



# Koala Protocol

- **A flat, lazy and topology-aware protocol for decentralized clouds**

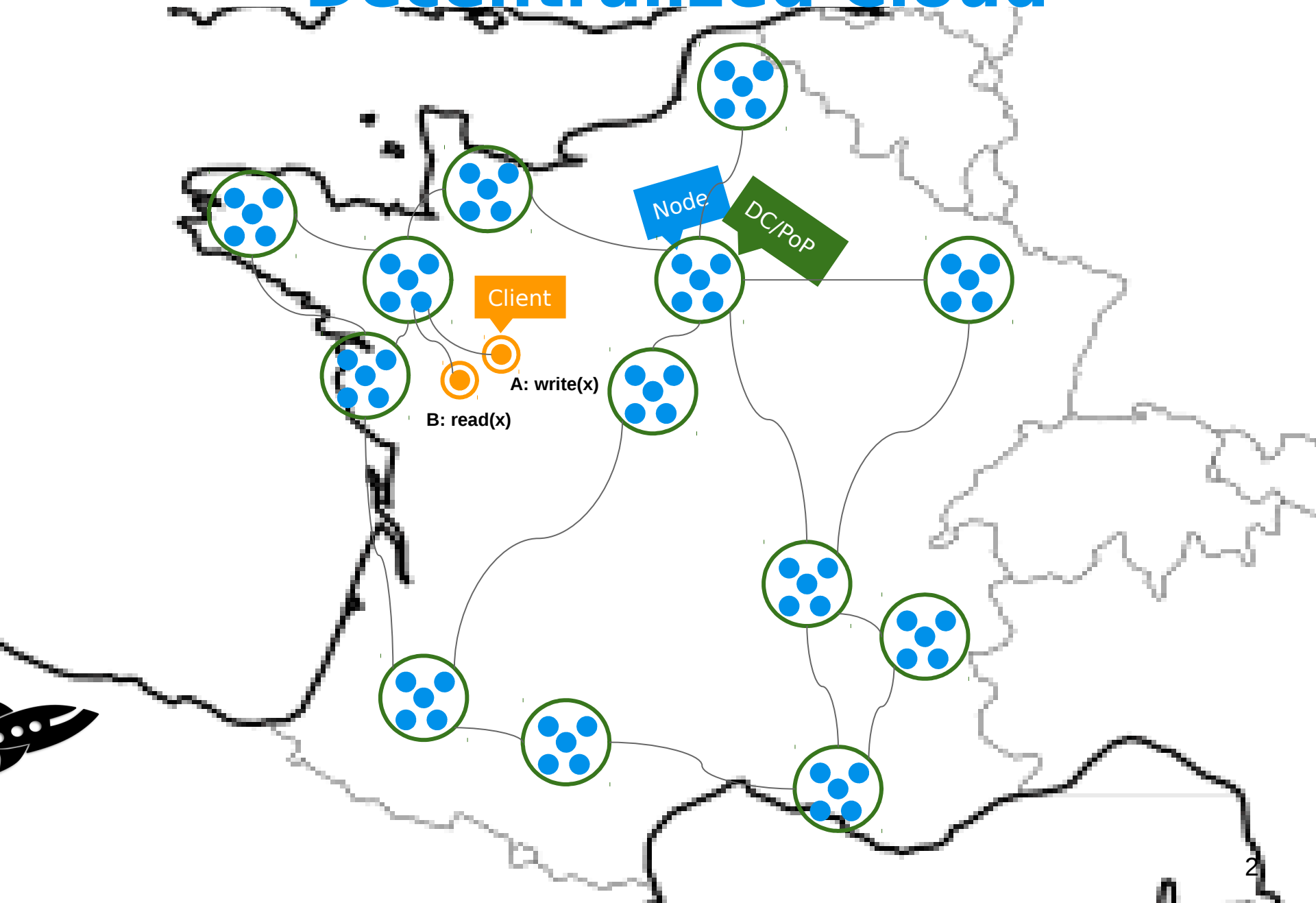
**Genc Tato**



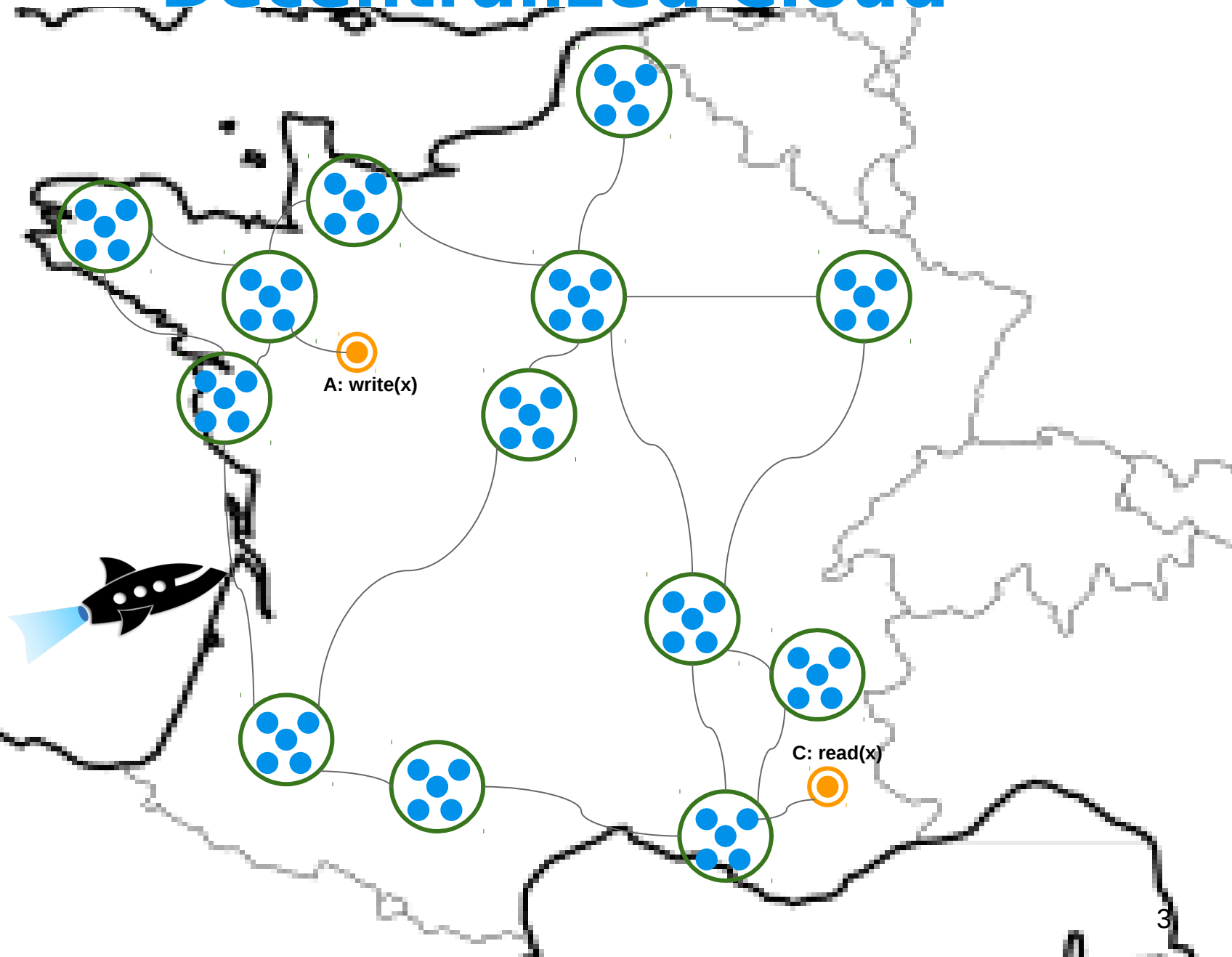
Supervised by:

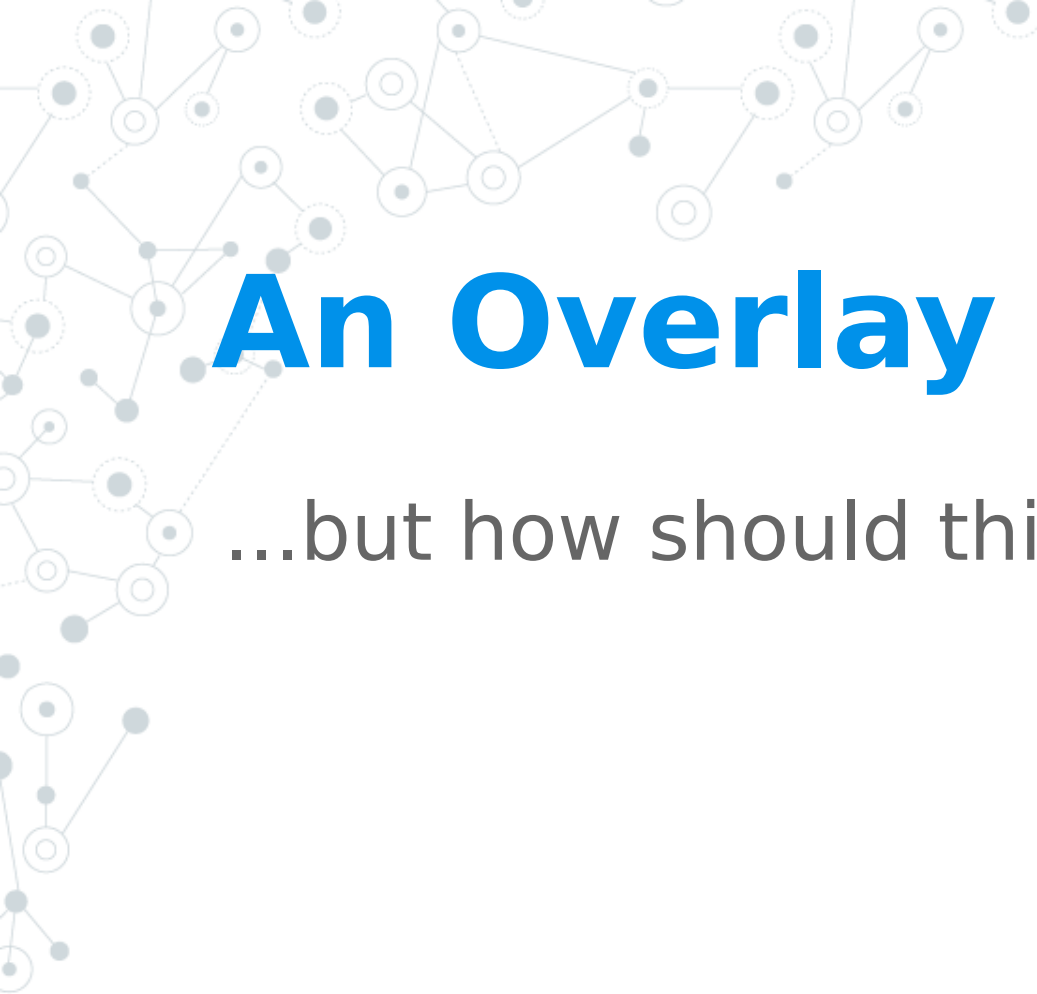
**Cédric Tedeschi**  
**Marin Bertier**

# Decentralized Cloud



# Decentralized Cloud



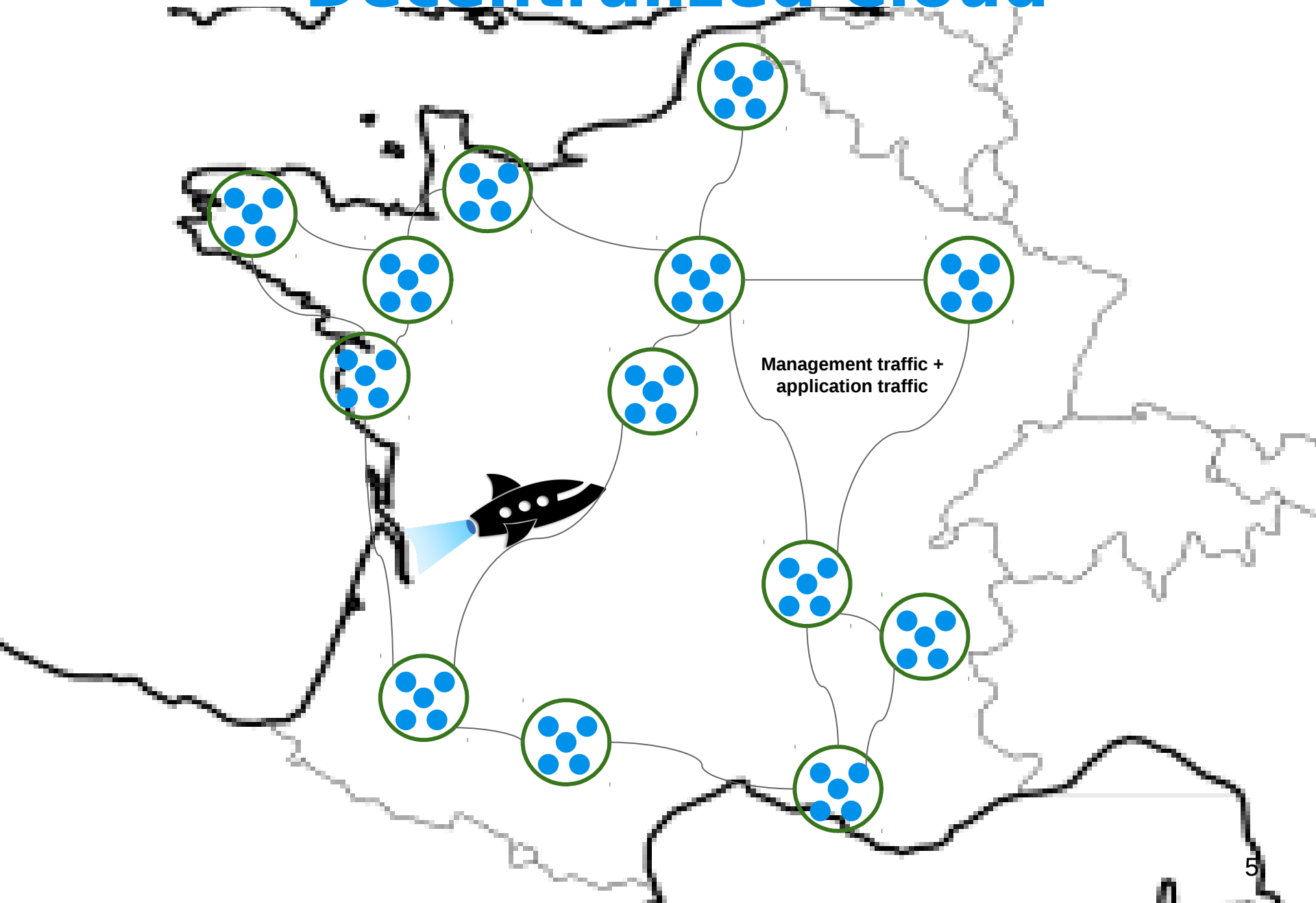


# An Overlay Network

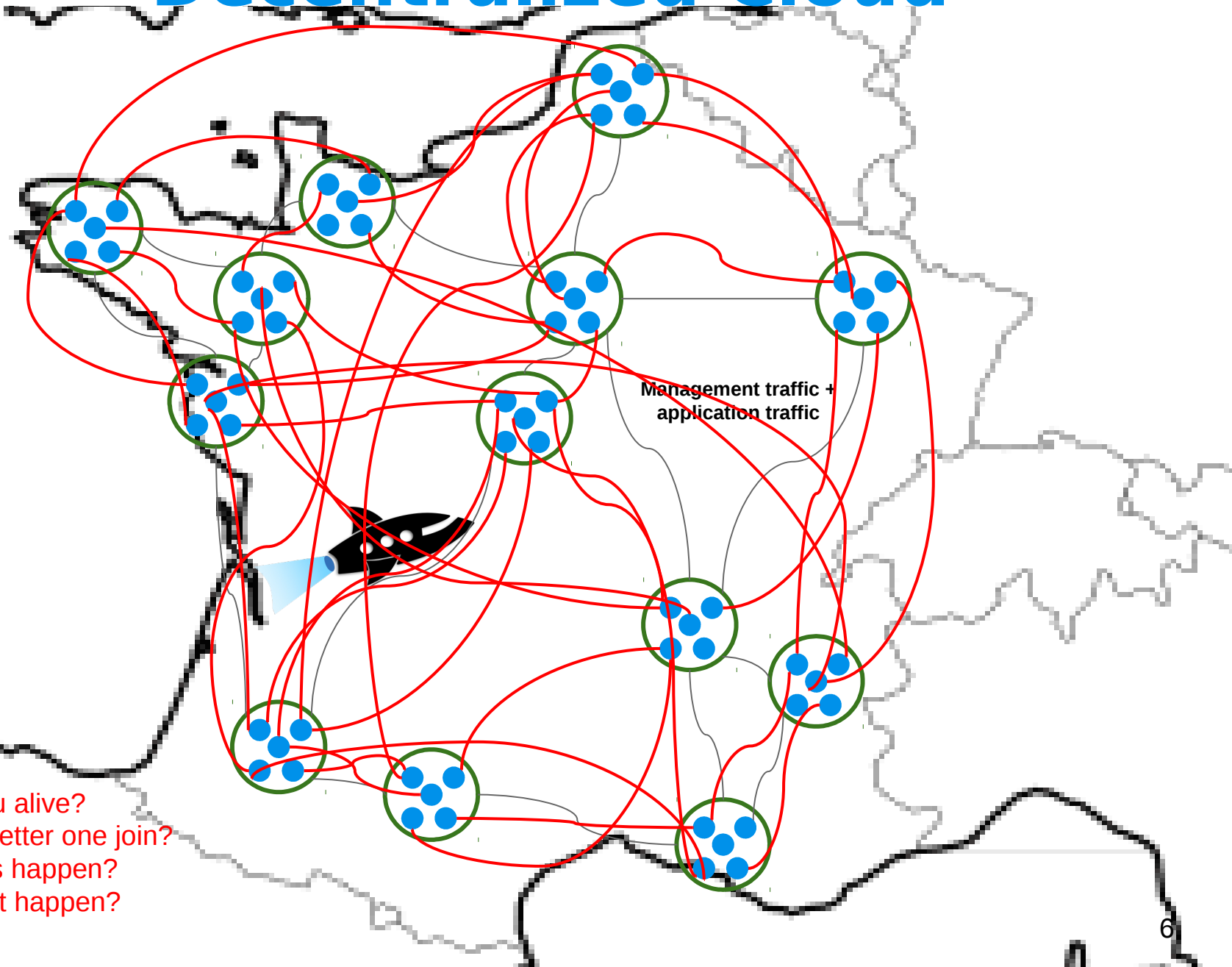
...but how should this overlay be?



# Decentralized Cloud



# Decentralized Cloud



Are you alive?  
Did a better one join?  
Did this happen?  
Did that happen?

A decorative background featuring a network diagram with nodes and connections. The nodes are represented by circles of varying sizes, some solid and some hollow, connected by thin lines. The diagram is positioned in the top-left and bottom-right corners of the slide.

# An Overlay Network

...but how should this overlay be?

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and edges. The nodes are represented by circles of varying sizes, some with concentric rings, and the edges are thin lines connecting them. The overall style is light gray and minimalist.

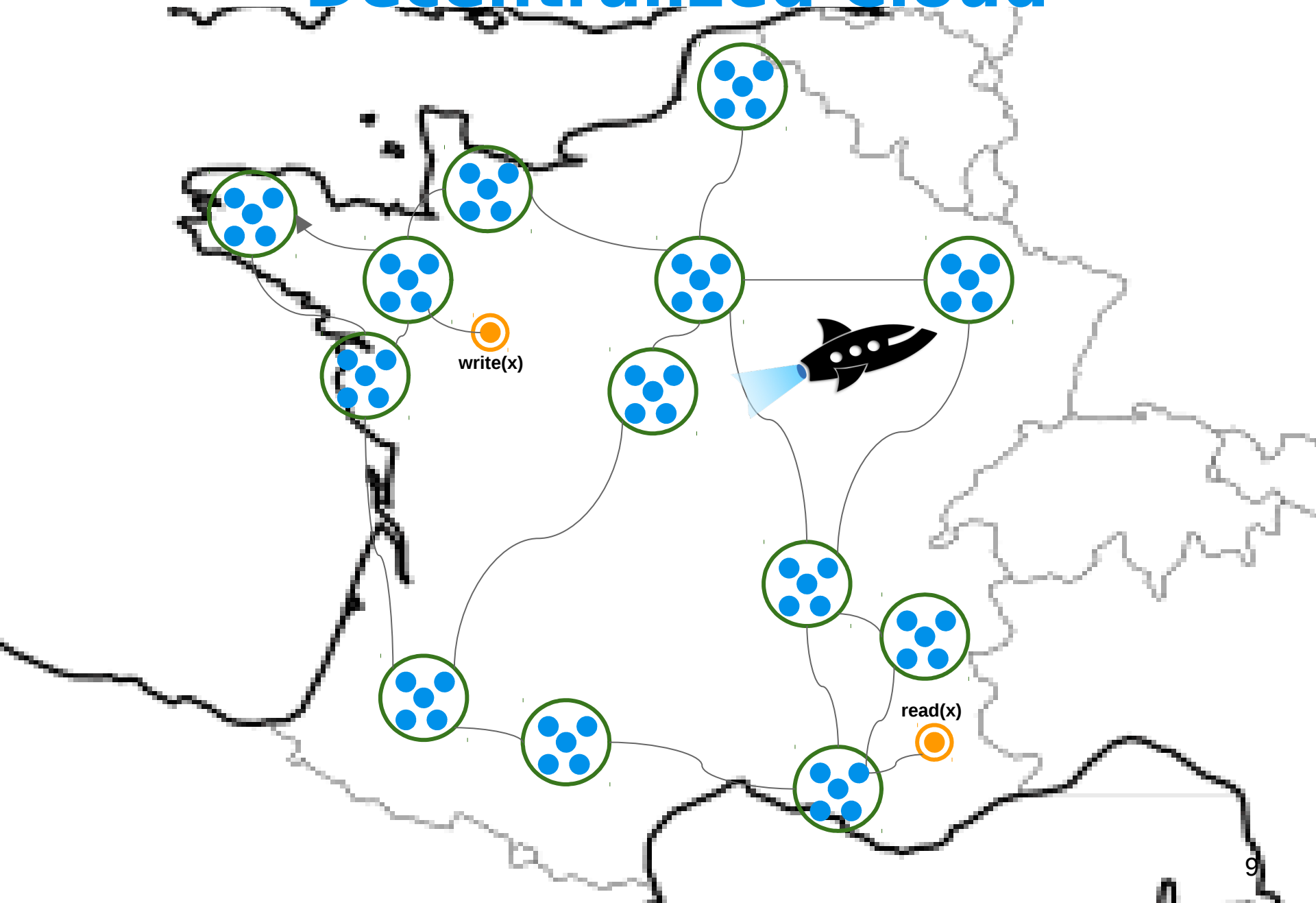
# An Overlay Network

...but how should this overlay be?

**1.Lazy (update overlay only when needed)**

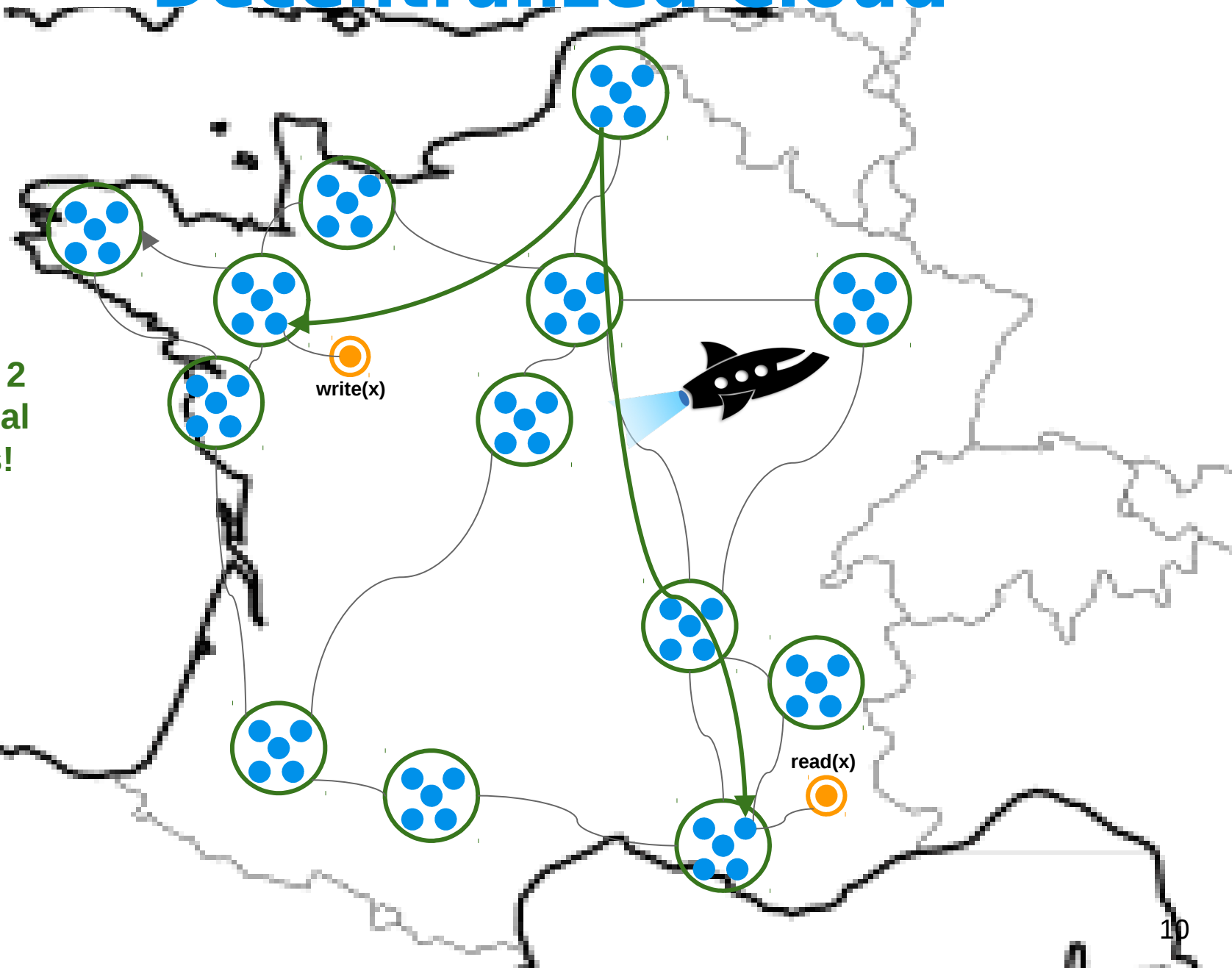


# Decentralized Cloud

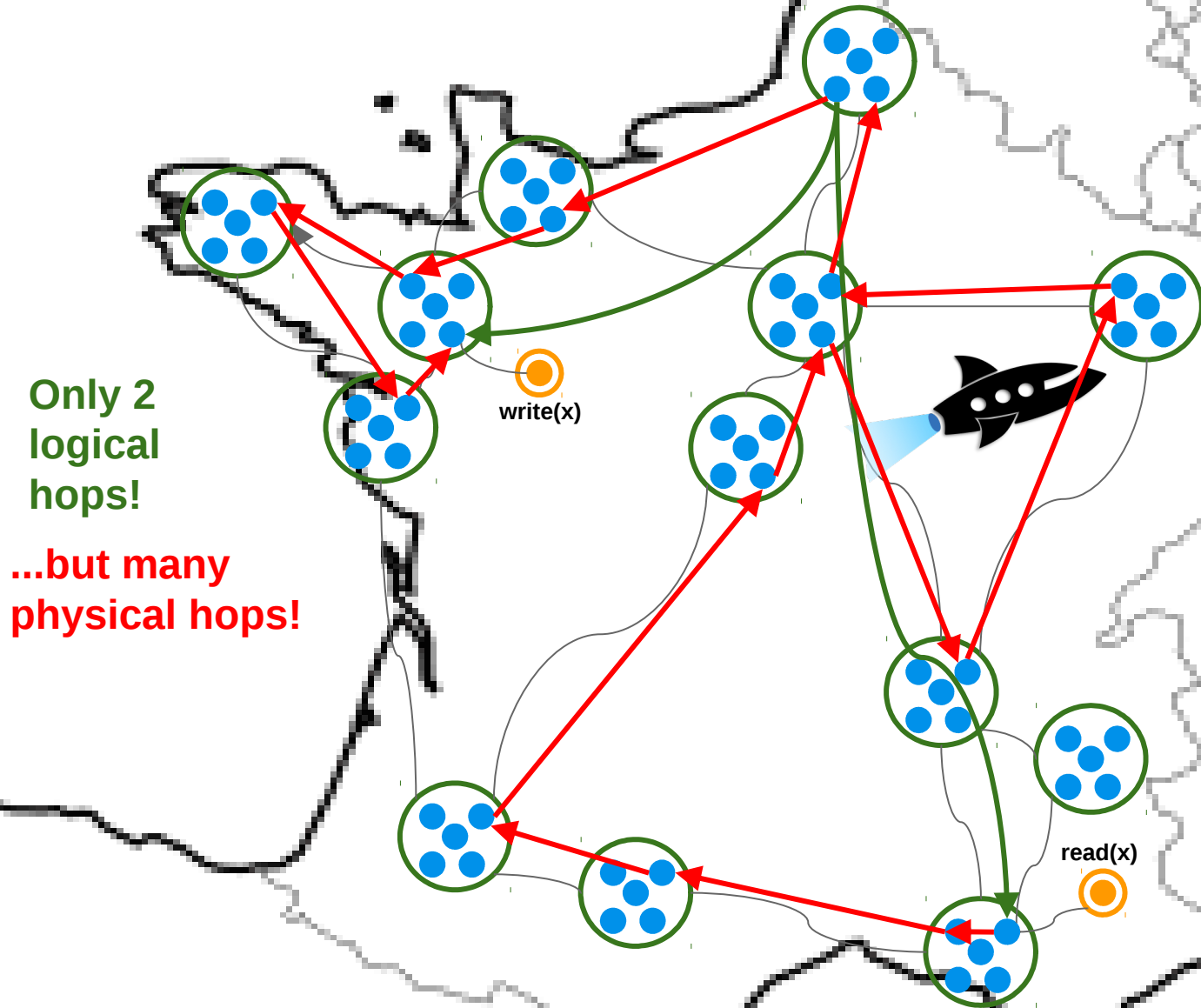


# Decentralized Cloud

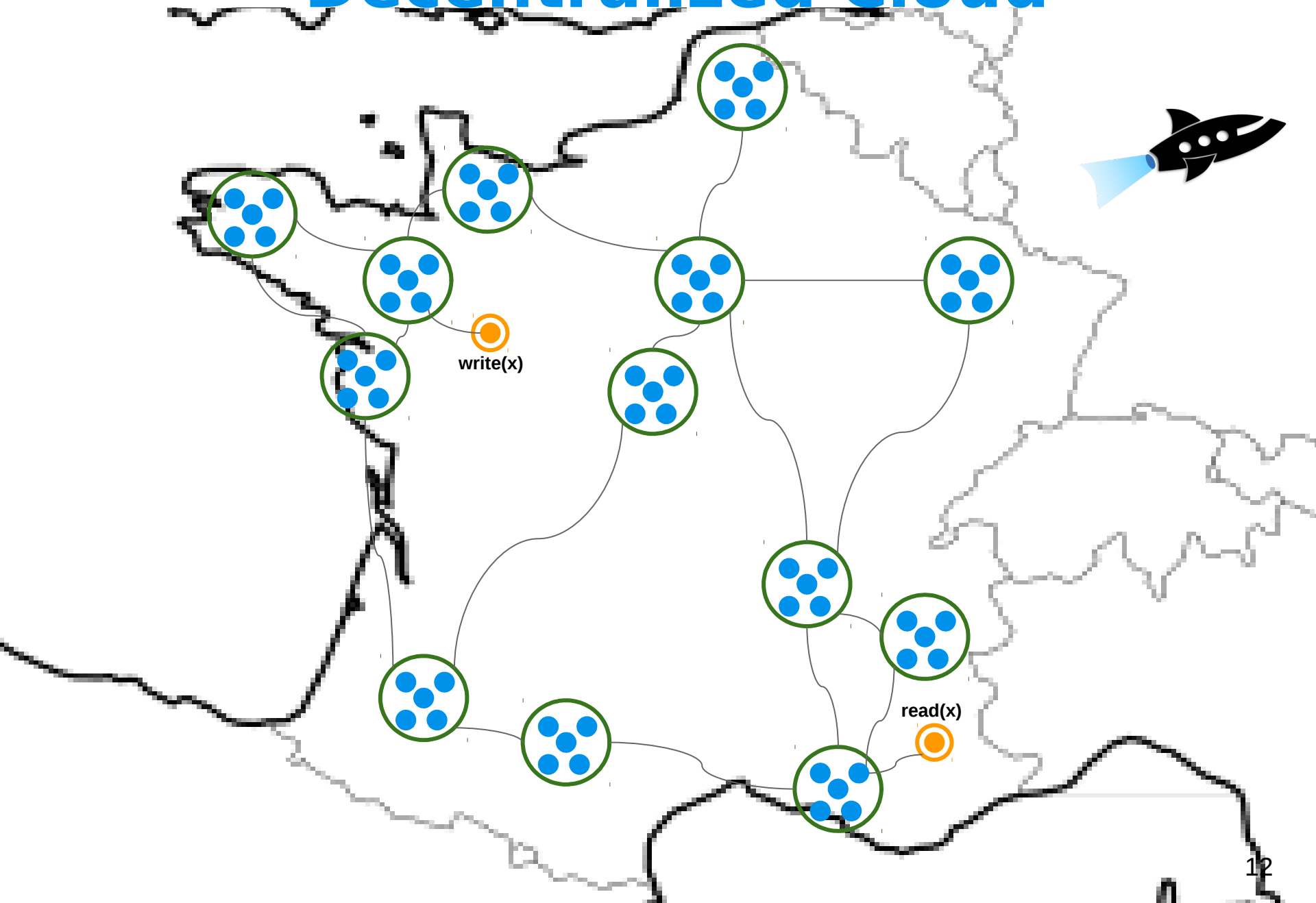
Only 2  
logical  
hops!



# Decentralized Cloud



# Decentralized Cloud

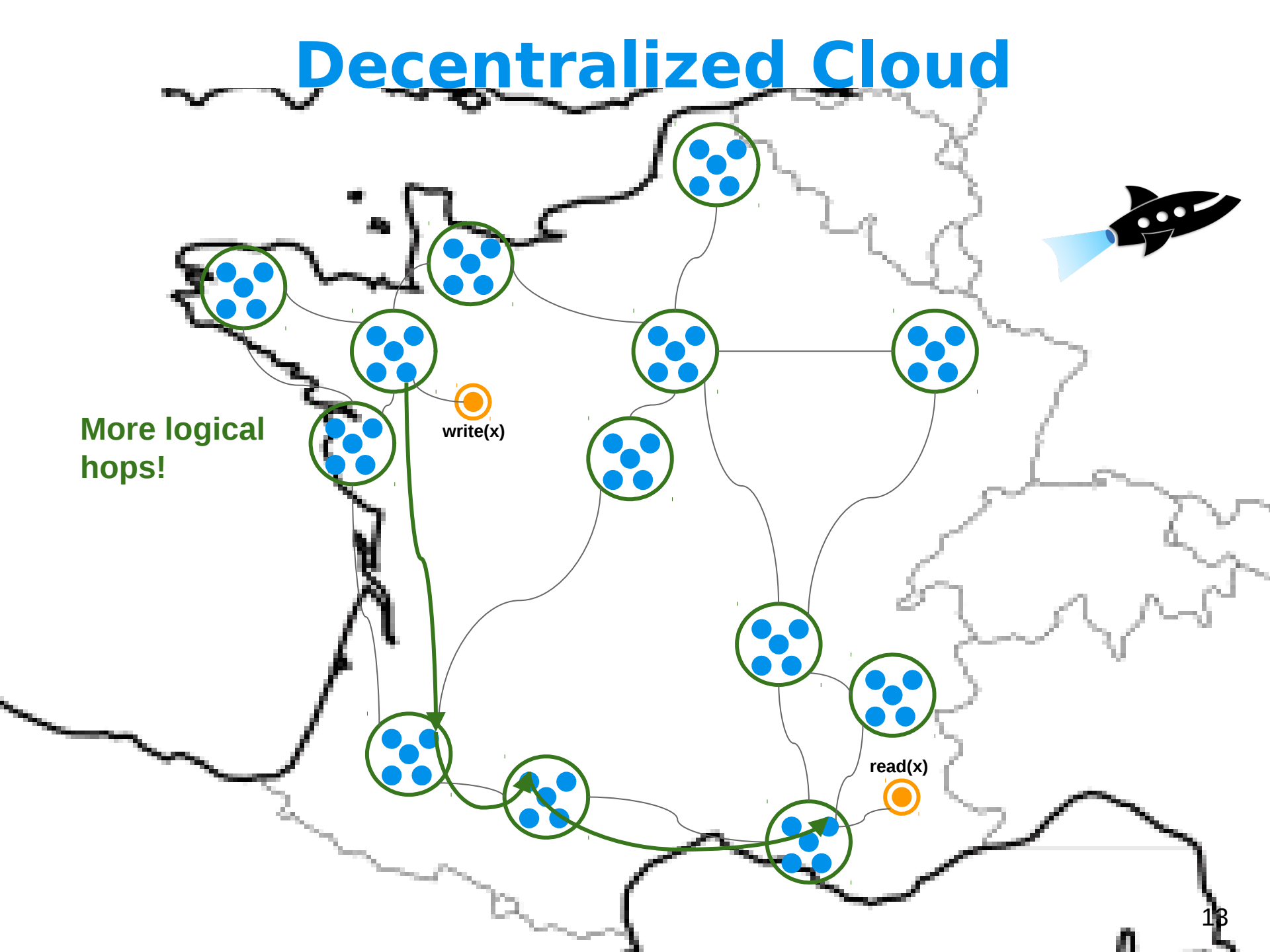


# Decentralized Cloud

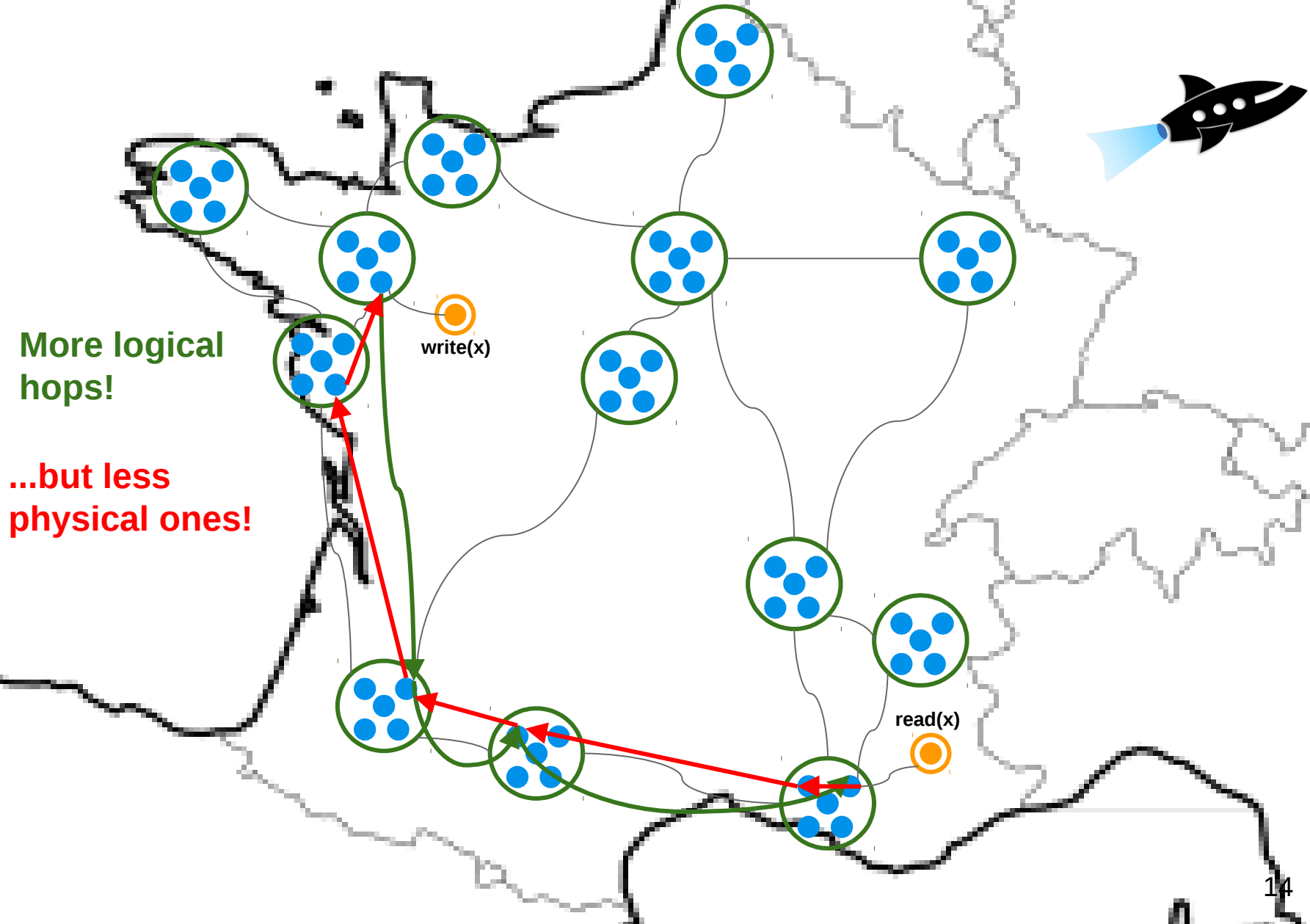
More logical  
hops!

write(x)

read(x)



# Decentralized Cloud



A decorative background network diagram consisting of various nodes (circles) and connecting lines (edges). Some nodes are solid grey, some are hollow white, and some are dashed grey. The connections are a mix of solid and dashed lines, creating a complex web-like structure.

# An Overlay Network

...but how should this overlay be?

1.Lazy (update overlay only when needed)



# An Overlay Network

...but how should this overlay be?

1. Lazy (update overlay only when needed)

**2. Locality-aware**



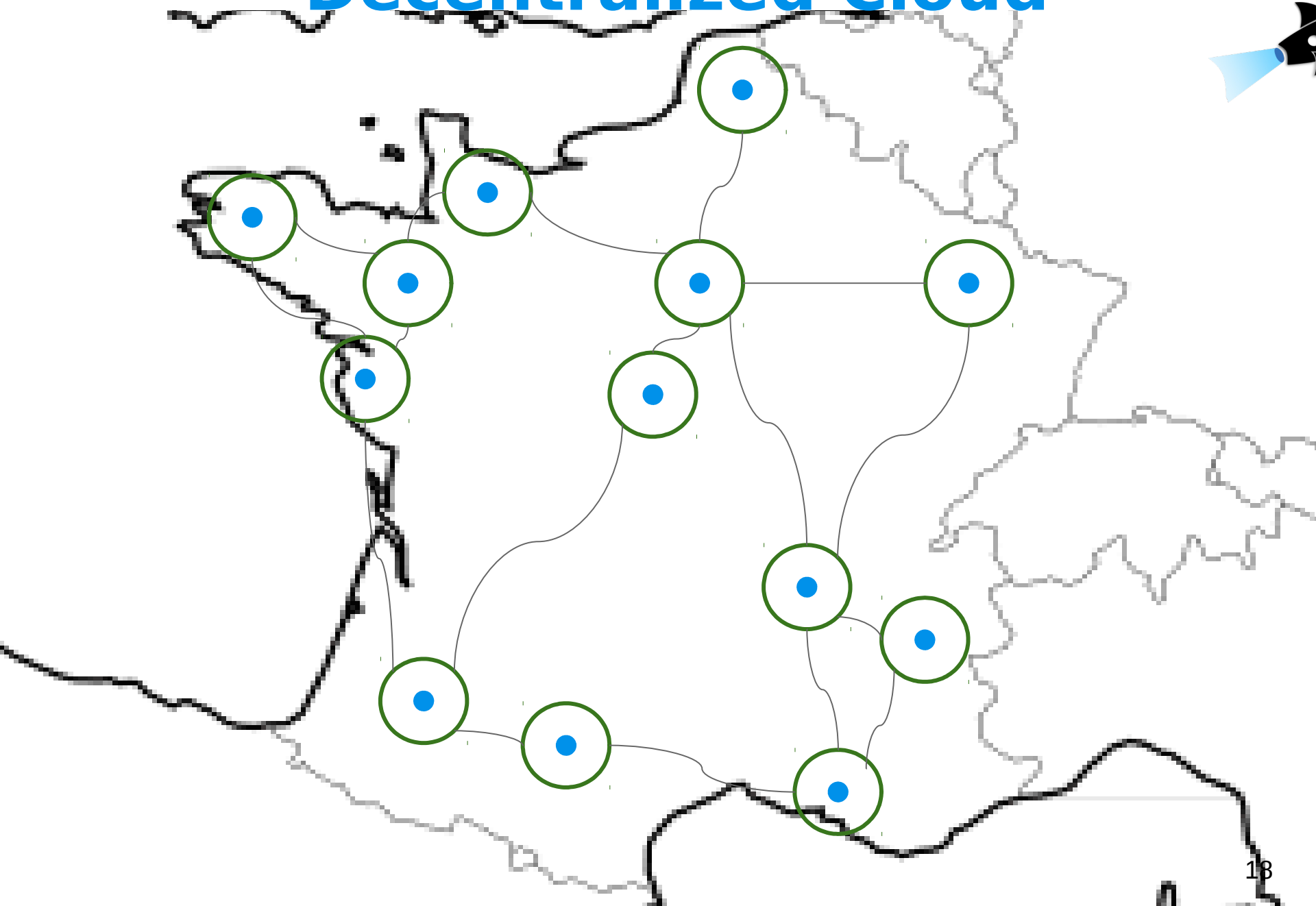
# How to?

1. How do we implement laziness?
2. How do we integrate locality-awareness?

First, focus on communication between different PoPs.

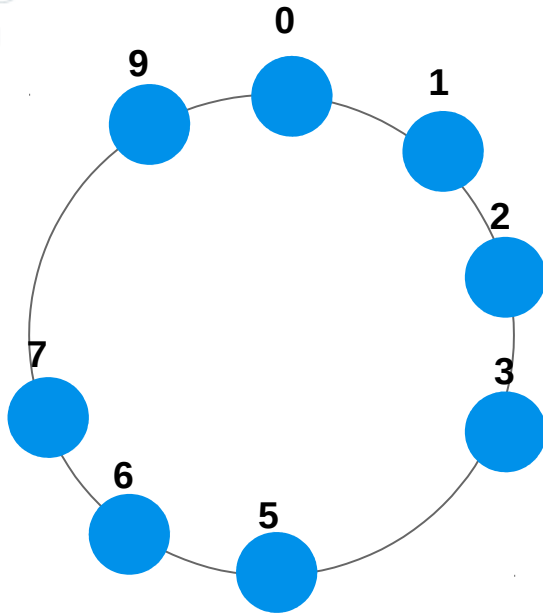
(nodes in same PoPs is discussed later)

# Decentralized Cloud



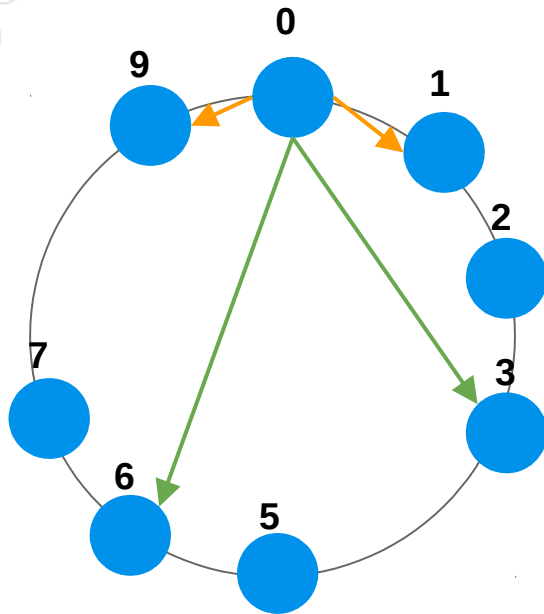
# The basics

Nodes are organized in a ring and are identified by a circular id.



# The basics

Nodes are organized in a ring and are identified by a circular id.



Routing table of node 0

| ID | IP      | RTT | Dist | IID |
|----|---------|-----|------|-----|
| 9  | a.a.a.a | 50  | 1    | -   |
| 1  | b.b.b.b | 150 | 1    | -   |
| 3  | x.x.x.x | 125 | 3    | 4   |
| 6  | y.y.y.y | 250 | 4    | 7   |

Neighbors

Long links

# Ideal long links

Still want to be a  $O(\log N)$  hops protocol

# Ideal long links

A decorative network diagram in the top right corner of the slide. It features a complex web of interconnected nodes and edges. Some nodes are represented by solid grey circles, while others are hollow circles with a dot inside. The edges are thin grey lines, some of which are solid and others dashed, creating a sense of depth and connectivity.

Still want to be a  **$O(\log N)$**  hops protocol

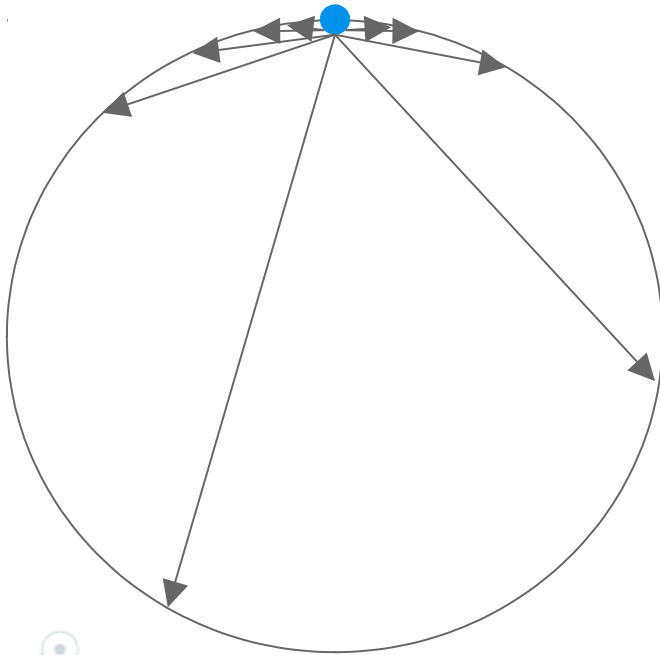
Tell me who your long links are, I will tell you how efficient your routing is!



# Ideal long links

Still want to be a  $O(\log N)$  hops protocol

Tell me who your long links are, I will tell you how efficient your routing is!



Continuous Kleinberg distribution:

$$p(d) = 1/(d \cdot \ln N)$$

Where:

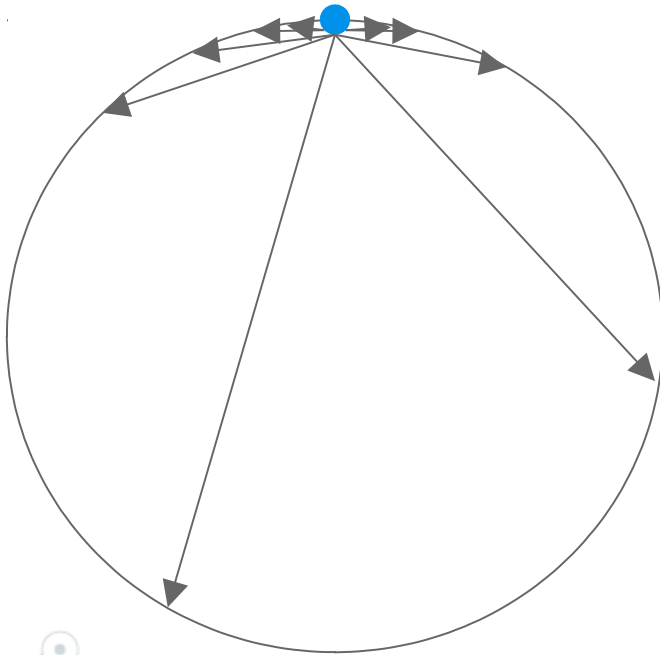
$d$  = logical distance,

$N$  = total nr. of nodes

# Ideal long links

Still want to be a  $O(\log N)$  hops protocol

Tell me who your long links are, I will tell you how efficient your routing is!



Continuous Kleinberg distribution:

$$p(d) = 1/(d \cdot \ln N)$$

Where:

$d$  = logical distance,  
 $N$  = total nr. of nodes

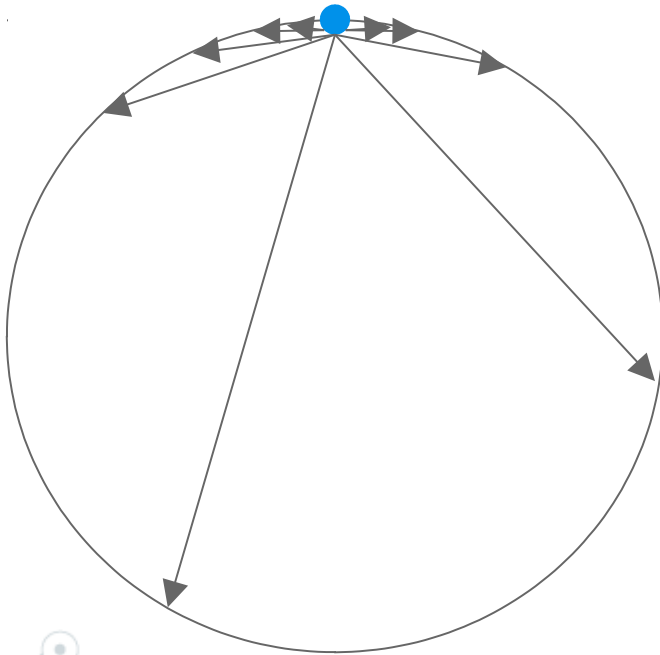
Generate IDs using  $p(d)$  -> **Ideal IDs**



# Ideal long links

Generating IDs does not mean contacting, it might not exist.

Lazy: do not search, wait to learn



Continuous Kleinberg distribution:

$$p(d) = 1/(d * \ln N)$$

Where:

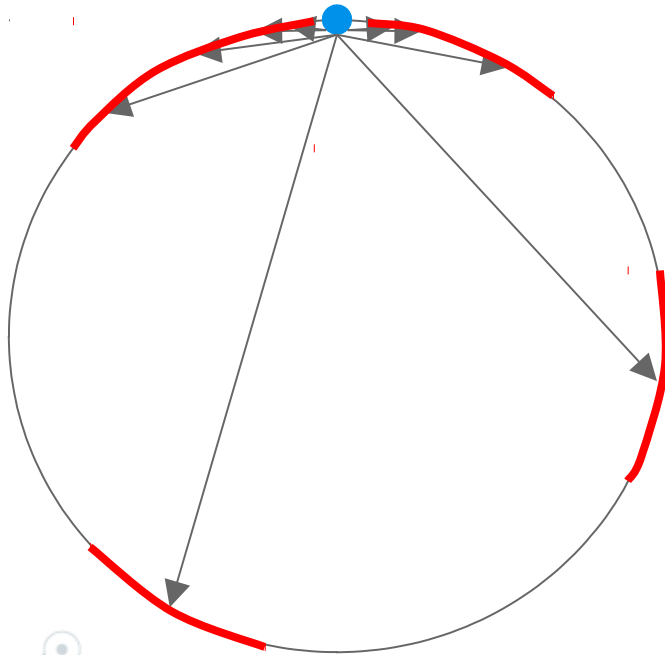
$d$  = logical distance,

$N$  = total nr. of nodes

# Ideal long links

Generating IDs does not mean contacting, it might not exist.

Lazy: do not search, wait to learn



Continuous Kleinberg distribution:

$$p(d) = 1/(d * \ln N)$$

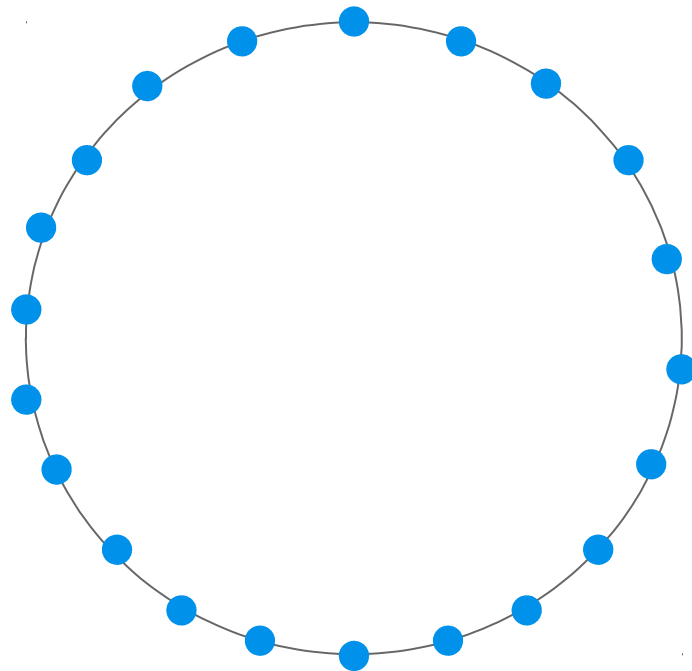
Where:

$d$  = logical distance,  
 $N$  = total nr. of nodes

Close to ideal is still ideal.

# Laziness: Piggybacking

Embed information about nodes within the message

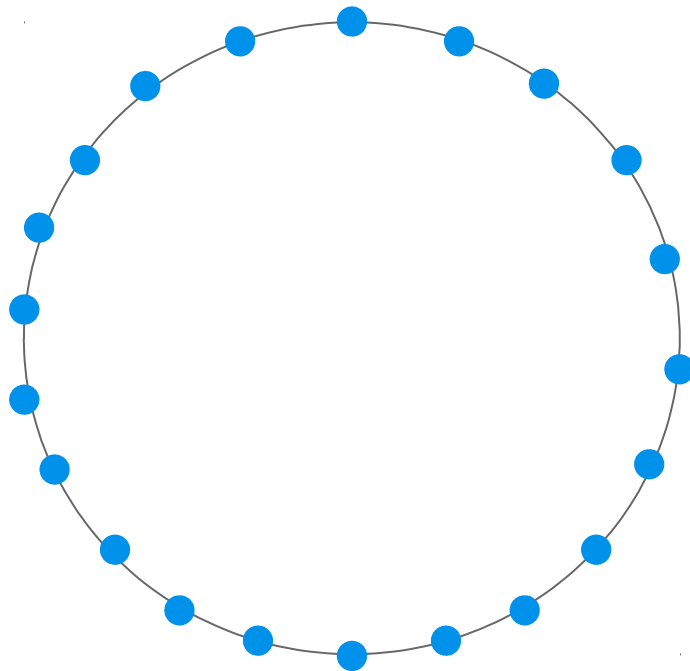


2 sources:



# Laziness: Piggybacking

Embed information about nodes within the message



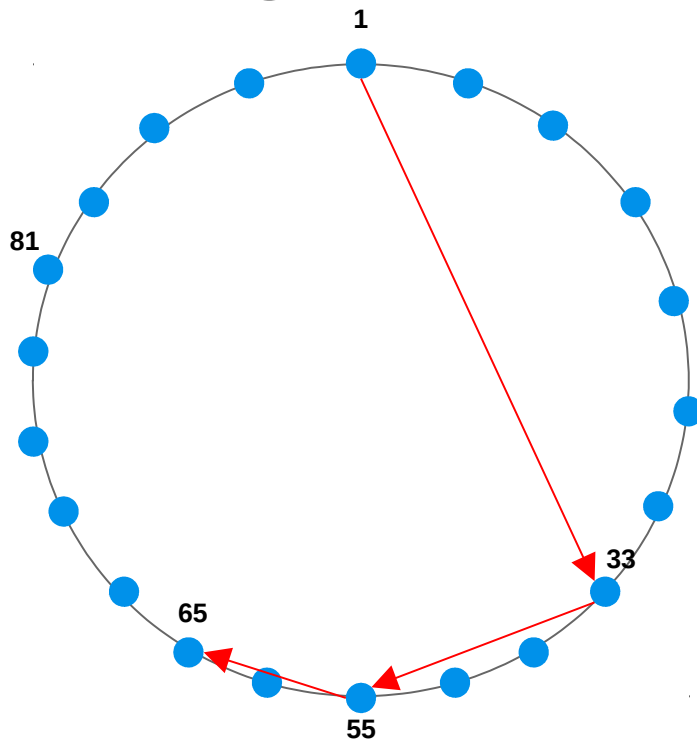
2 sources:

1. Nodes in the path



# Laziness: Piggybacking

Embed information about nodes within the message



2 sources:

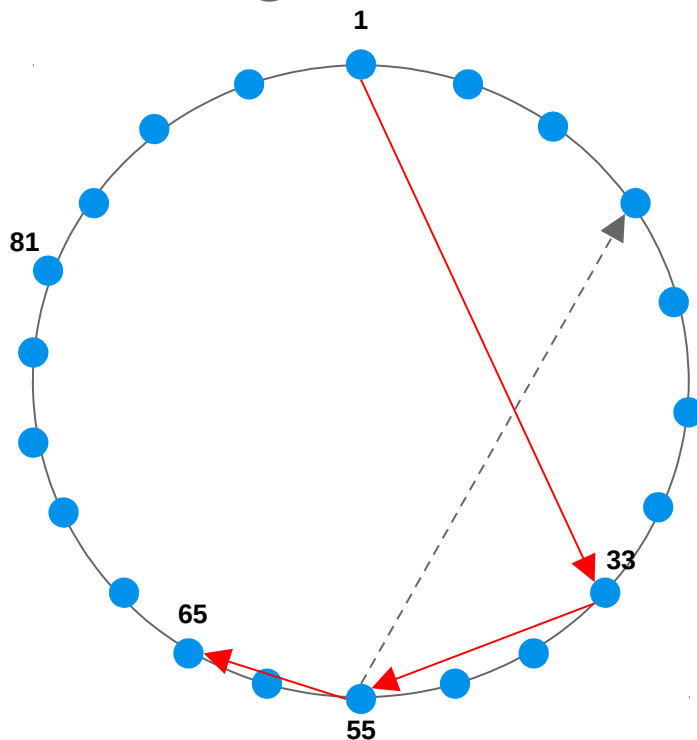
1. Nodes in the path

Path: 1, 33, 55, 65



# Laziness: Piggybacking

Embed information about nodes within the message



2 sources:

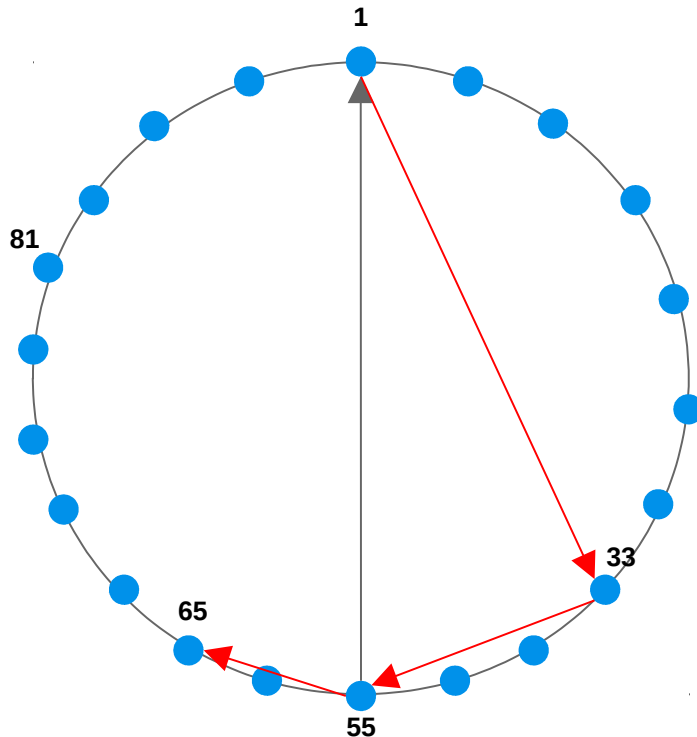
1. Nodes in the path

Path: 1, 33, 55, 65



# Laziness: Piggybacking

Embed information about nodes within the message



2 sources:

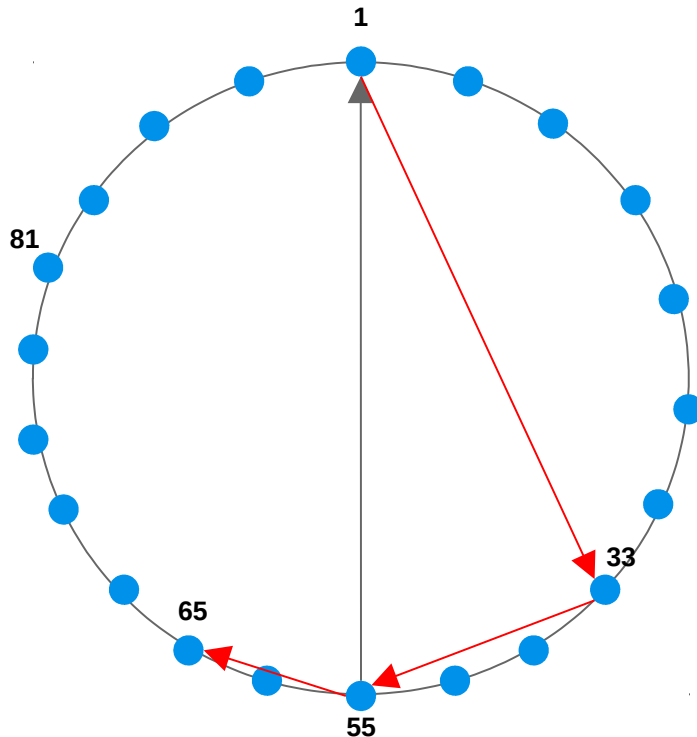
1. Nodes in the path

Path: 1, 33, 55, 65



# Laziness: Piggybacking

Embed information about nodes within the message



2 sources:

1. Nodes in the path
2. Nodes in the routing table of the nodes in the path

Path: 1, 33, 55, 65





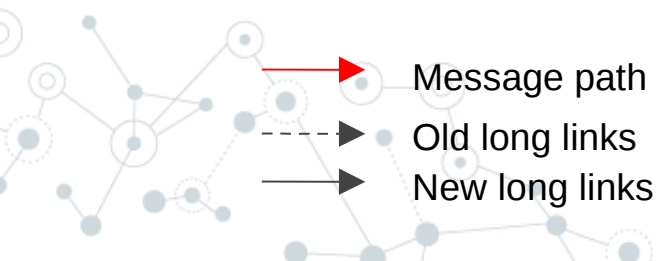
# backing

odes within the

es:

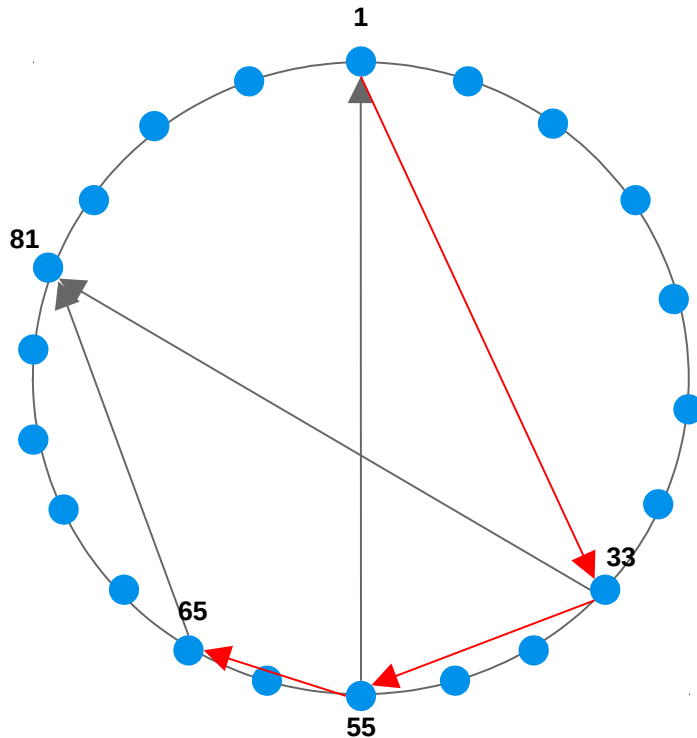
1. Nodes in the path
2. Nodes in the routing table of the nodes in the path

Path: 1, 33, 55, 65



# Laziness: Piggybacking

Embed information about nodes within the message



2 sources:

1. Nodes in the path
2. Nodes in the routing table of the nodes in the path

Path: 1, 33, 55, 65



# Locality-awareness

A decorative network graph in the top right corner, consisting of various sized circles (nodes) connected by thin lines (edges). Some nodes are solid grey, while others are hollow with a grey outline. The connections form a complex, interconnected web.

Find the cheapest logical path (latency-wise)



# Locality-awareness

A decorative network graph in the top right corner, featuring nodes of varying sizes (some solid grey, some hollow white) connected by thin grey lines, forming a complex web-like structure.

Find the cheapest logical path (latency-wise)

We can only choose the cheapest next hop!



# Locality-awareness

A decorative network diagram in the top right corner of the slide. It consists of numerous nodes, represented by circles of varying sizes and shades of gray, connected by thin, light gray lines. The nodes are arranged in a complex, interconnected pattern, suggesting a network topology.

Find the cheapest logical path (latency-wise)

We can only choose the cheapest next hop!

Keep right direction by reducing logical distance



# Locality-awareness

A decorative network graph in the top right corner, featuring various nodes (some solid, some hollow) connected by lines, representing a network structure.

Find the cheapest logical path (latency-wise)

We can only choose the cheapest next hop!

Keep right direction by reducing logical distance

A tradeoff between logical distance and latency



# Locality-awareness

Find the cheapest logical path (latency-wise)

We can only choose the cheapest next hop!

Keep right direction by reducing logical distance

A tradeoff between logical distance and latency

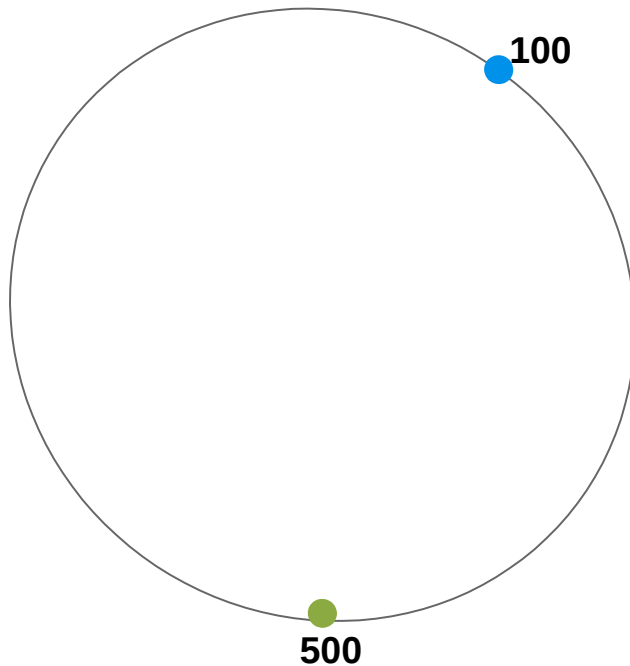
For each entry we calculate:

$$Q(re) = 1/(\alpha \times re.distance + (1-\alpha) \times norm(re.RTT))$$

# Locality-awareness: Routing

Route from 100 to 500

N=1000



Routing table of node 100

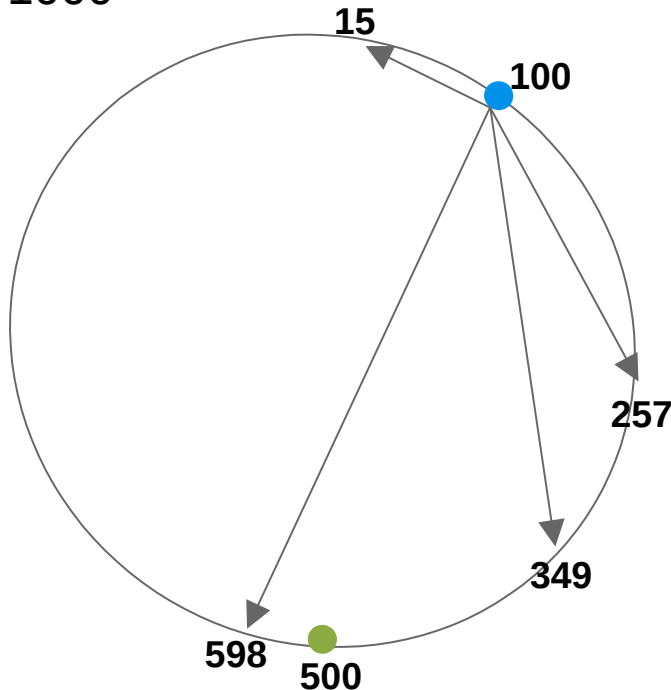
$$Q(re) = 1/(\alpha \times re.distance + (1-\alpha) \times norm(re.RTT))$$



# Locality-awareness: Routing

Route from 100 to 500

N=1000



Routing table of node 100

| ID  | RTT  | DistToDest |
|-----|------|------------|
| 15  | 340  | 485        |
| 257 | 105  | 243        |
| 349 | 194  | 151        |
| 598 | 1230 | 98         |

$$Q(\text{re}) = 1/(\alpha \times \text{re.distance} + (1-\alpha) \times \text{norm}(\text{re.RTT}))$$



# Evaluation

Three criteria:

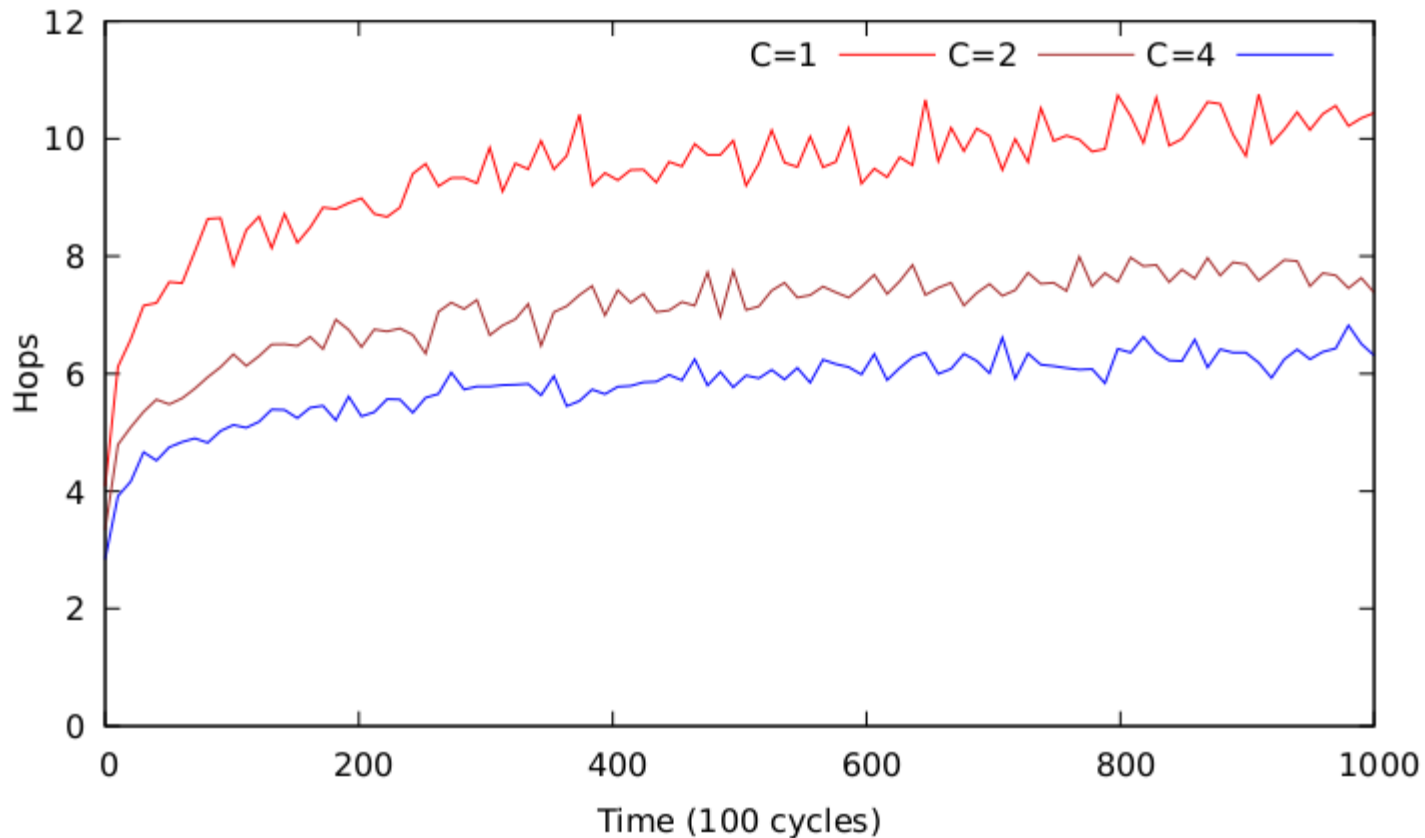
©Scalability

©Locality-awareness trade-off

©Resilience

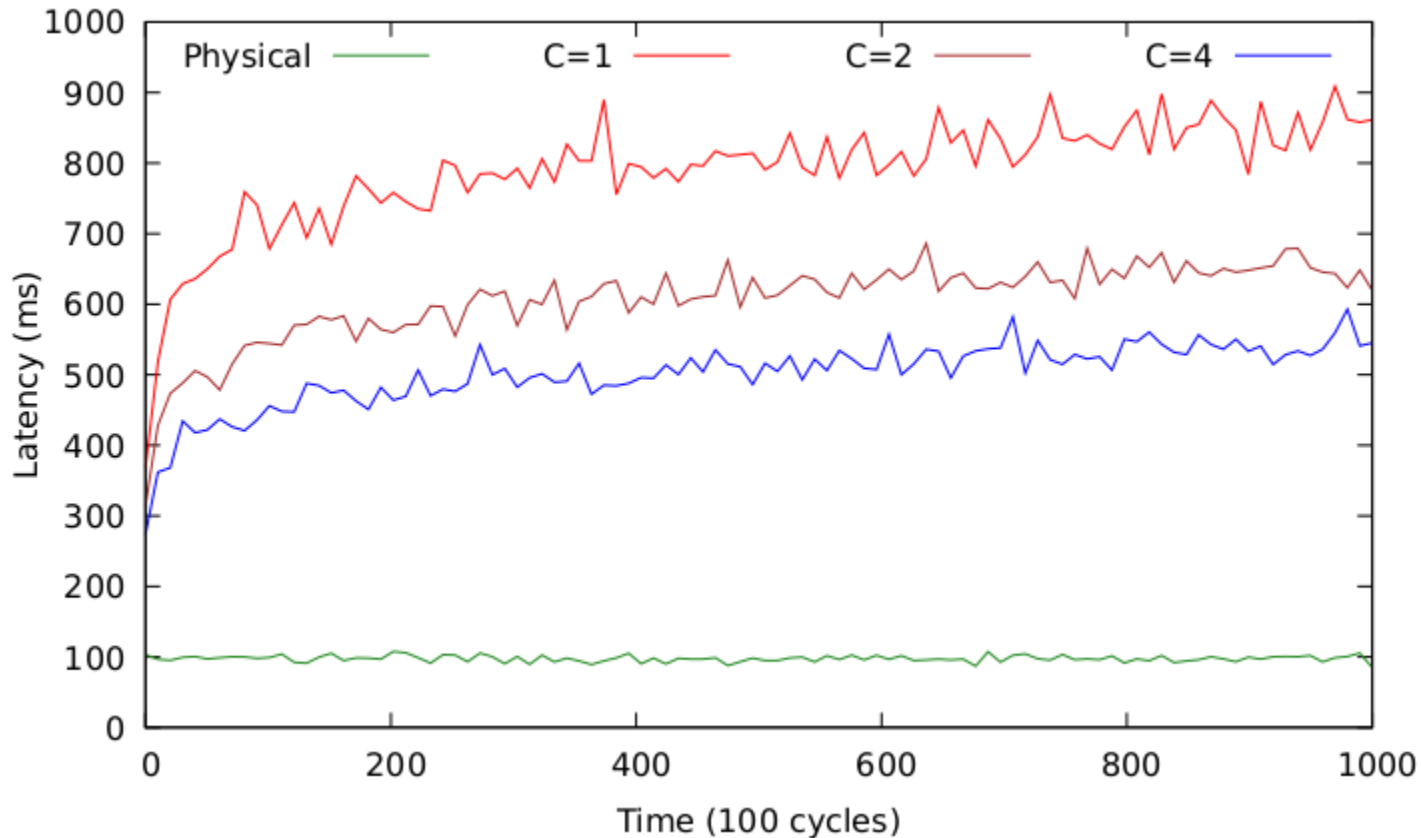
# Scalability (hops)

Add nodes each cycle until 100K  
# of long links =  $C * \log N$

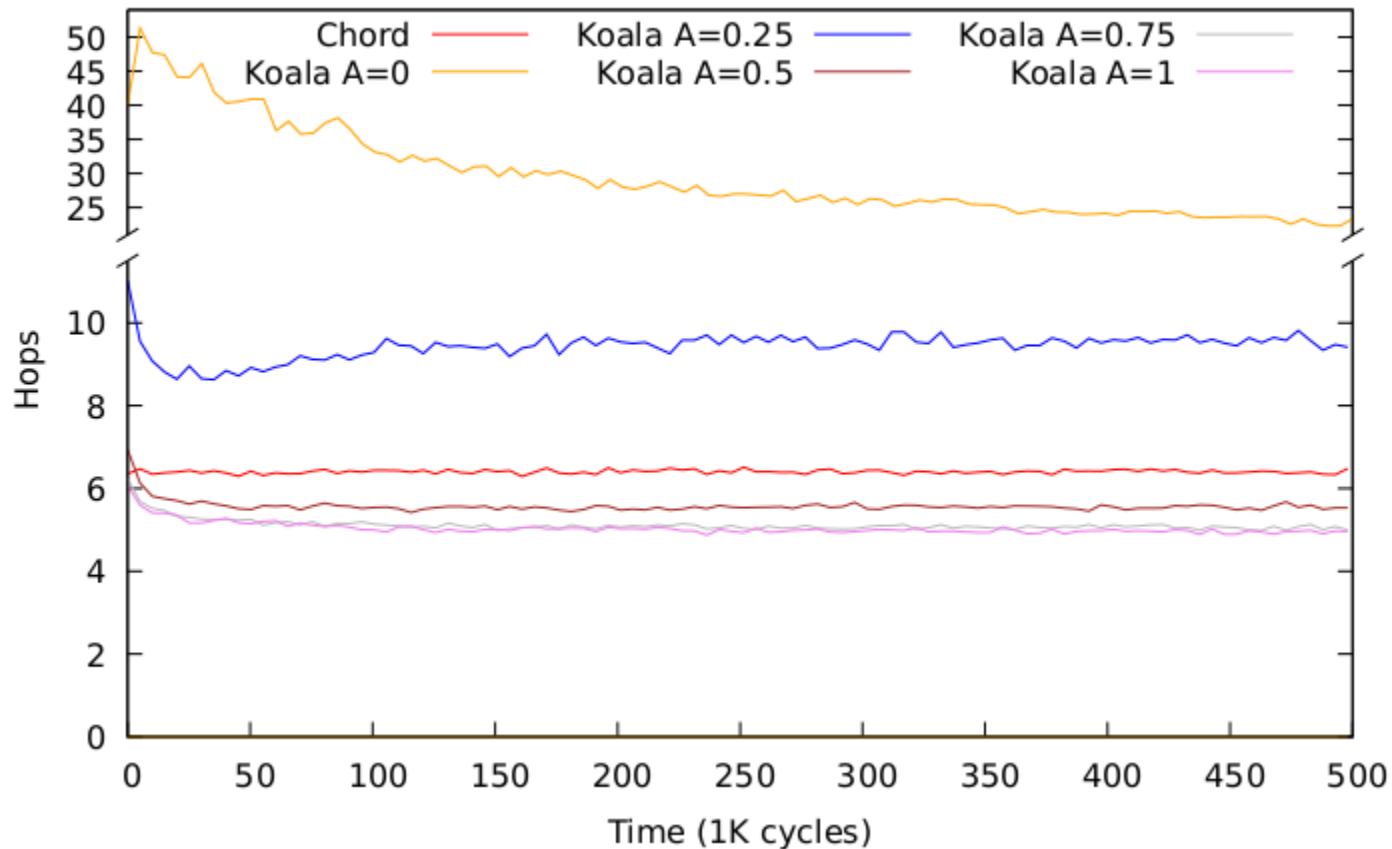
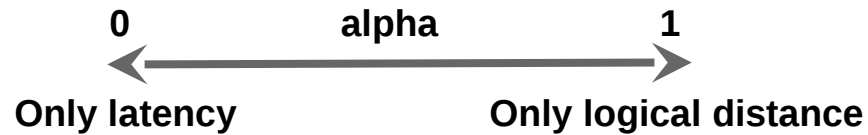


# Scalability (latency)

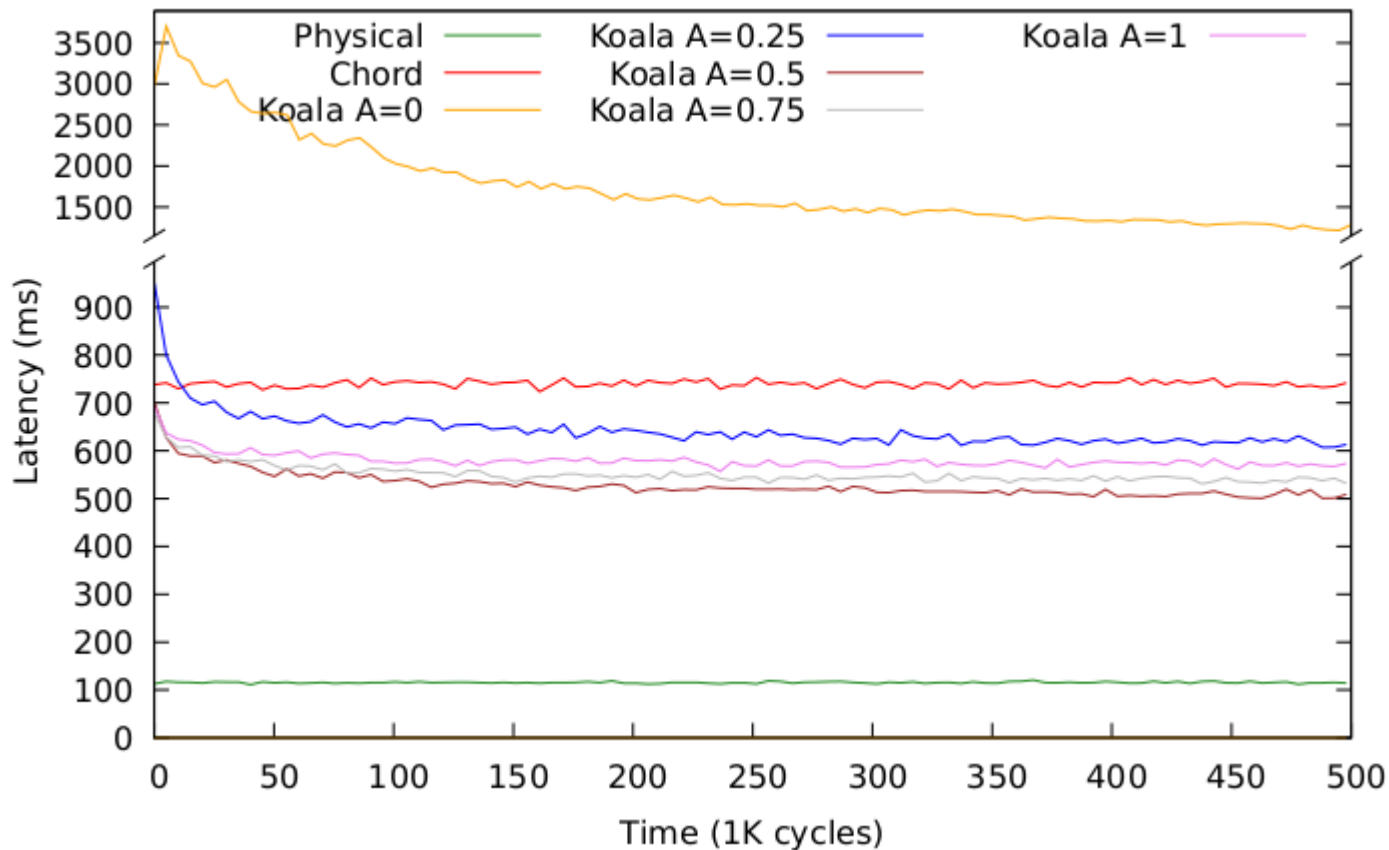
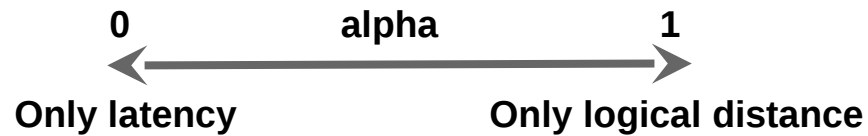
Add nodes each cycle until 100K  
# of long links =  $C * \log N$



# Locality-aware vs Greedy (hops)



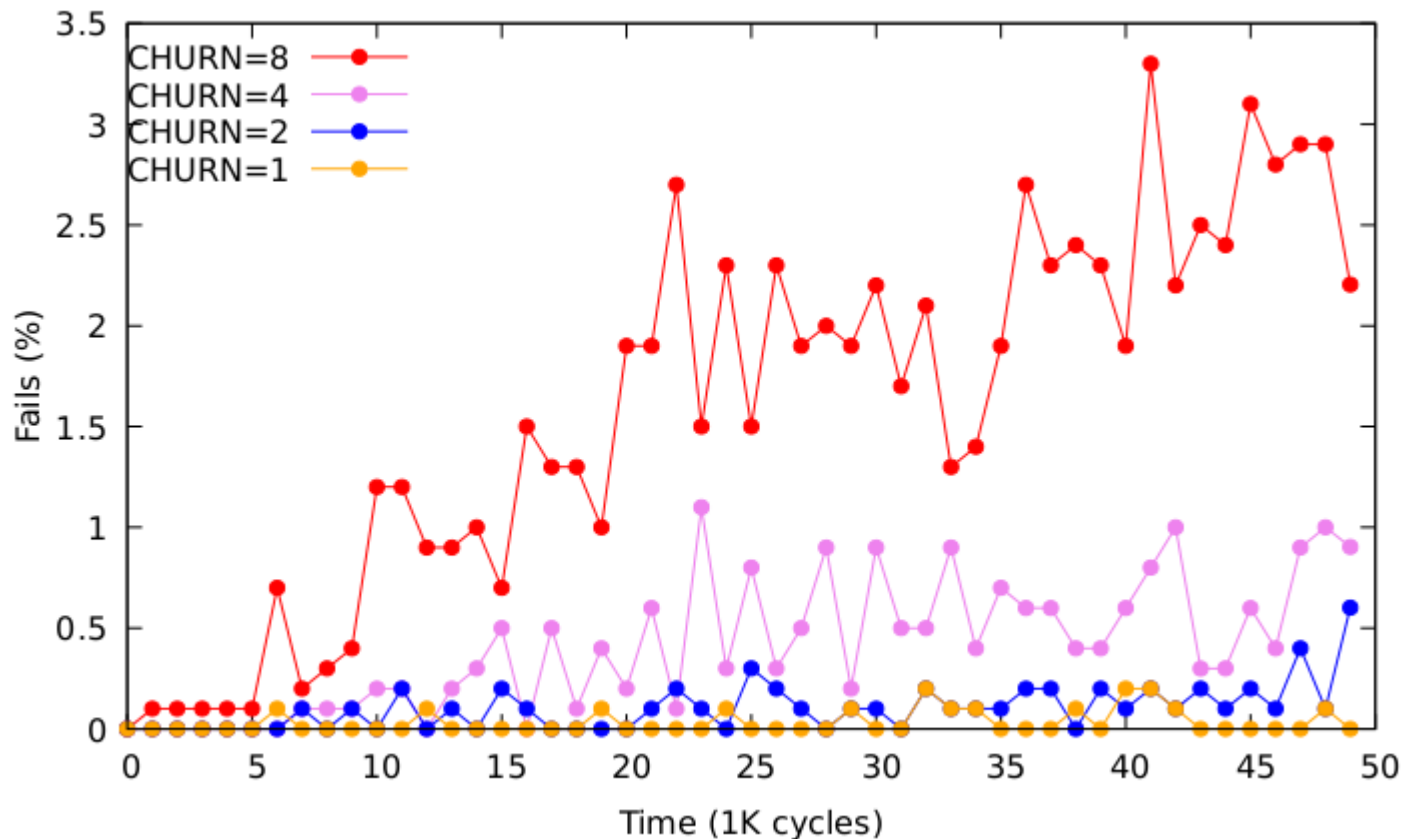
# Locality-aware vs Greedy (latency)



# Resilience

$N = 5K$

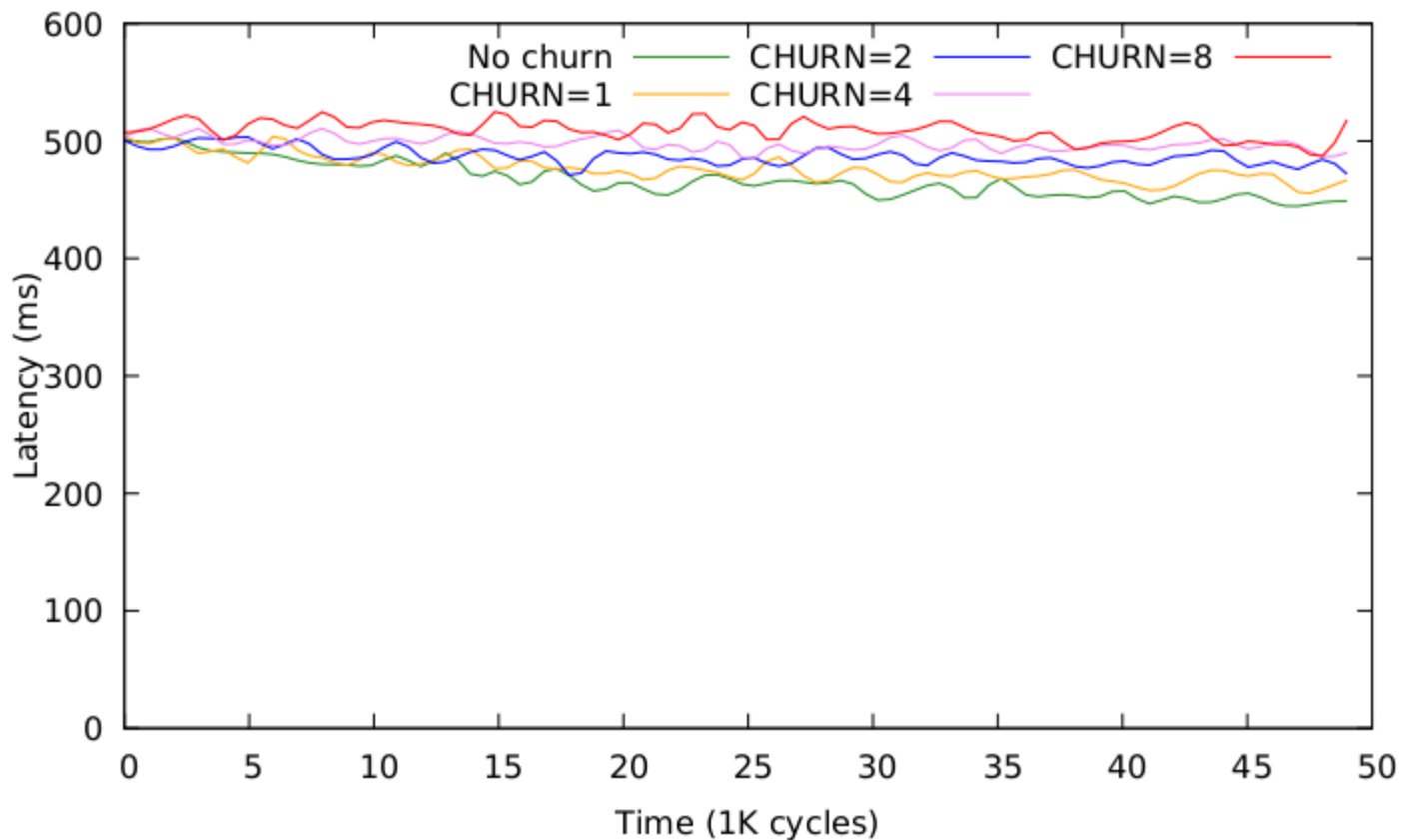
Every 80 cycles CHURN nodes leave and the same number joins



# Resilience

N = 5K

Every 80 cycles CHURN nodes leave and the same number joins

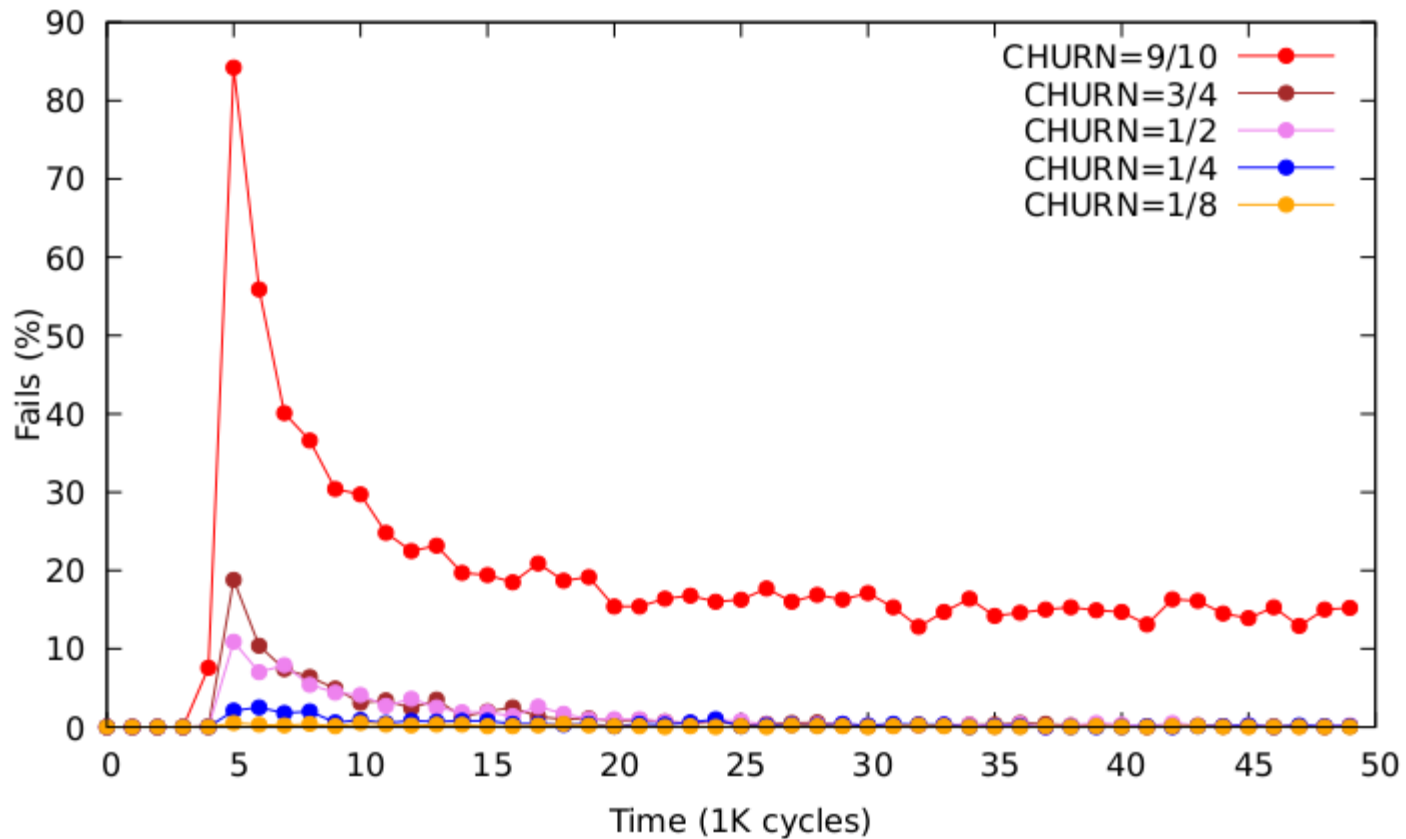




# Recovery

$N = 10K$

At cycle 5K we introduce an unexpected failure





# Results submitted to EuroPar 2017





## 2-level structure

...but so far we have assumed a PoP has only one server

Need to distinguish between local and global nodes.

A decorative background network diagram consisting of various nodes (circles) and edges (lines) in light gray, scattered across the slide.

# An Overlay Network

...but how should this overlay be?

1. Lazy (update overlay only when needed)
2. Locality-aware



# An Overlay Network

...but how should this overlay be?

1. Lazy (update overlay only when needed)
2. Locality-aware
- 3. Consider 2-level structure**

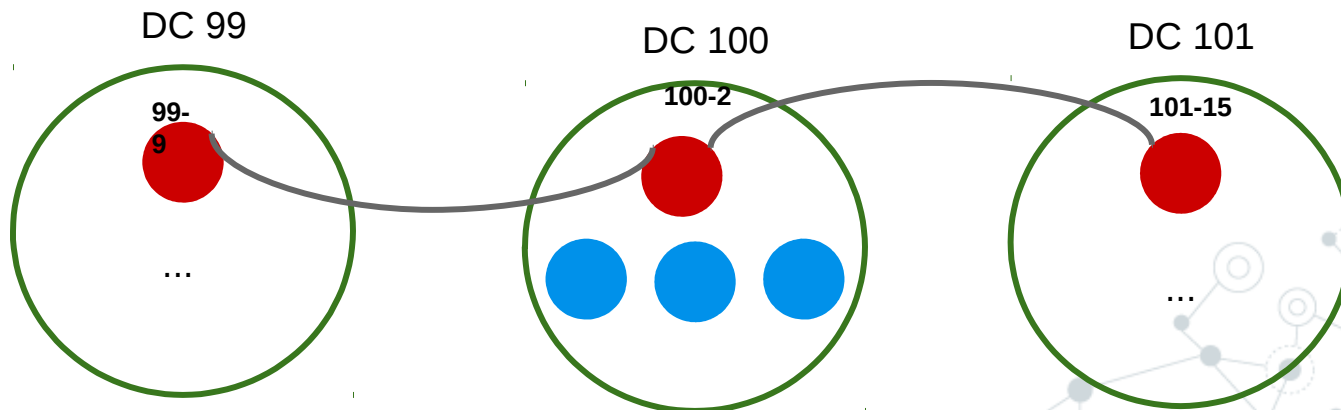
# Hierarchical?

A leader in each PoP.

Responsible for global communication.

Upside: smaller global overlay

Downside: Needs replication and leader election protocols are cumbersome!



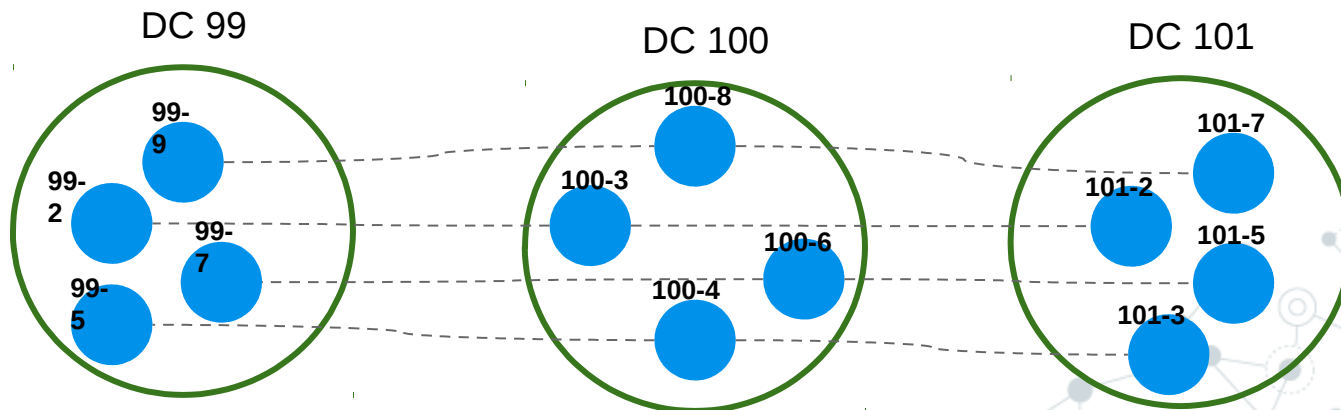
# Flatten hierarchies

Nodes are still in a local group

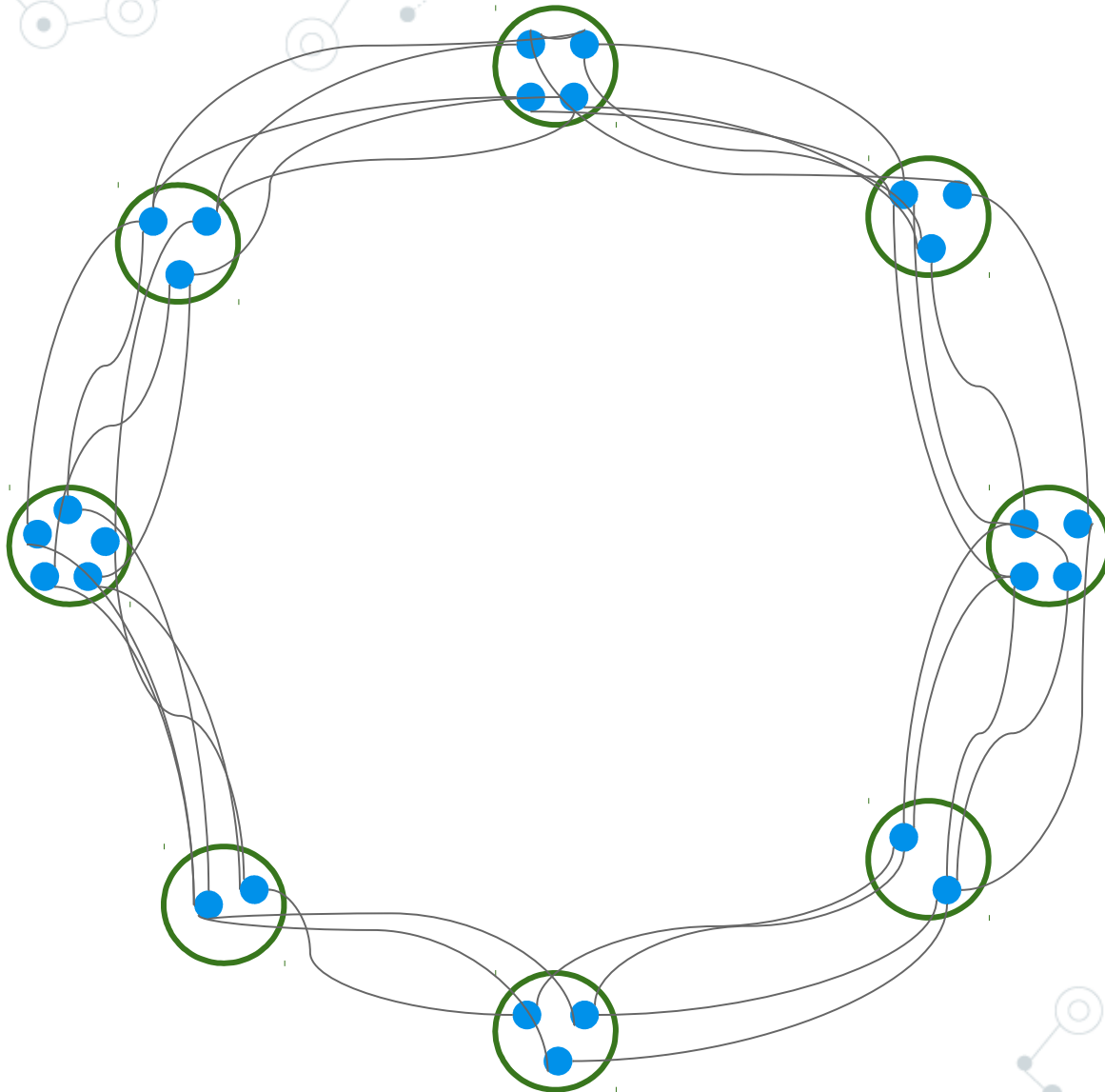
Every node has enough knowledge to route globally

Upside: Load balancing, resilience, comparable size of global system

Downside: Multiple rings



# The multi-ring





# Intra-PoP communication

We opt for a 0-hop DHT approach.

Since the number of servers is limited, everybody can now everybody else.

Consider ways to implement a lazy 0-hop DHT



A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles inside, suggesting a hierarchical or central structure. The lines are thin and gray, connecting the nodes in a non-linear fashion.

# **Thank you!**

## **Questions?**