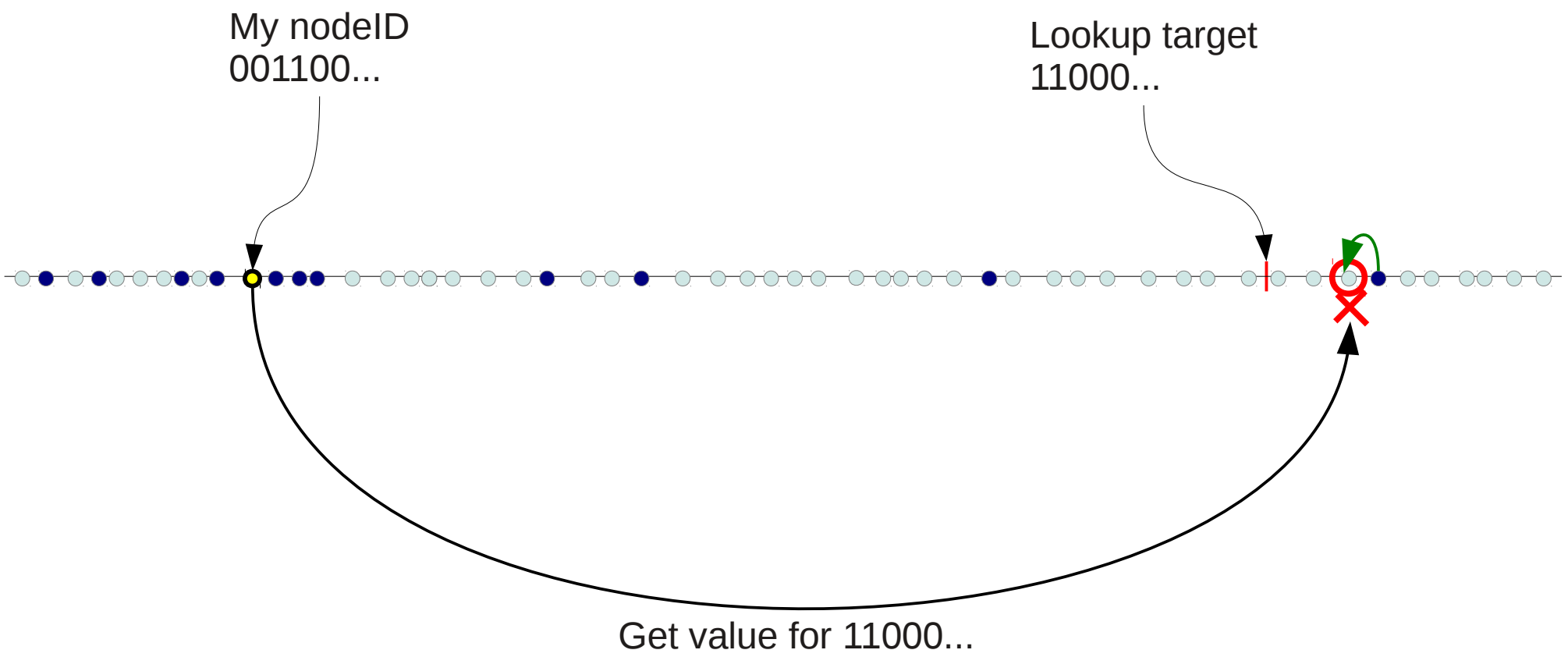
Transitivity



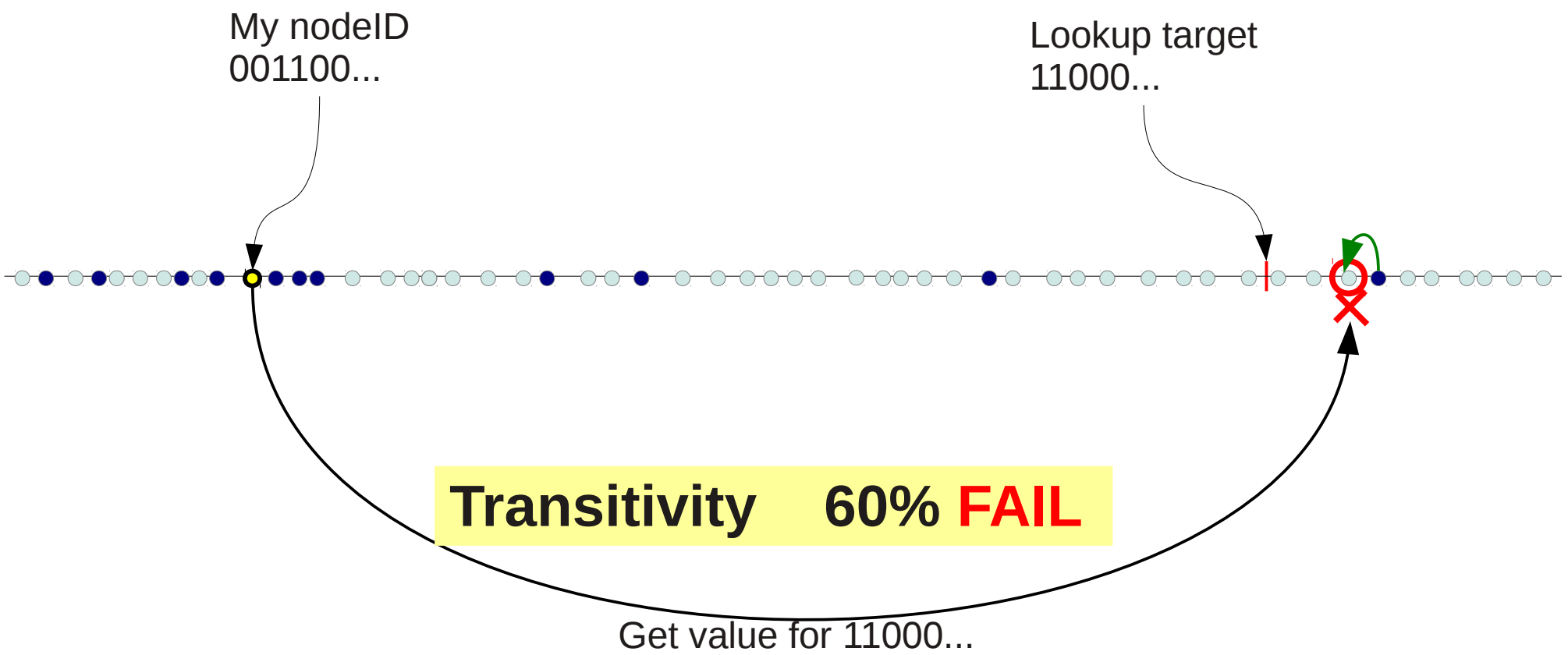
Transitivity



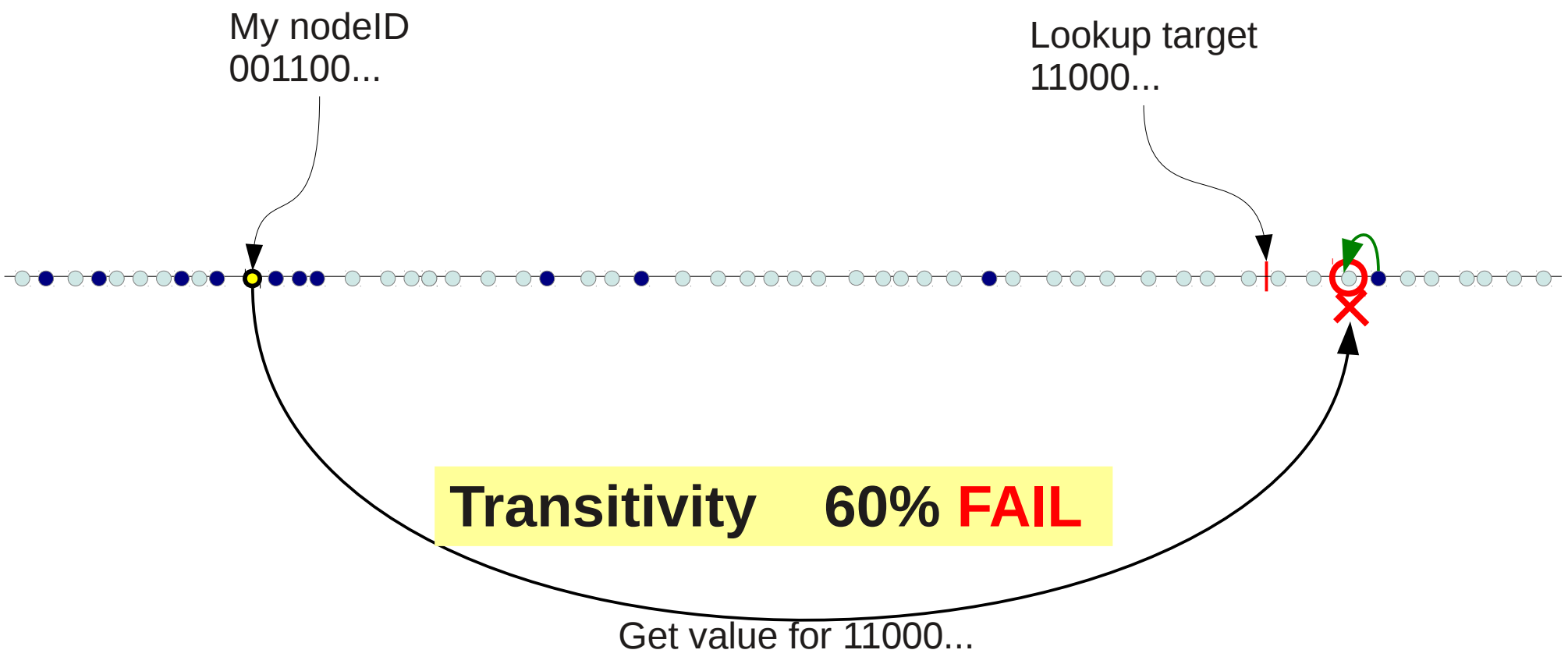
Transitivity



Transitivity

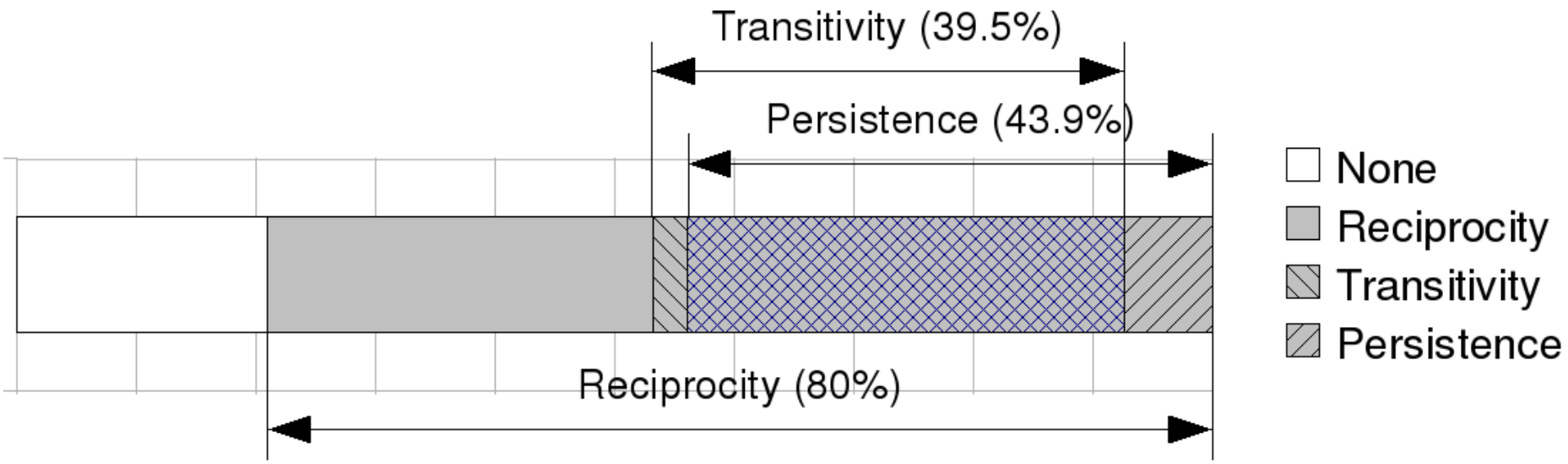


Transitivity



Solution:
Check from different IP (hard!)

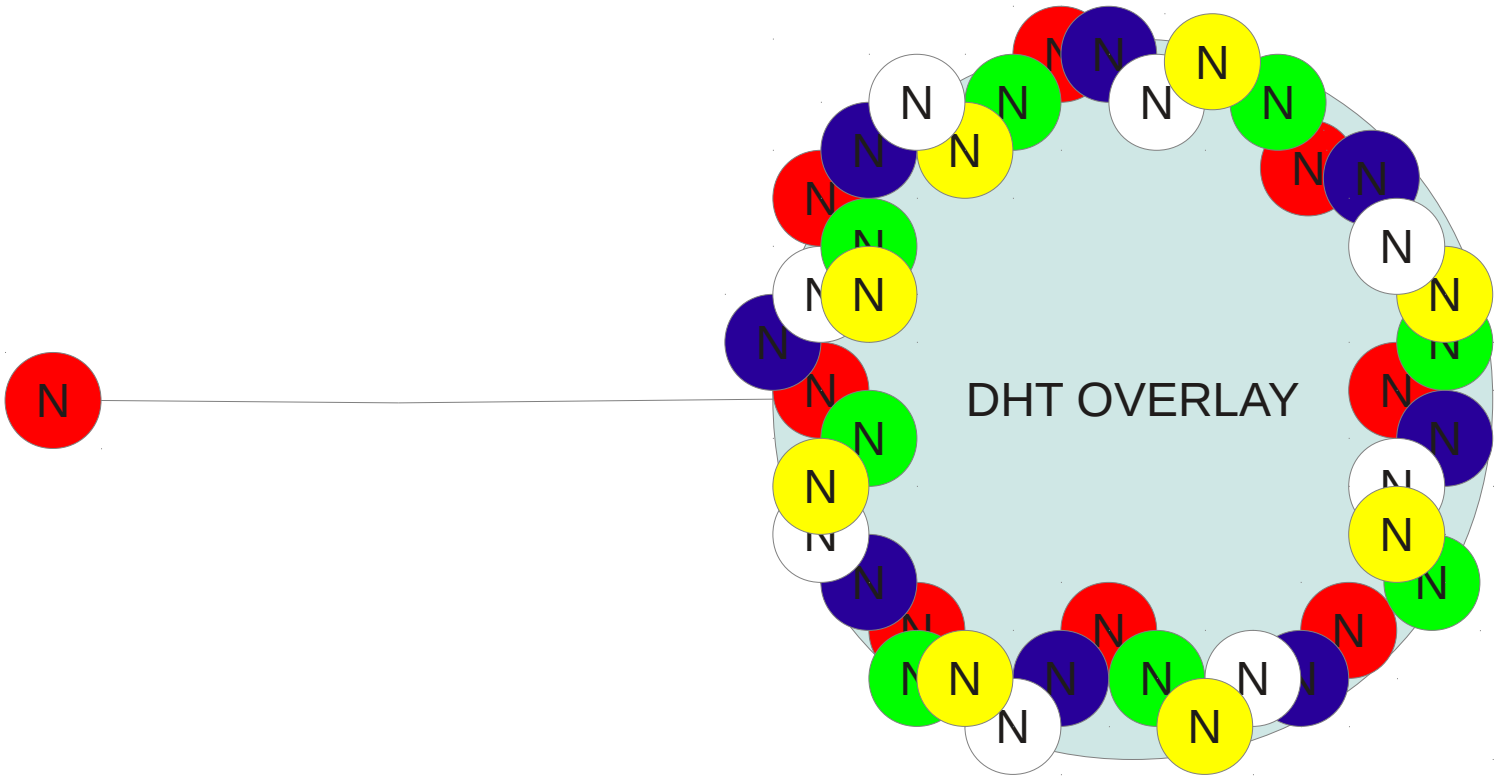
Connectivity Properties



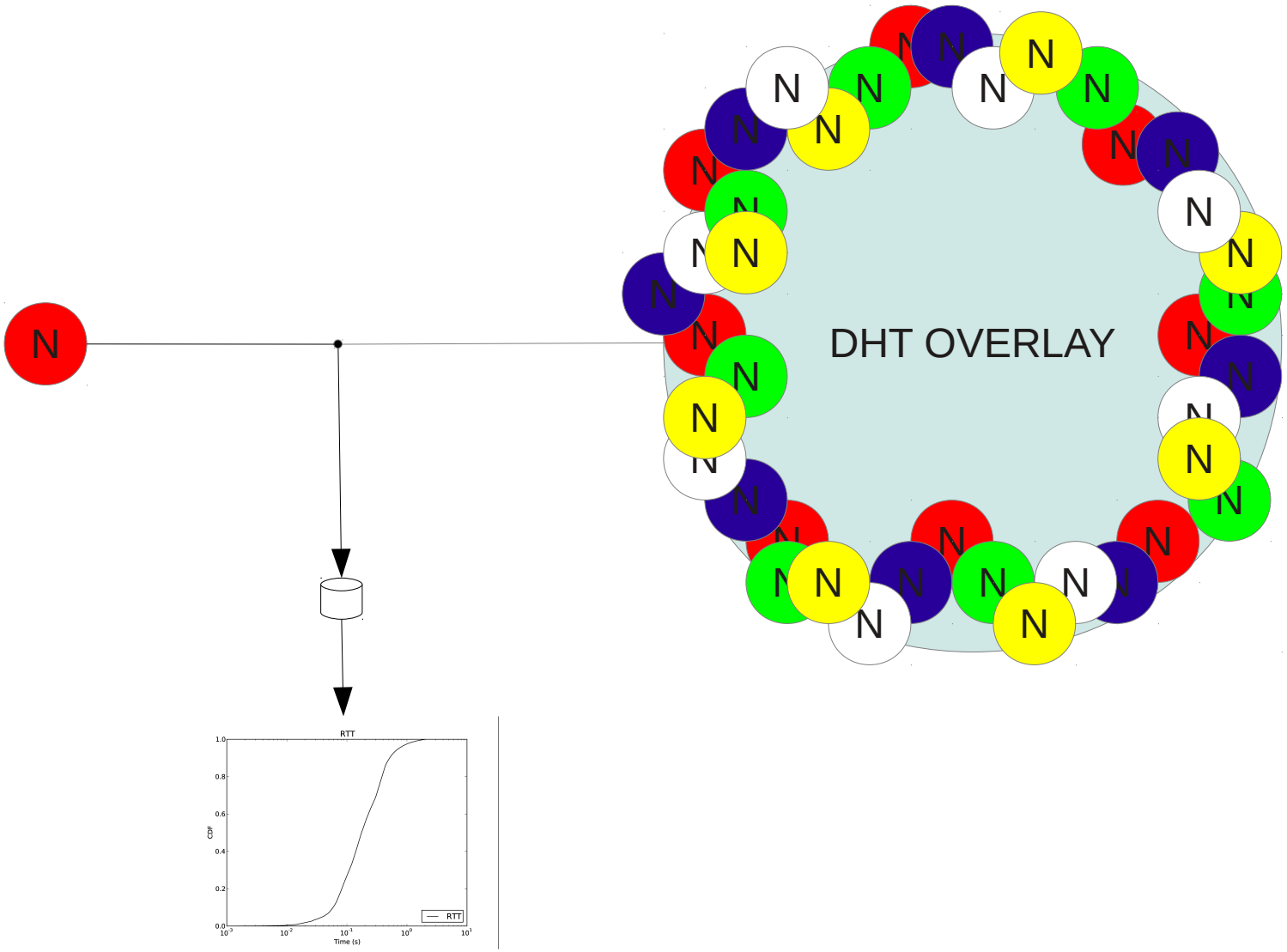
- 2/3 of the nodes show “bad connectivity”
- Clear overlap: transitivity & persistence
  - Quarantine could be “good enough”

Sub-Second Lookups in Mainline DHT

Measuring performance



Measuring performance



µTorrent

- 60% of Mainline DHT nodes
- Median lookup latency: **~650 ms**
- But …
  - **> 1 s 27% of lookups**

Can we do better?

(while keeping backward-compatibility)

Our MainlineDHT nodes

- **Modular architecture**
  - **Lookup** module
  - **Routing table management** module

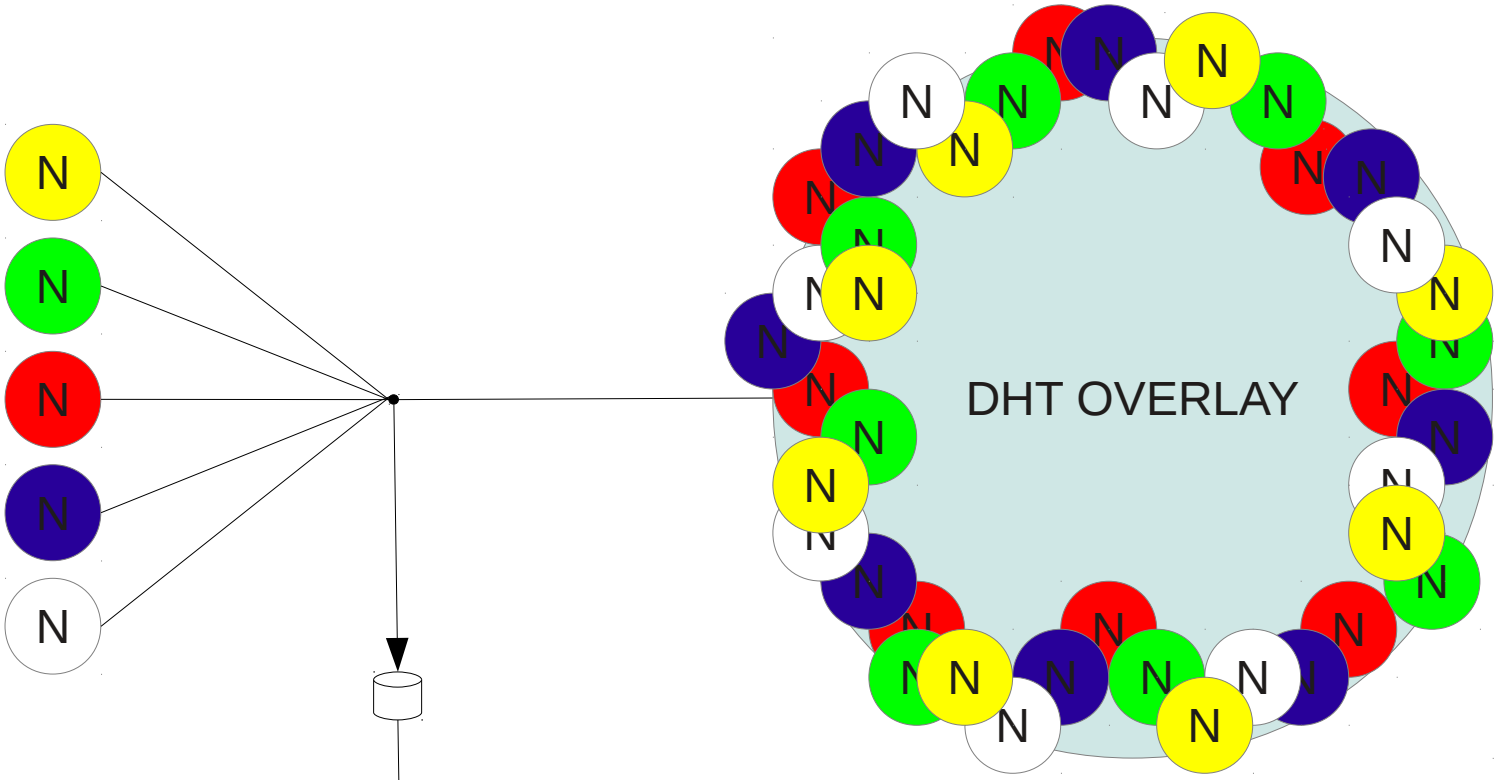
Lookup Parameters

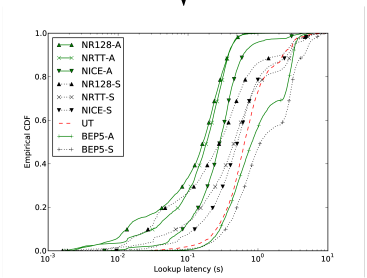
- **Standard** lookup module
  - Same as µTorrent
- **Aggressive** lookup module
  - More parallel queries
  - [Stutzbach-06] [Steiner-09]

Routing Table Management

- Handle **connectivity issues (quarantine)**
- **Continuous** refresh
- Prefer **low-latency** neighbors
- Enlarge **most frequently used** buckets

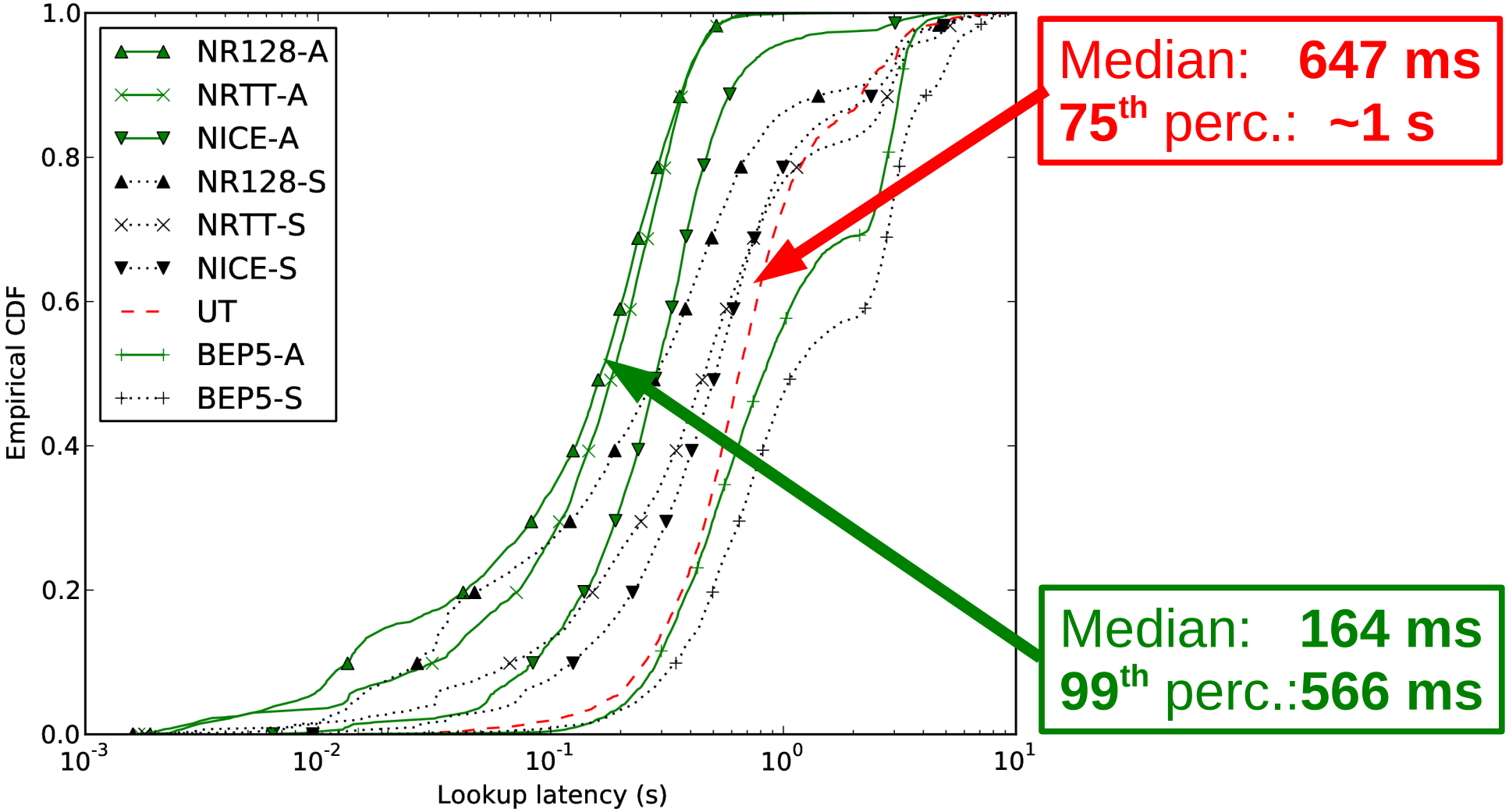
Performance comparative





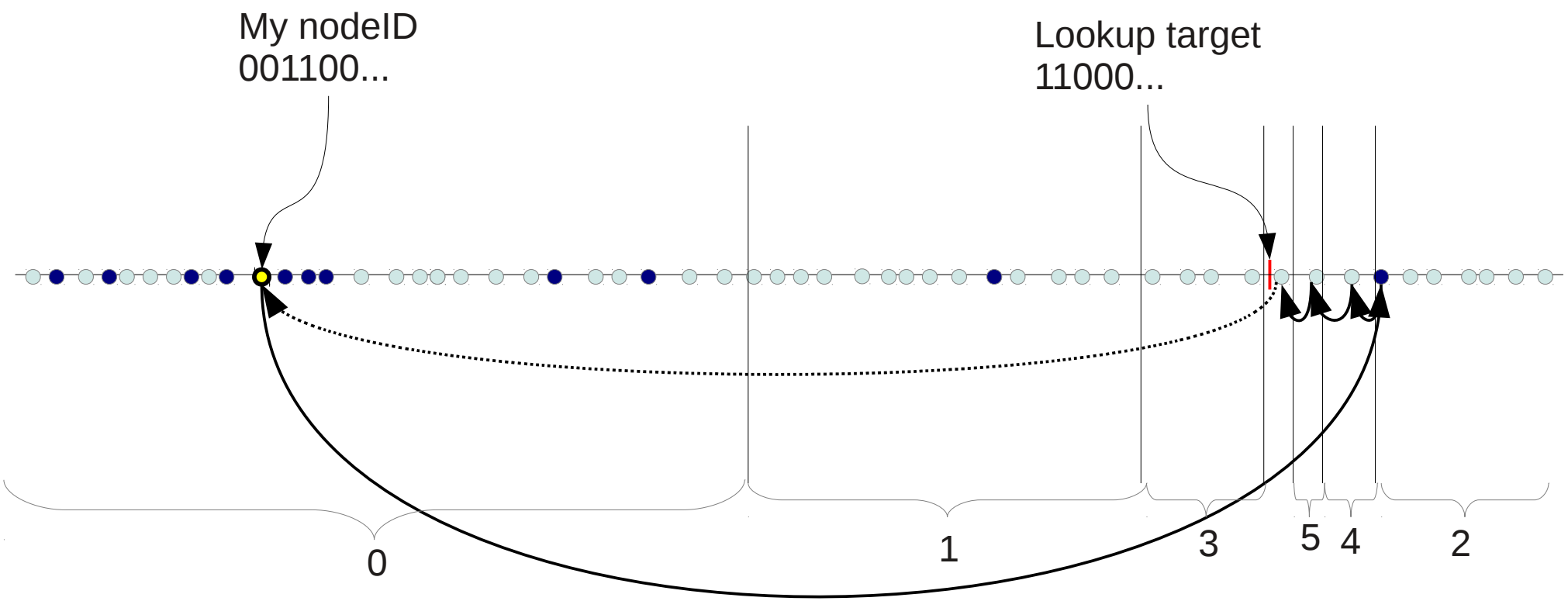
- µTorrent + 8 nodes
- 3078 lookups/node
- 80+ hours, 11+ GB
- 4.4 M unique IPs
- **Very consistent results**

Lookup Latency

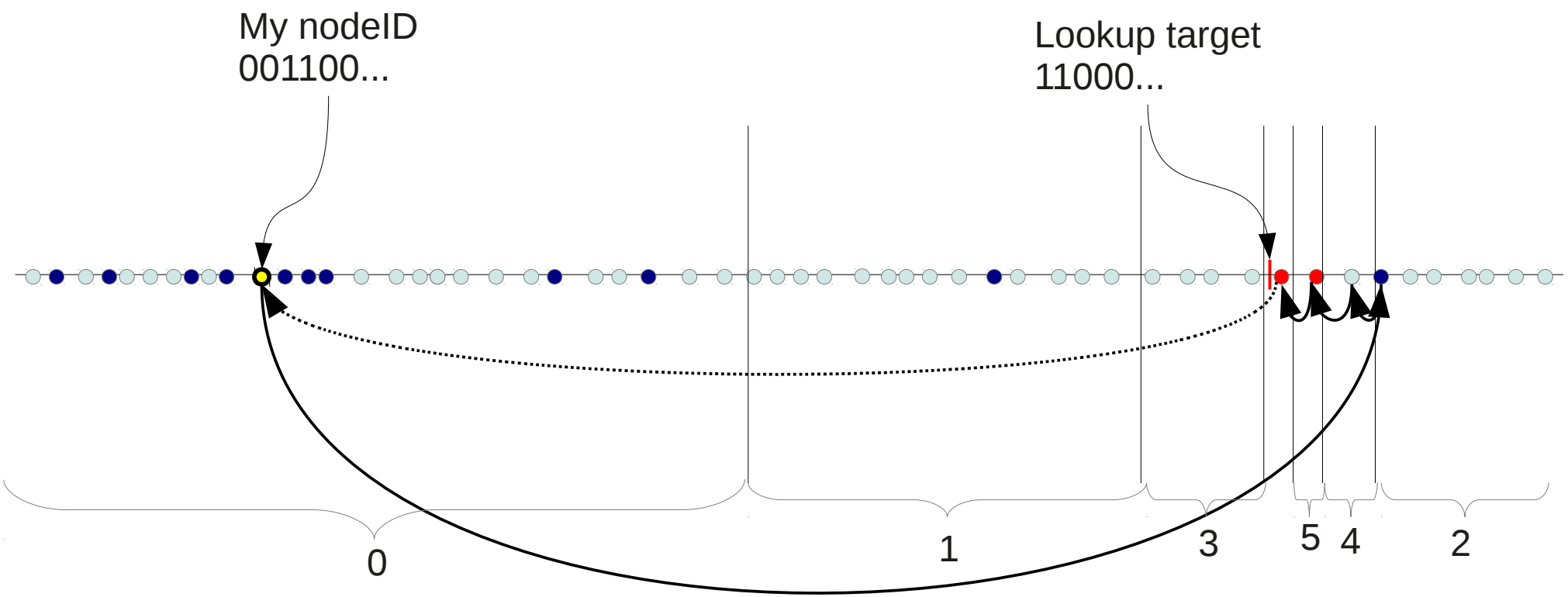


Scalability & Locality

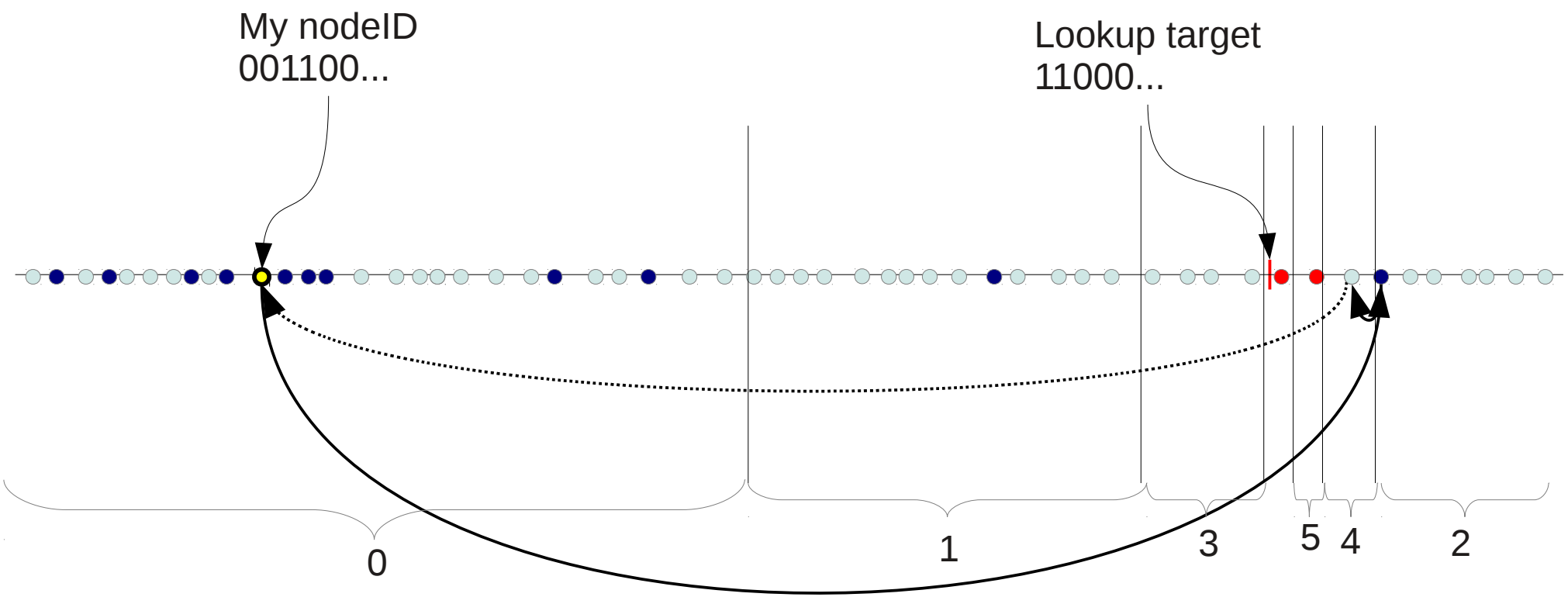
Scalability and Locality



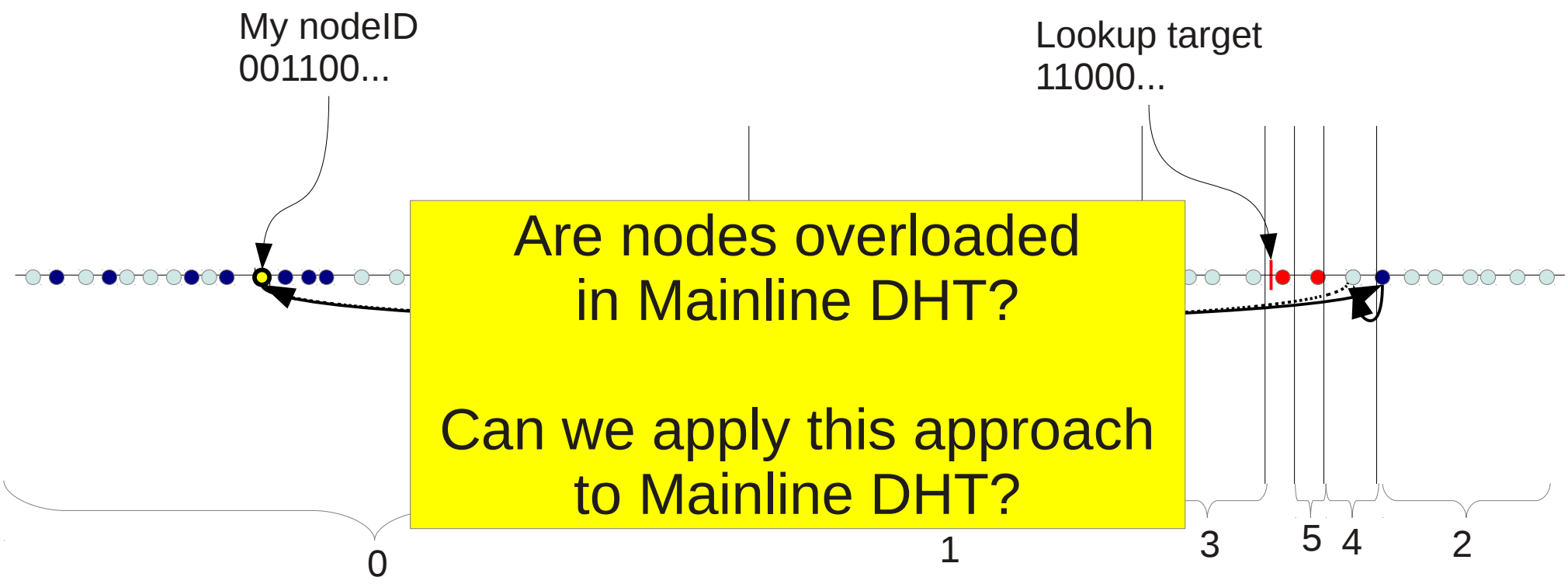
Scalability and Locality



Scalability and Locality



Scalability and Locality



</main results>

<conclusion>

Contributions

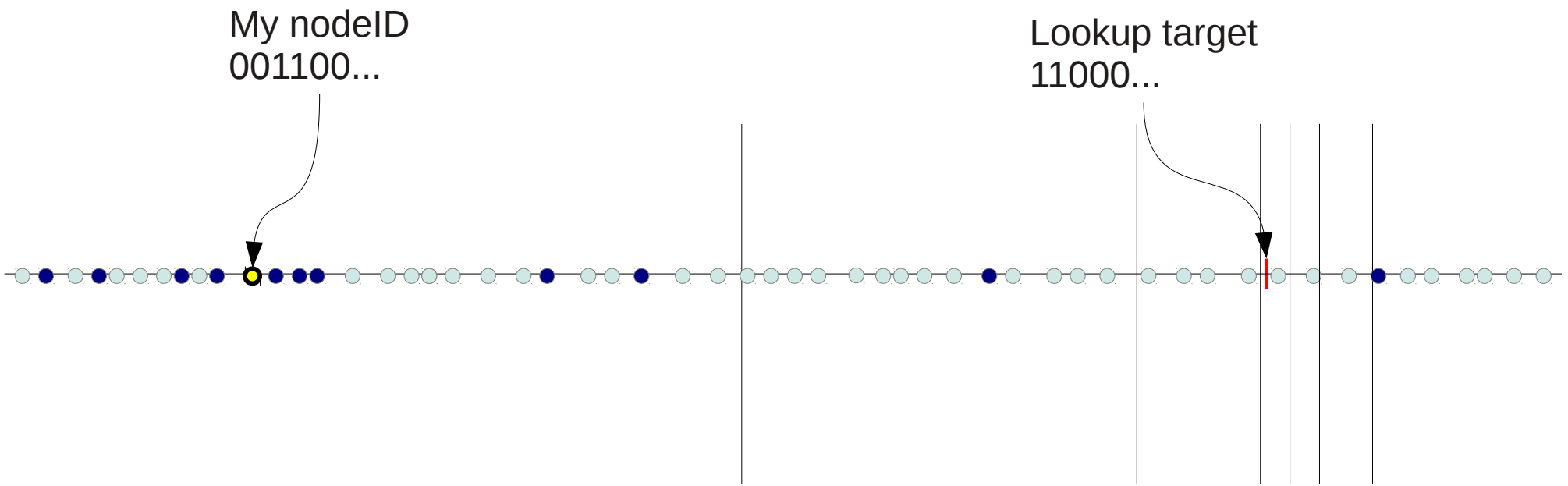
- **Connectivity properties**
- **Sub-second lookups**
- Scalability and Locality
- Source code

Future Work

- Integrate scalability and locality into MDHT
- Privacy and security

Future Work

- Integrate scalability and locality into MDHT
- Privacy and security



</conclusion>

ROYAL INSTITUTE
OF TECHNOLOGY

Thank you!

http://people.kth.se/~rauljc/lic/