Peer to peer: beyond file sharing

Anne-Marie Kermarrec

ASAP, INRIA

A case for decentralized systems

- Economical reasons
- Performance
- Some applications are intrinsically distributed
- Enhanced availability
- Resource sharing (data, storage, bandwidth) and aggregation
- Flexibility (load balancing)
- Incremental growth

Growing need of working collaboratively, sharing and aggregating distributed (geographically) distributed.

The new deal in distributed computing

Distributed systems are evolving

- **Scale shift**
- **Dynamics**

Traditional algorithms are no longer efficient : scalability

Peer to peer communication paradigm fills this gap

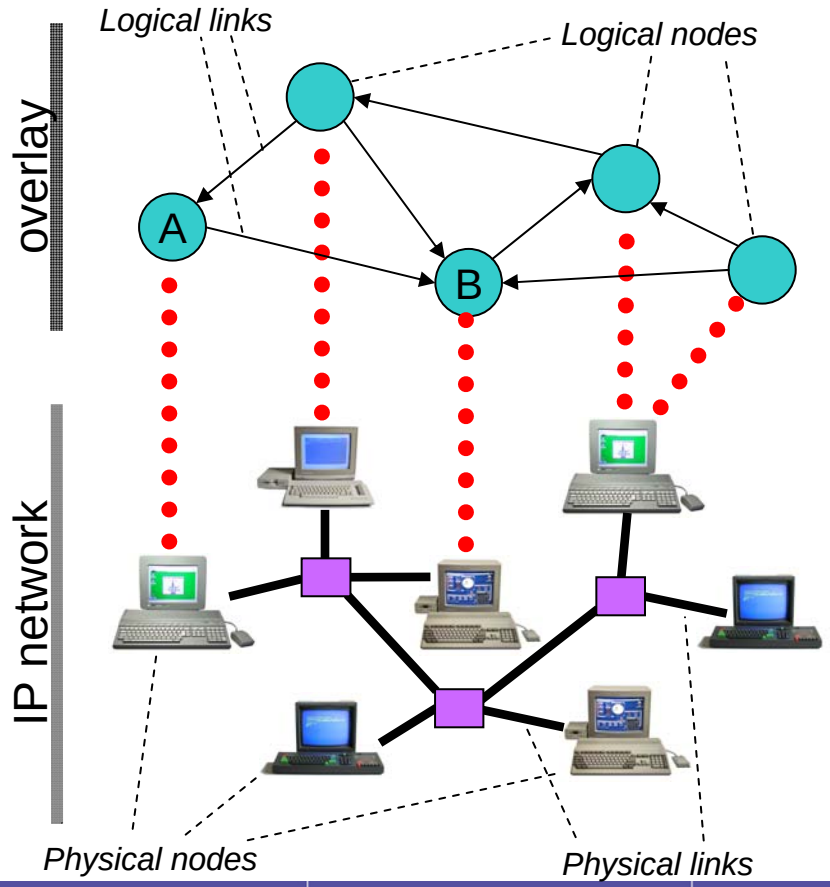
- Fully decentralized
- Self-organizing/enhanced availability
- Symmetric peers/load balancing
- Local knowledge of the system/global convergence

What makes P2P interesting?

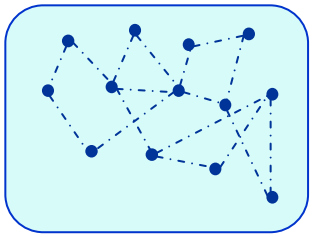
- End-nodes are promoted to active components!
- Nodes participate, interact, contribute to the services they use: nodes share benefits AND duties
- Harness huge pools of resources available at the edge of the Internet
- Irregularities and unpredictability considered as the norm

Peer-to-Peer Systems

The core: overlay networks



Peer to peer overlay networks



Routing capabilities

Flexibility

Unstructured networks

Fully structured networks

- Provide various functionalities/performance: search, dissemination, etc
- Common characteristics
  - Self-organizing
  - Local knowledge
  - Resource aggregation
- Resulting properties
  - Scalability
  - Resilience to churn

Impact of the structure on search

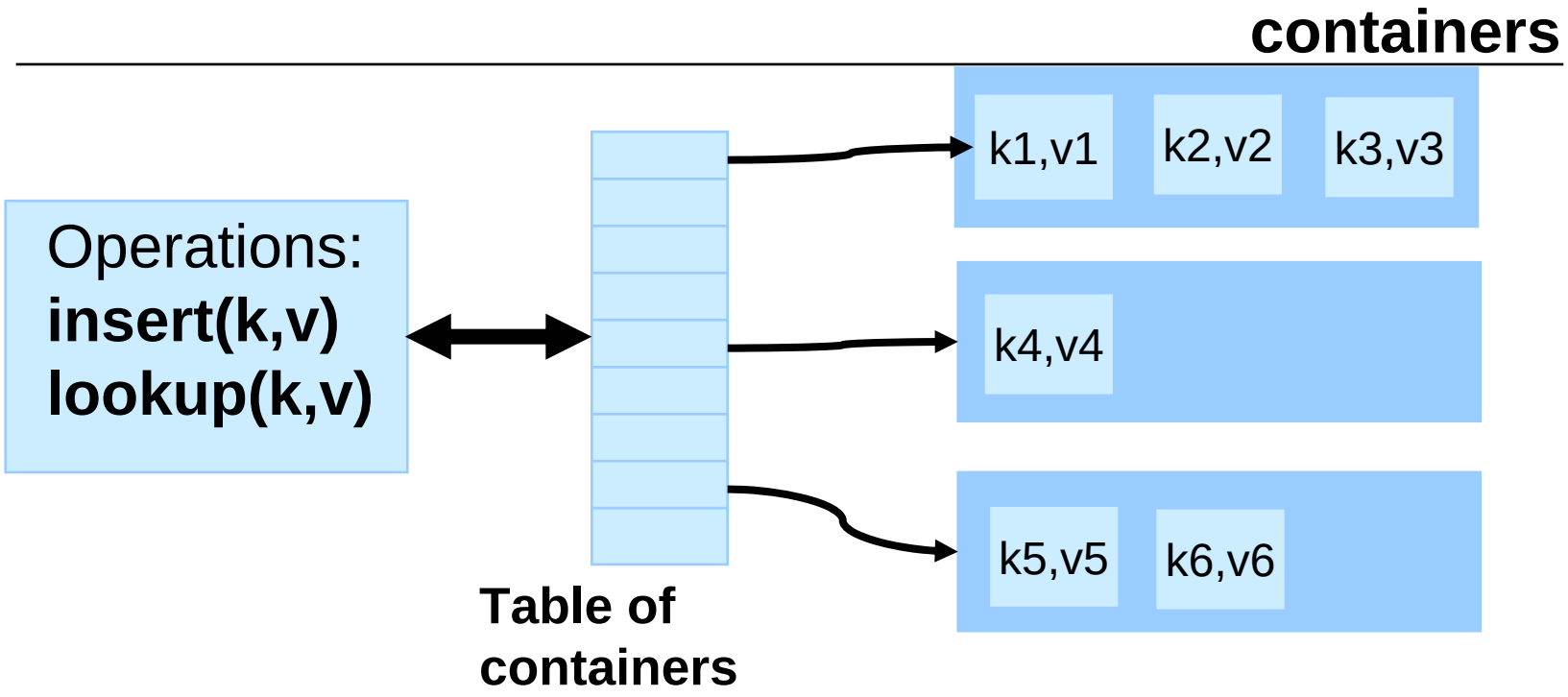
- Several ways of organizing a P2P overlay network
  - Search techniques: flooding versus routing
  - Expressiveness
  - Completeness
- Structured P2P overlay: DHT functionality
  - Support for exact search
- Unstructured gossip-based P2P overlays
  - Support for keyword-based search or range queries
- Weakly structured gossip-based overlays
  - Improve search efficiency upon fully unstructured overlays

1-Structured P2P networks

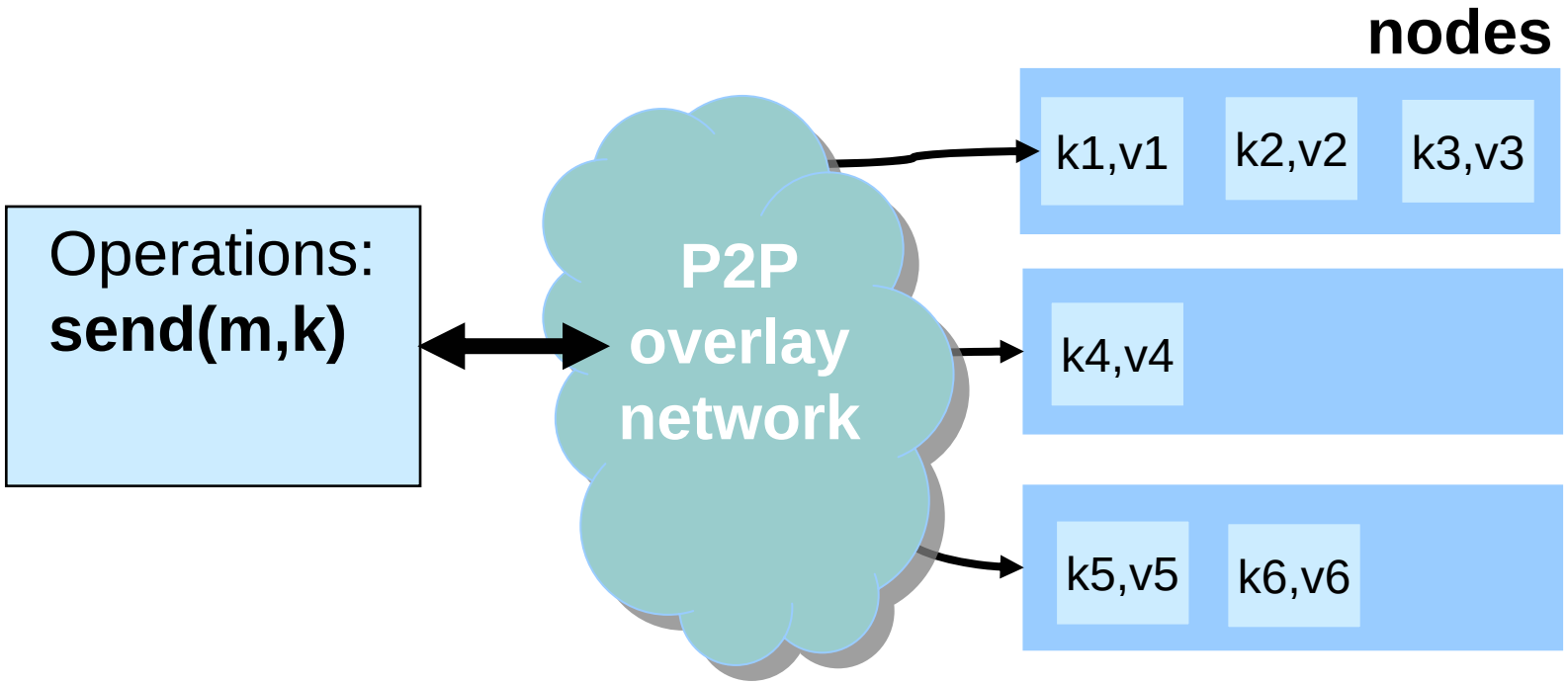
- Basic functionality: distributed hash table

- Applications
  - Content-delivery networks
  - Storage systems, Caching
  - Naming services
  - Multicast

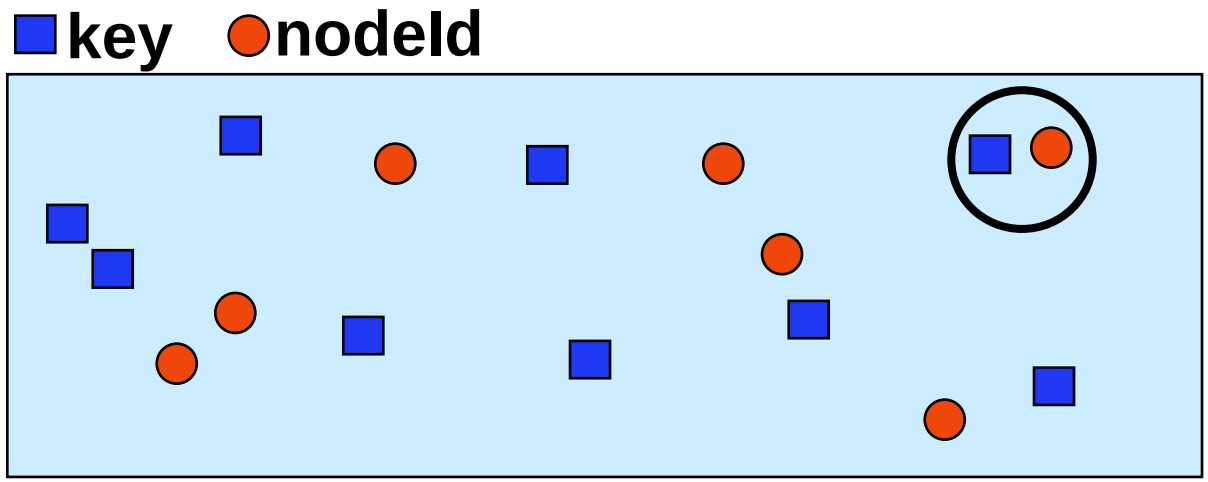
Distributed Hash Table (DHT)



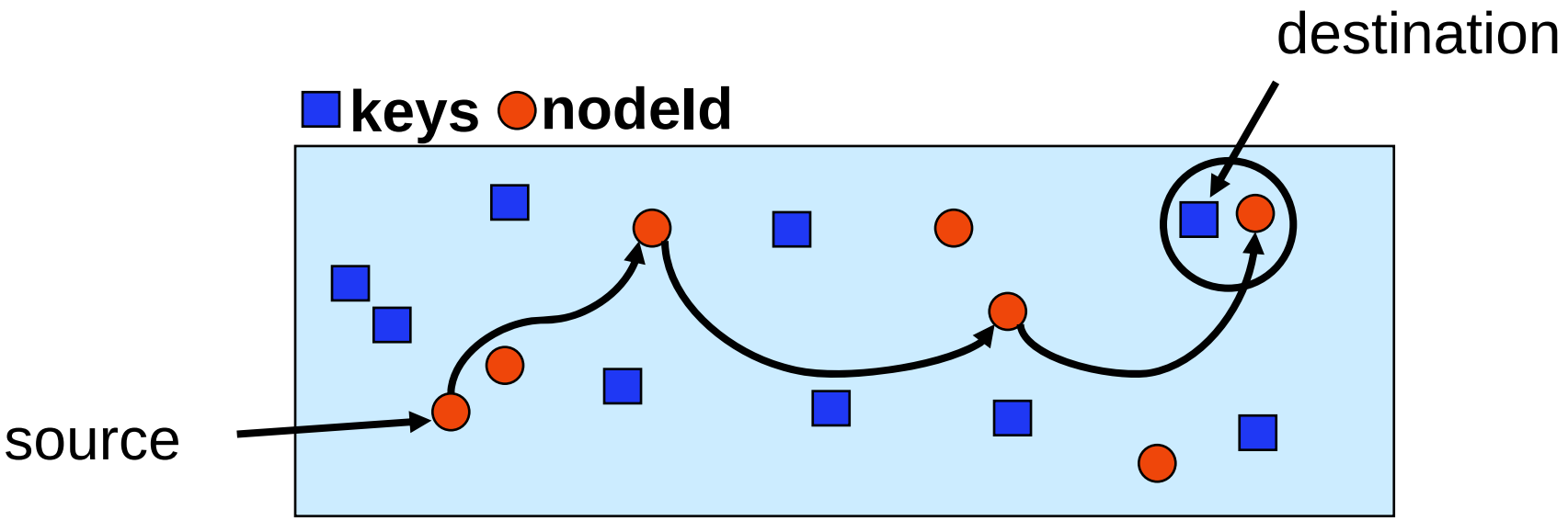
Distributed Hash Table



Mapping



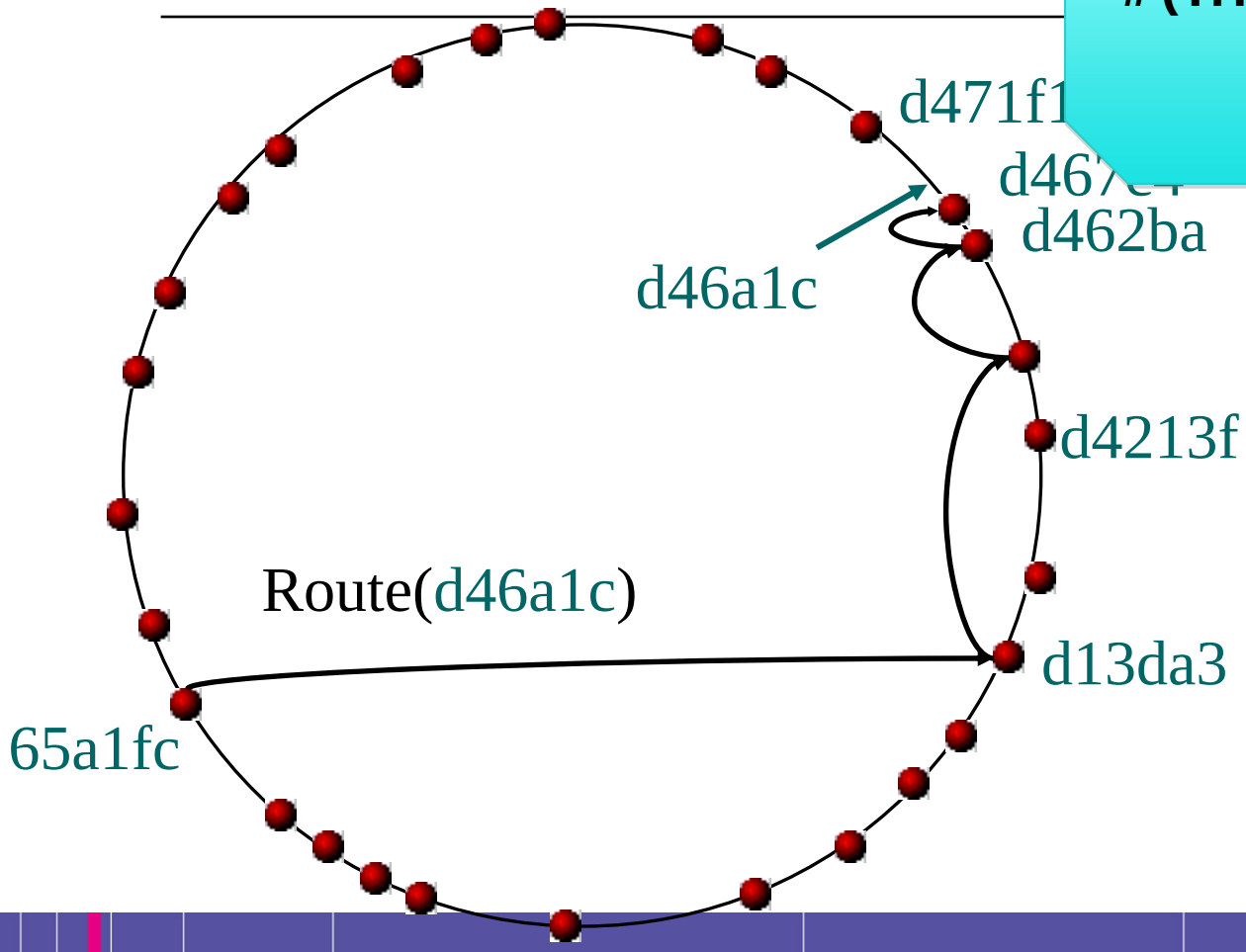
Sending messages to keys



Pastry: Routing

Exemple

#(Irisatech-10/10/08) = d46a1c



Properties

- $\log_{16}$ N hops
- Size of the state maintained (routing table): O(*log N*)

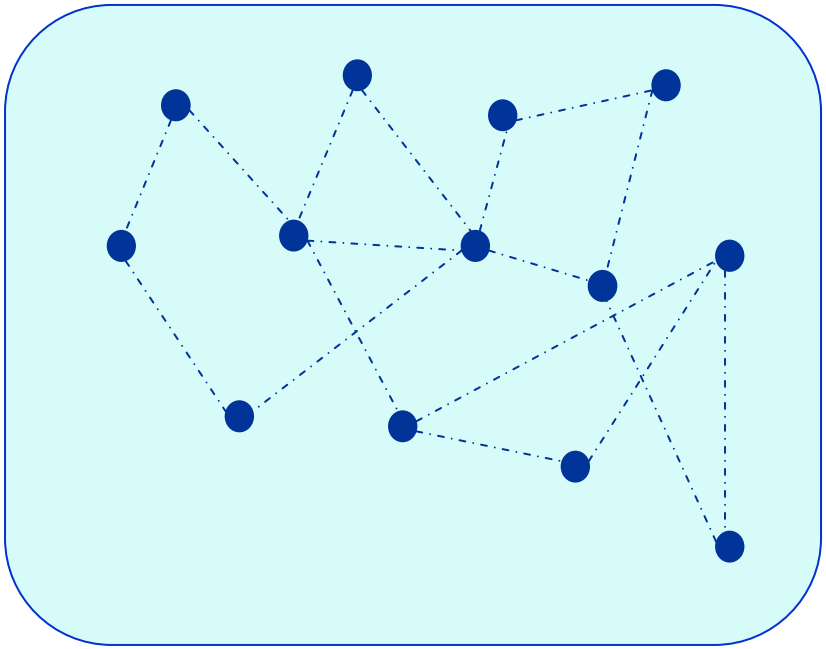
Pastry: Routing table(#65a1fcx)

Line 0	0 x	1 x	2 x	3 x	4 x	5 x		7 x	8 x	9 x	a x	b x	c x	d x	e x	f x
Line 1	60 x	61 x	62 x	63 x	64 x		66 x	67 x	68 x	69 x	6a x	6b x	6c x	6d x	6e x	6f x
Line 2	650 x	651 x	652 x	653 x	654 x	655 x	656 x	657 x	658 x	659 x		65b x	65c x	65d x	65e x	65f x
Line 3	65a0 x		65a2 x	65a3 x	65a4 x	65a5 x	65a6 x	65a7 x	65a8 x	65a9 x	65aa x	65ab x	65ac x	65ad x	65ae x	65af x

1-Unstructured P2P networks

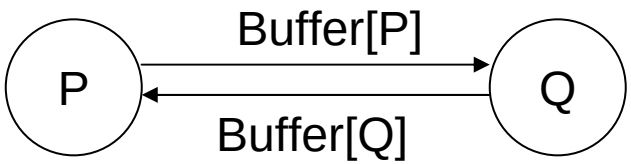
- Flexible infrastructure

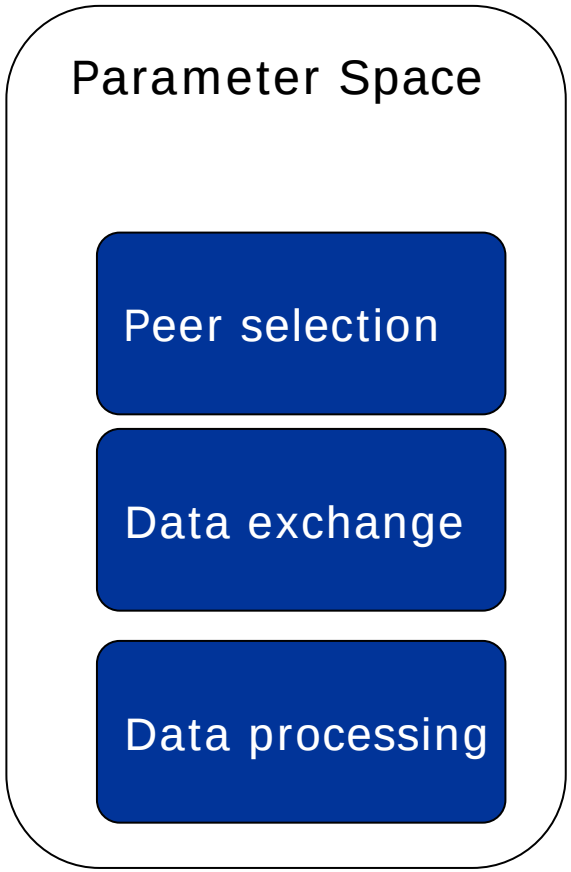
- Applications
  - Video streaming
  - Content-based search
  - Multicast



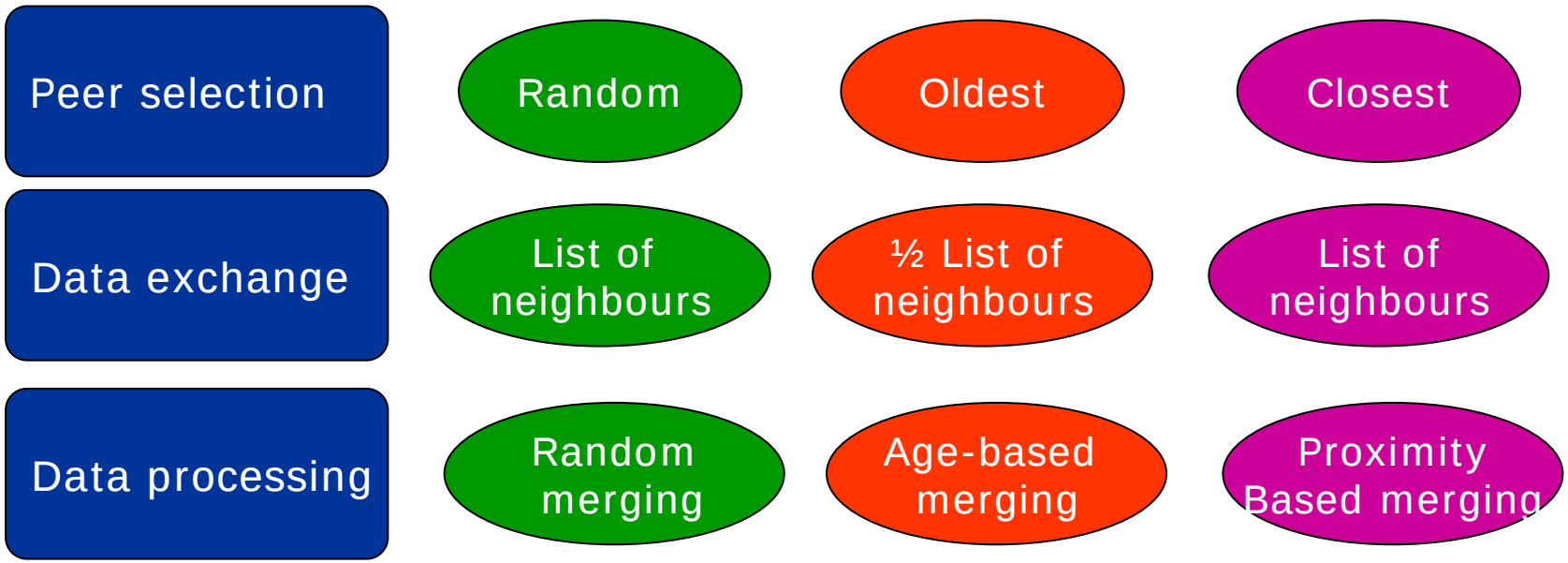
Gossip-based generic substrate

- Each node maintains a set of neighbours (c entries)
- Periodic peerwise exchange of information
- Each process runs an active and passive threads

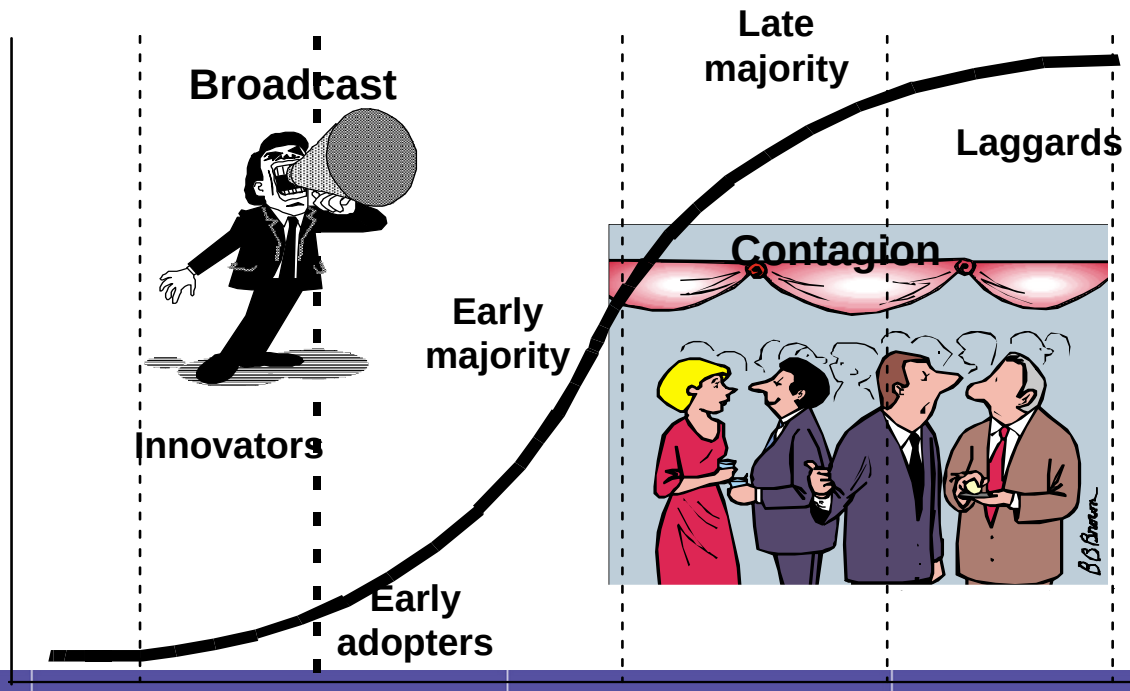




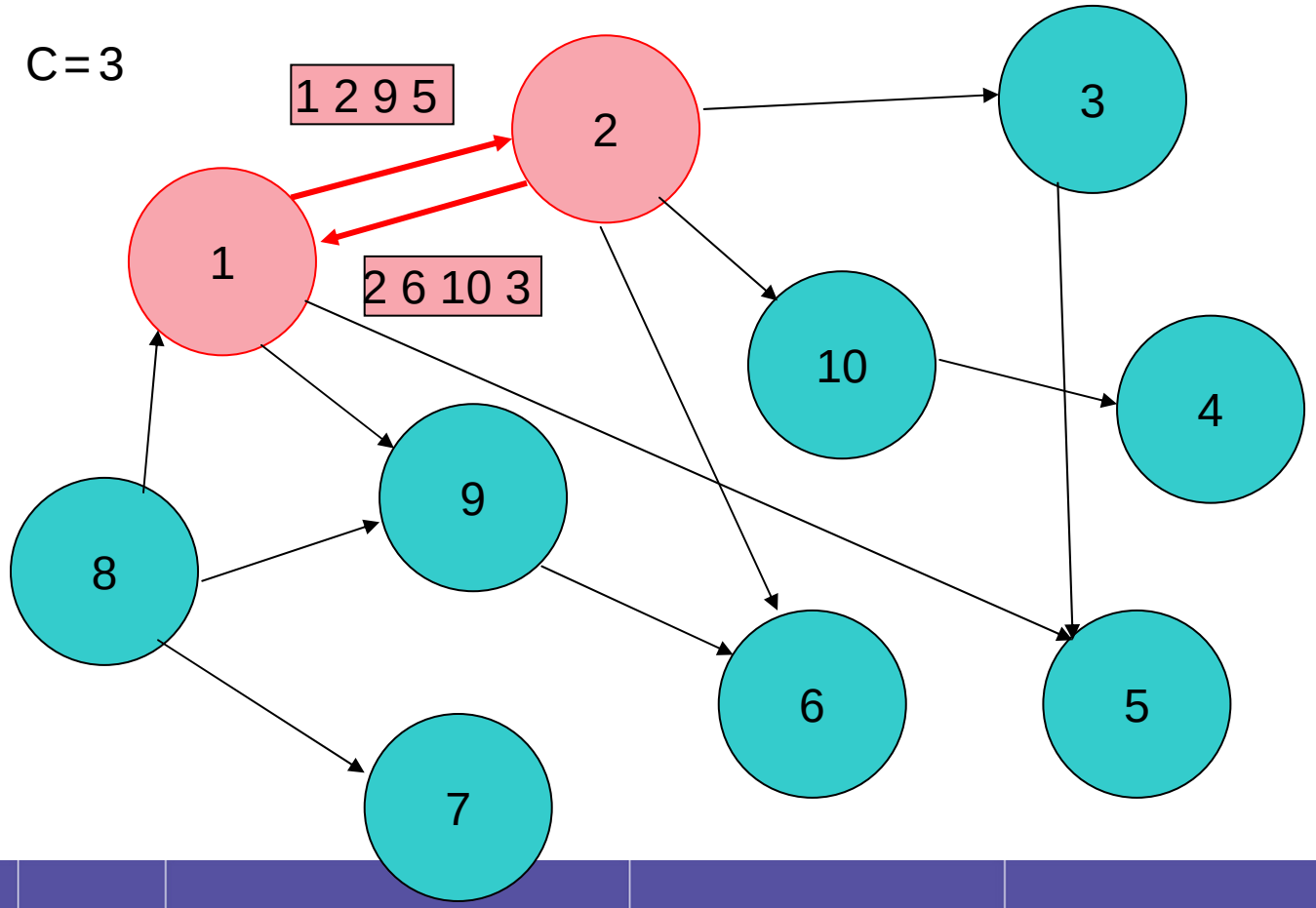
Overlay maintenance



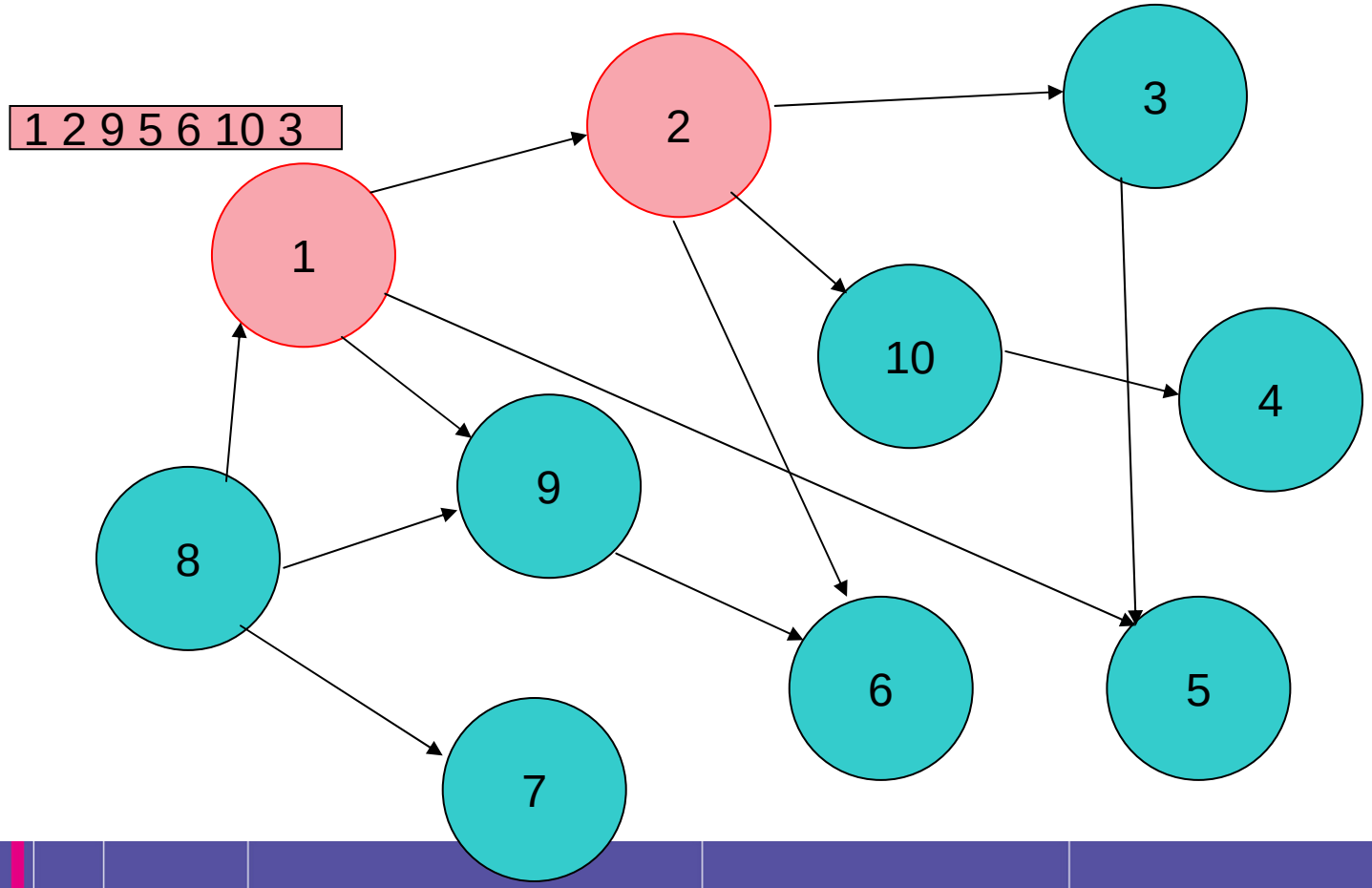
Why are we interested in building random graphs?



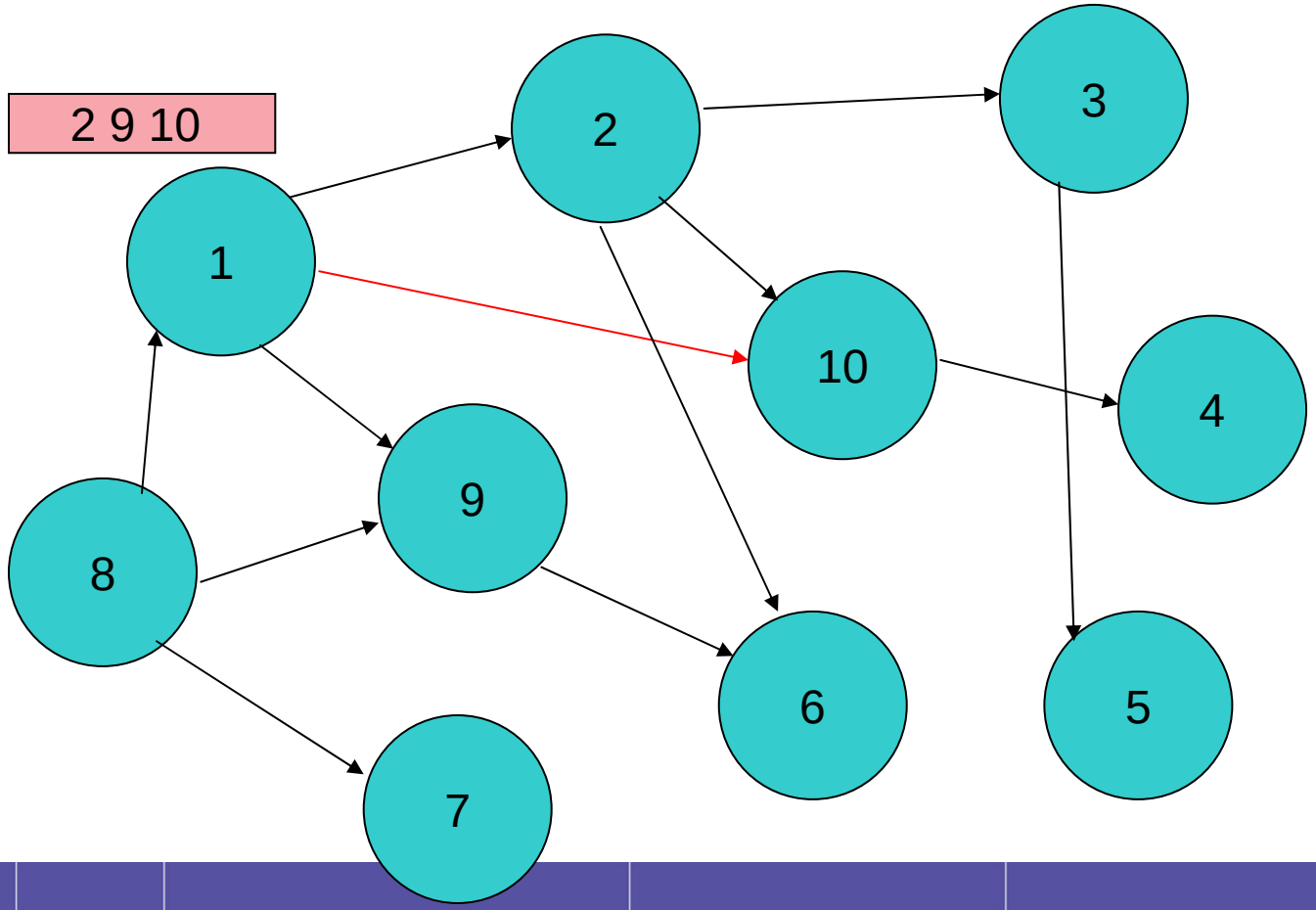
Example: Gossip-based generic protocol



Example: Gossip-based generic protocol



Example: Gossip-based generic protocol



Resulting graphs properties

Relationship « who knows who »

- Highly dynamic
- Capture quickly changes in the overlay networks

Flexible and powerful infrastructure

The take-away slide