Dependable Distributed Systems
Master of Science in Engineering in Computer Science

AA 2023/2024

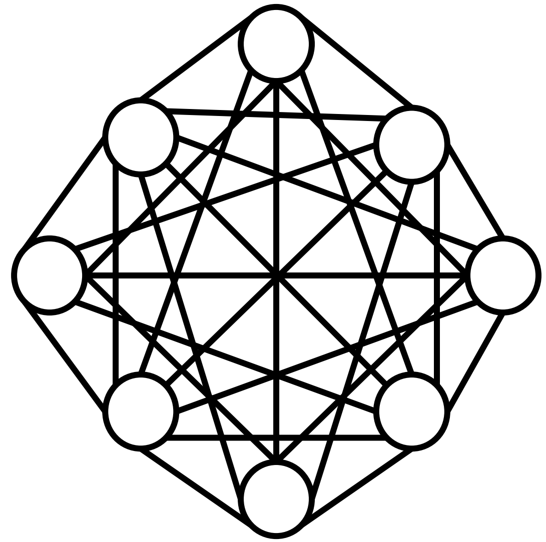
LECTURE 21: OVERLAY NETWORK AND MANAGEMENT

Schema

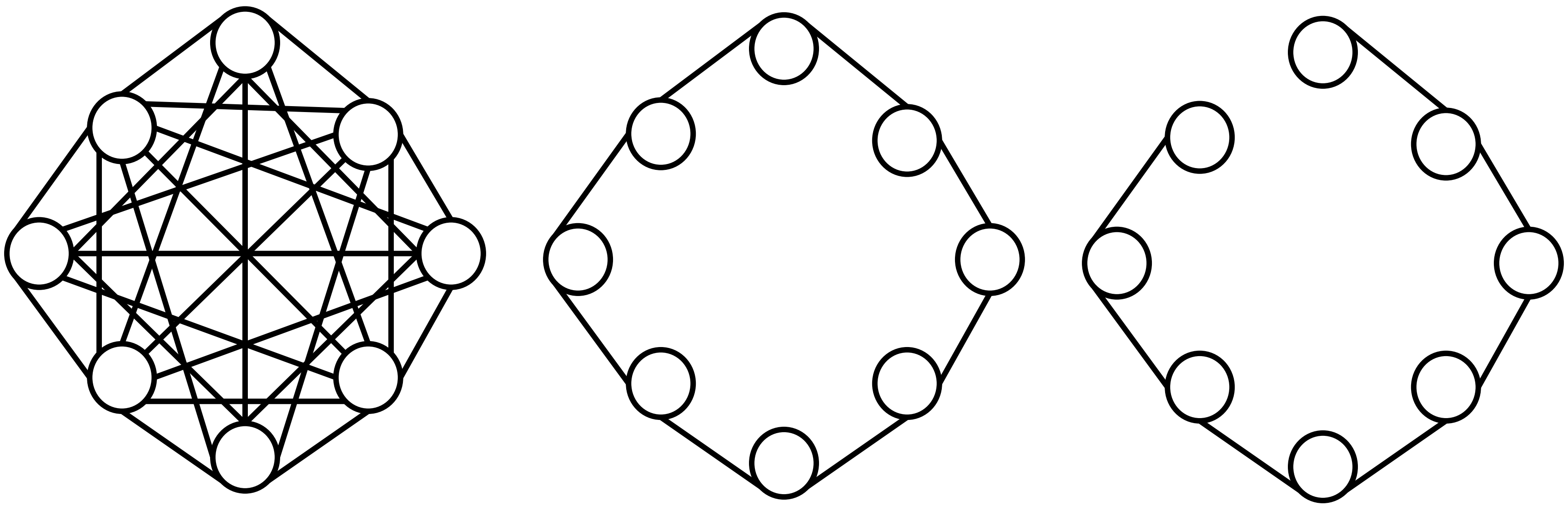
A Distributed System Representation - Graph

Processes

Link

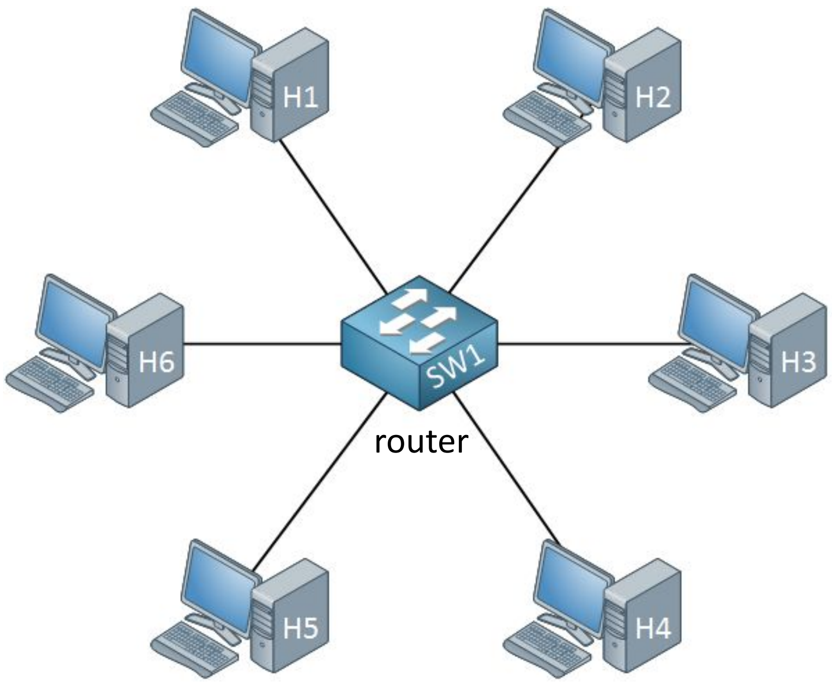


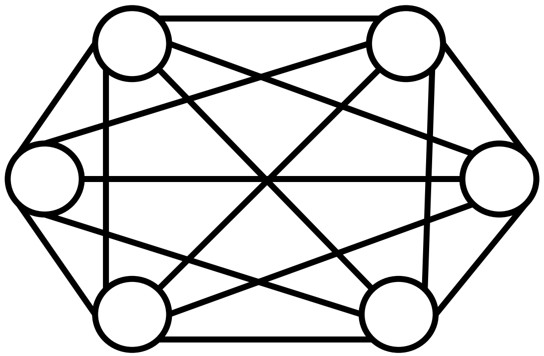
Network Topology



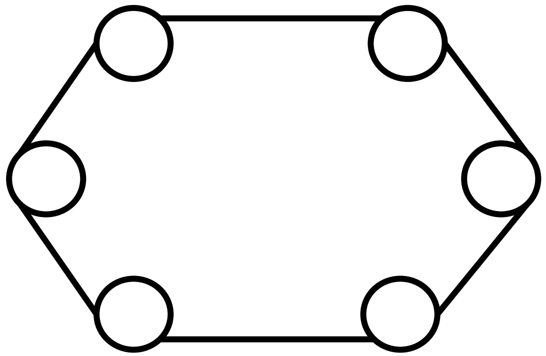
Links: model the capability of a pair of processes in exchanging messages

Network Topology (behind)





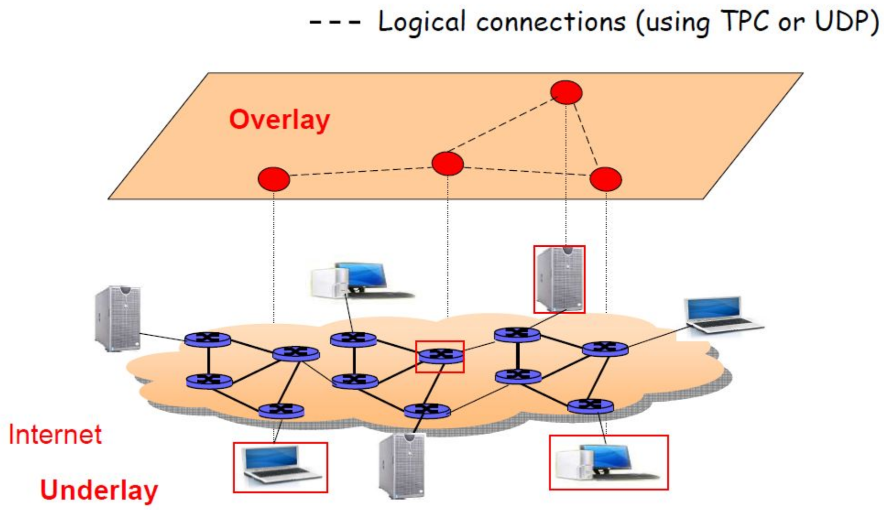
If all machines can exchange messages among them

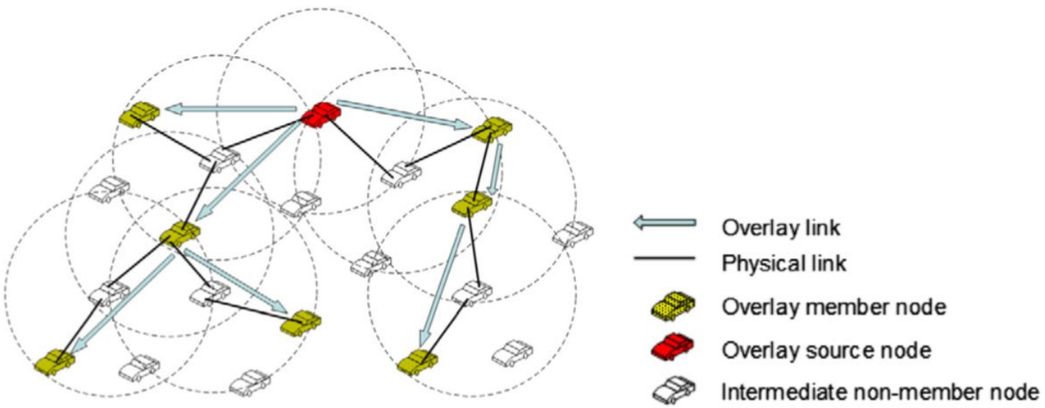


If every machine i can exchange messages only with i%6 ± 1

Overlay Network

A <u>logical</u> structure over a physical network





Overlay Network

An overlay network is a network that is **built on top of an existing network**.

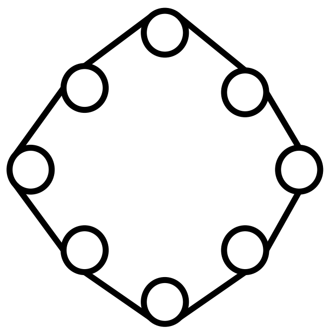
The overlay therefore **relies on** the so-called **underlay network for** basic networking functions, namely **routing and forwarding**

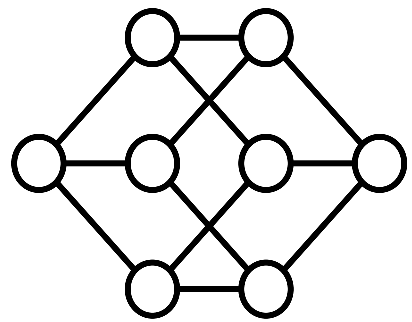
Most of **overlay networks are built in separate layer** on top of the TCP/IP networking suite

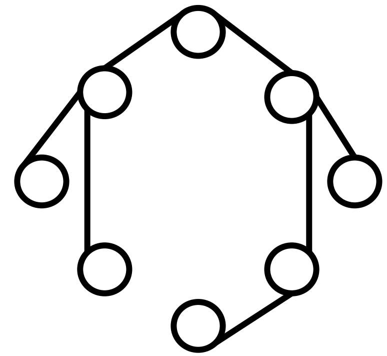
The nodes in an overlay network are connected via **logical links**

A link between two overlay nodes may take several hops in the underlying network

Basic Graph Metrics







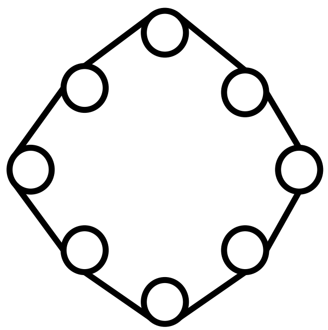
Distance(a,b): length of the shortest path between nodes a and b

Diameter: max among all distances

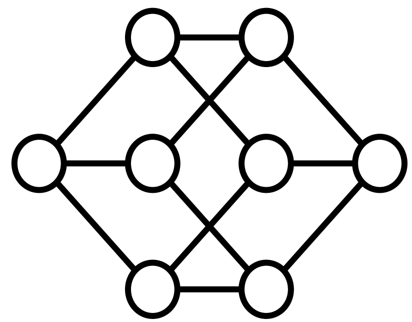
Closeness centrality: the average length of the shortest path between the node and all other nodes in the graph

Betweenness centrality: how important a node is to the shortest paths through the network

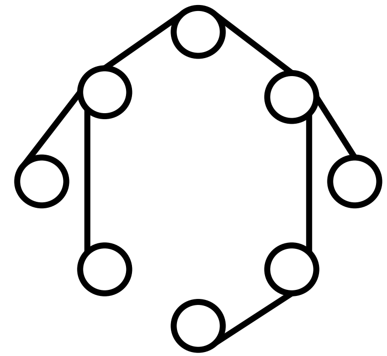
Basic Graph Metrics



Connected: it exists a path between every two nodes

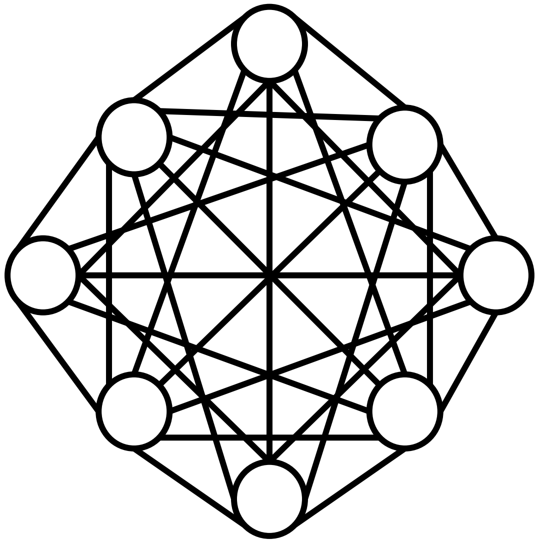


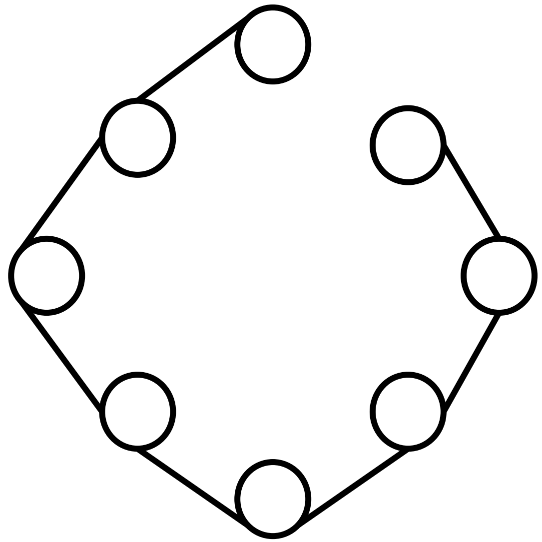
Edge-Connectivity: minimum number of *edges* that has to be removed to disconnect the network

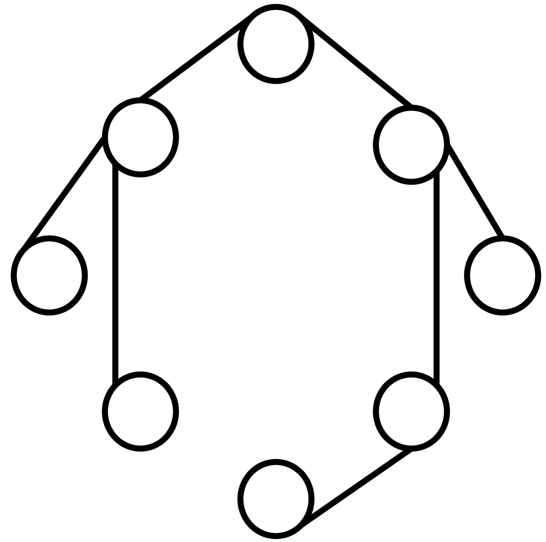


Node-Connectivity: minimum number of *nodes* that has to be removed to disconnect the network

Example (BEB - Performance)





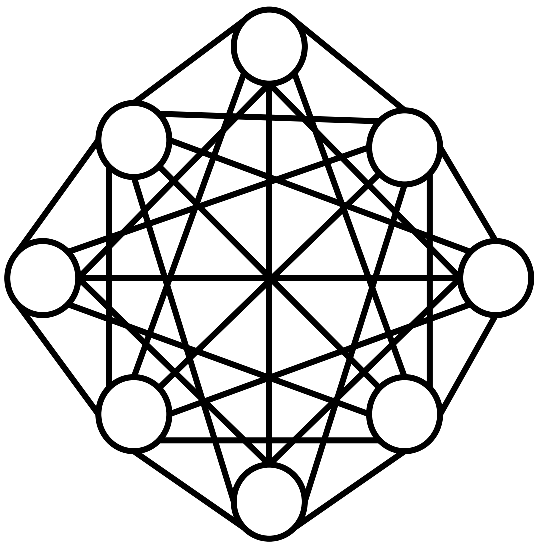


Load: the source has to send n messages
Latency: O(1) hop

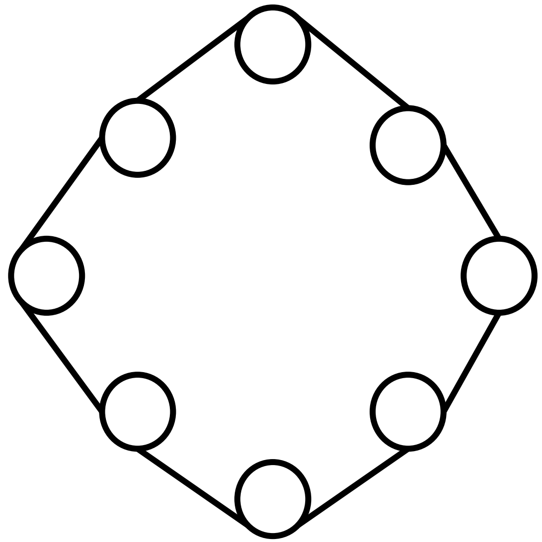
Load: the source has to send 1 message
Latency: O(n) hops

Load: the source has to send 3 messages
Latency: ≃ O(log2(n)) hops

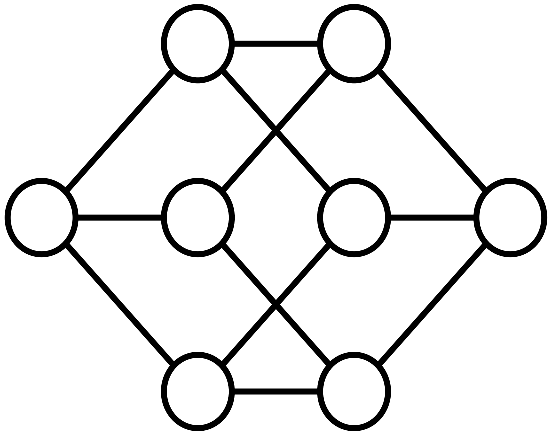
Example (BEB - Fault Tolerance)



Correctness: n-1 crashes



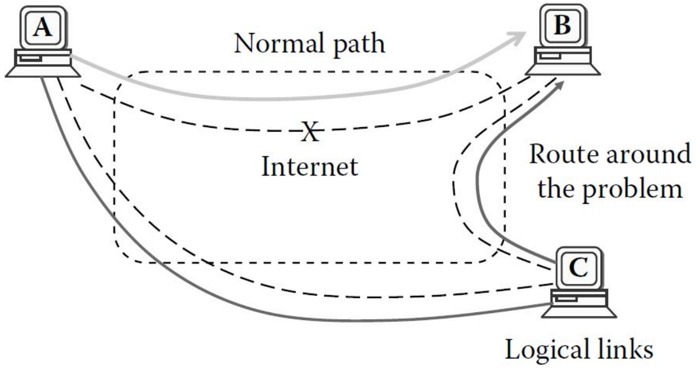
Correctness: 1 crash (worst case)



Correctness: 2 crashes (worst case)

Why set up an overlay?

- **Performance** and/or **scalability**
- **Induced**: processes may not know all the peers of the system (**membership**), or **reachability** issues
- **Incremental deployment**: do not require changes to the existing routers, can grow node by node
- **Adaptable**: The overlay algorithm can utilize a number of metrics when making routing and forwarding decisions. Thus the overlay can take application-specific concerns into account
- **Robust**: to node and network failures due to its adaptable nature; with a sufficient number of nodes in the overlay, the network may be able to offer multiple independent (router-disjoint) paths to the same destination. At best, overlay networks are able to route around faults.



Dynamic Distributed Systems in a nutshell

Informally, it is a distributed systems that **evolves over time**

Two types of evolutions:

- **Set of links** (i.e. the available links between the processes change over time)
- **Set of processes** (i.e. the actual processes in the system change over time)

In an actual distributed system **they may both change**

Overlay network protocols allows to **preserve correctness** of distributed protocols **despite the evolution** of the system

Note: the **evolution** of a distributed system can either be **intentional** or due to **faults**

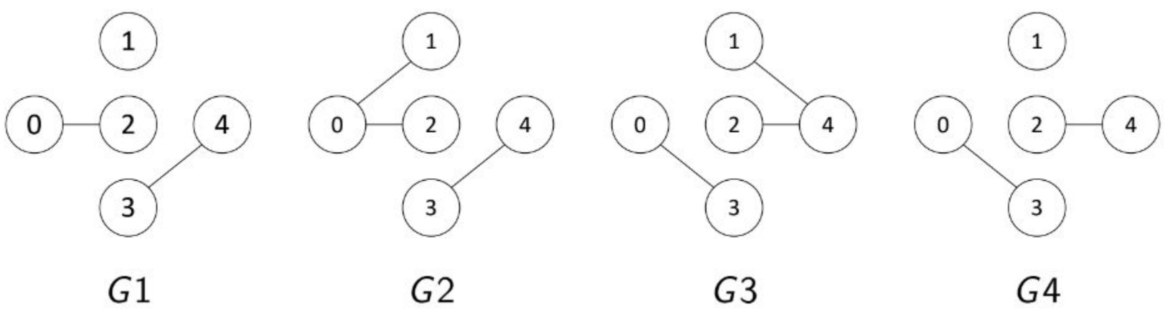
Dynamic Distributed Systems in a nutshell

Dynamicity of the processes:

- **Reconfiguration**: a protocol manages join and leaves (request to join, request to leave, reconfiguration of the system)
- **Churn**: joins and leaves are unmanaged, the churn is characterized by a patten (e.g. churn rate)

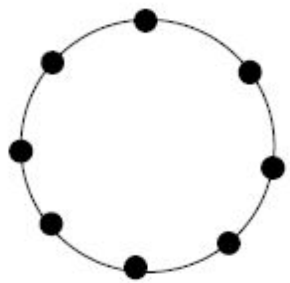
Dynamicity of the links:

- **Time-varying graphs**, general network features, …

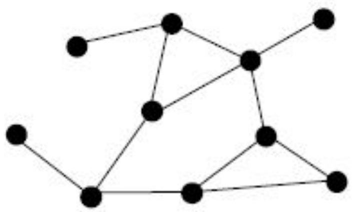


Classes of Overlay Networks

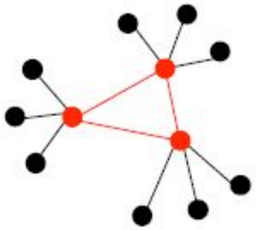
- **Structured**
  - nodes are arranged in a **restricted structure** (ring, tree, etc.)



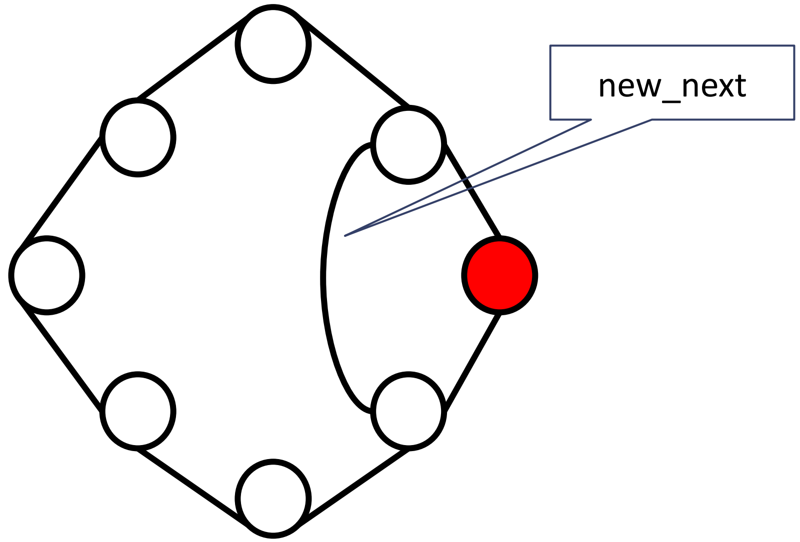
- **Unstructured**
  - do not impose any structure on the overlay network, the topology results from **some loose rules**, without any prior knowledge of the topology



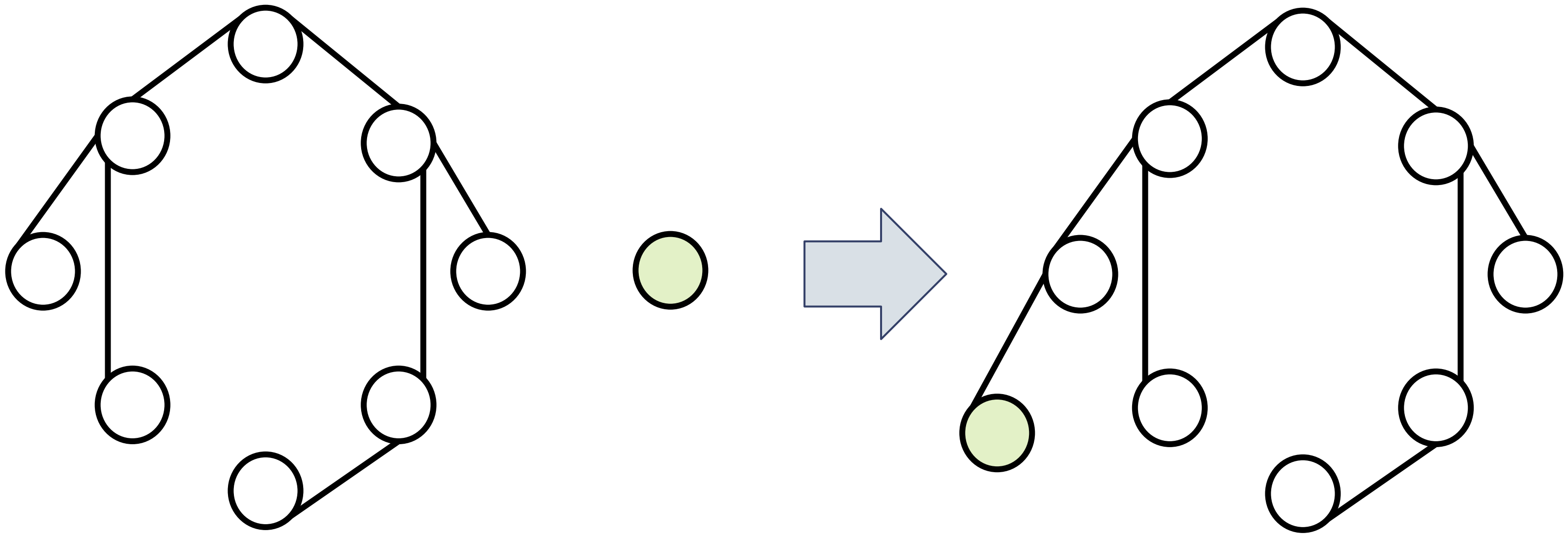
- **Hybrid and/or multi-level**:
  - E.g. supernodes form a small overlay



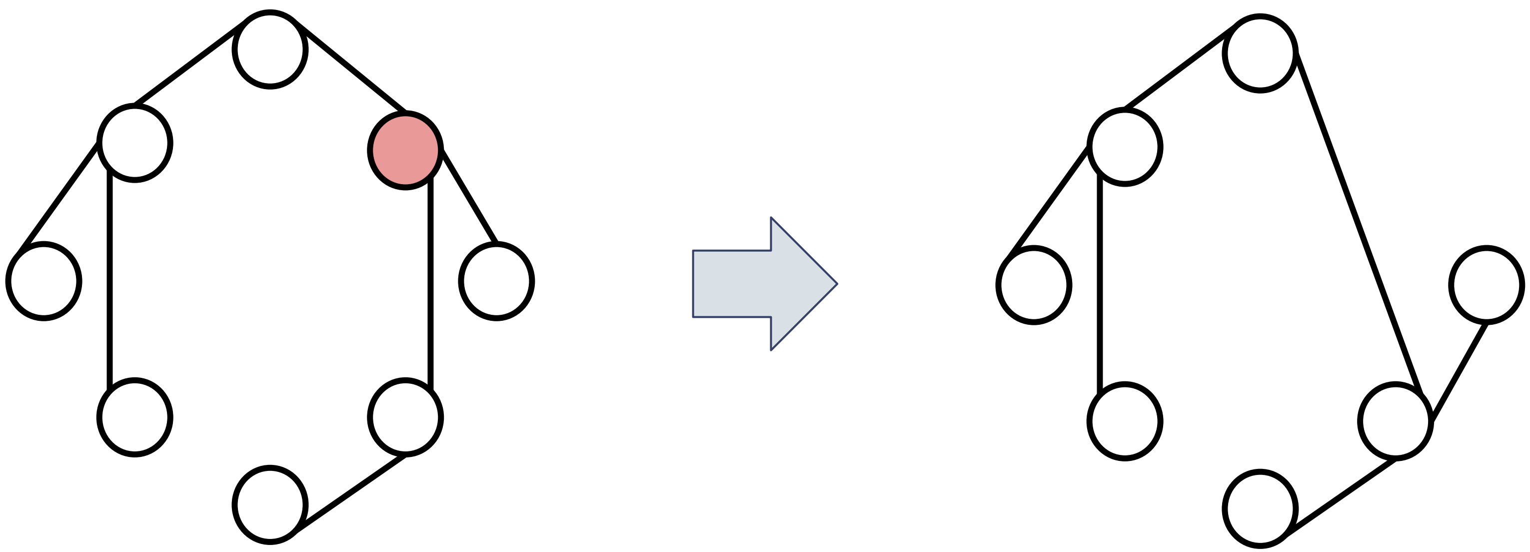
Structured Overlay Network - **Reconfiguration** Example



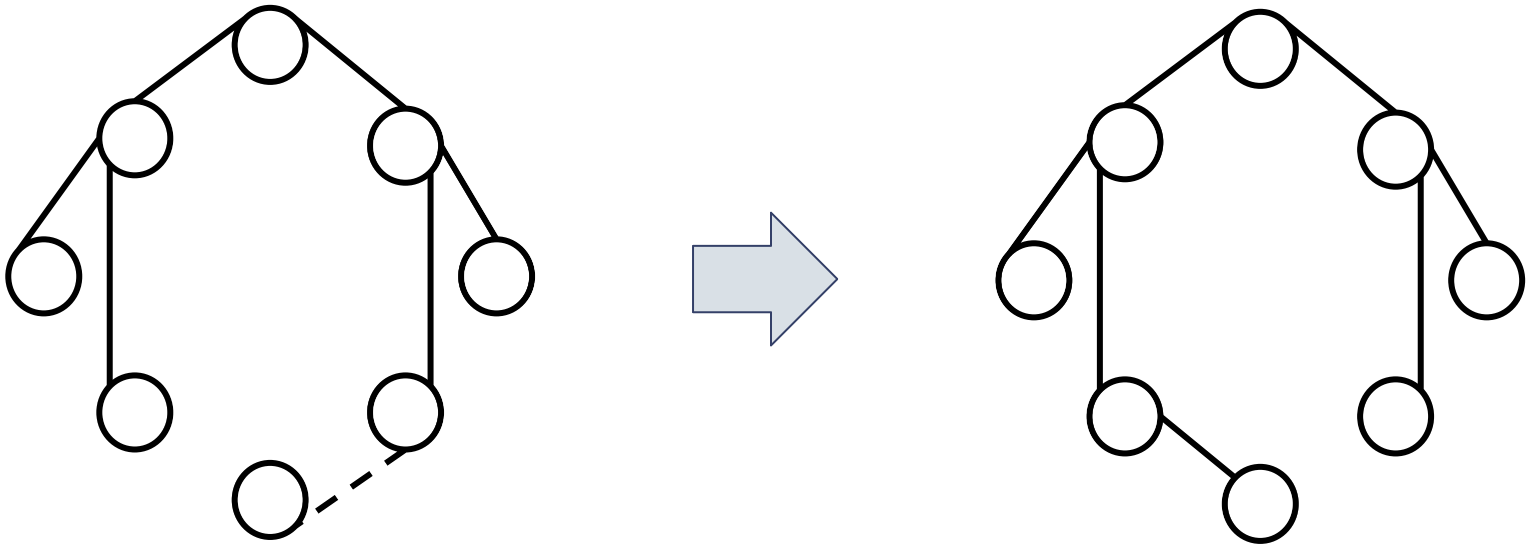
Structured Overlay Network - **Reconfiguration** Example



Structured Overlay Network - **Reconfiguration** Example



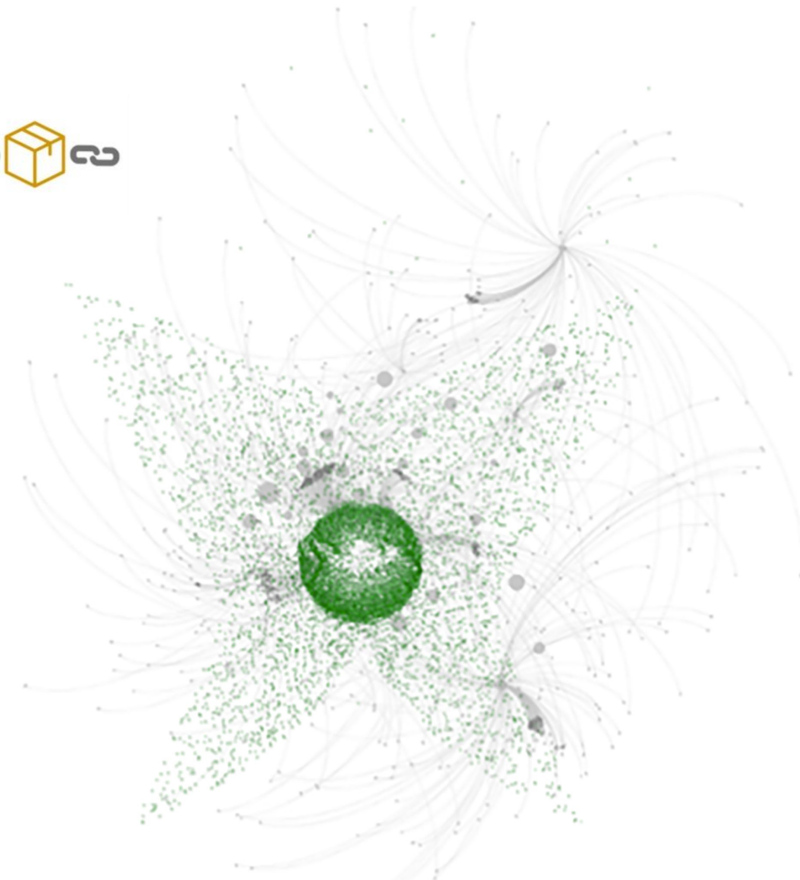
Structured Overlay Network - **Reconfiguration** Example



Structured Overlay, Data Dissemination and Retrieval

- **Routing Tables** - Messages/Requests generally follows defined paths
- **Defined Data Location** - Data are generally stored in defined locations
- **Deterministic Performance Metrics**

Unstructured Overlay Network - Bitcoin



From Essaid et al. "Bitcoin's dynamic peer-to-peer topology" *et al* (2020).

Bitcoin topology snapshot (on 2019/01/23). Node size represents degree connectivity, where nodes with a higher degree have larger circles

Bitcoin Network, Bootstrap

Before being able to send and receive protocol messages a node has to find other nodes to connect to join the network

Connection = Establish a link

- a node **first tries to connect to nodes it knows from participating previously**;
- **If no connections can be established** this way, or if the node connects for the very first time, then it **queries a list of well-known DNS seeds**. The DNS seeds return a random set of bootstrap nodes;
- As a last resource, it will try to connect to **hardcoded nodes**

Bitcoin Network, Bootstrap

- Peers discovery is done by **address rumouring, where connected nodes gossip about other potential active nodes**
- Once connected, **joining node asks for new neighbors and listen for announcements of new nodes**
- **Outbound destinations are randomly selected among known identities**
- **Each node attempts to keep a minimum number of connections *p* to other nodes open at all times**
- **No explicit way to leave the network**

Bitcoin Network

Random graph, a good **expander**, with high probability.

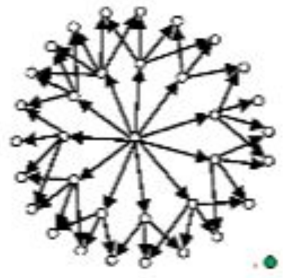
Intuitively, in graph theory an expander graph is a **sparse graph that has strong connectivity properties**, quantified using vertex, edge, or spectral expansion.

Note: The network formation process in the Bitcoin protocol is **designed to hide the global network structure**: while most of the nodes of the network can be easily discovered, the existence of an edge between two nodes is only known by the two endpoints

Unstructured Overlay, Data Dissemination and Retrieval

Flooding:

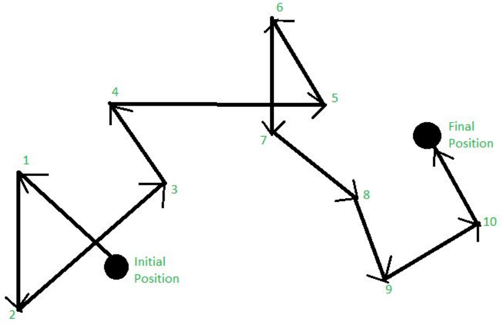
- An issuing node u **send/request** data item **to all its neighbors**.
- If v has the required data, it **respond directly** t, **or send it back to the original forwarder**, who will then return it to its original forwarder, and so on (multi-hop propagation).
- If v does not have the requested data, it **forwards the request to all of its own neighbors**.
- A request often has an associated time-to-live or **TTL** value, giving the maximum number of hops a request is allowed to be forwarded. **Choosing the right TTL value is crucial**: too small means that a request will stay close to the issuer and may thus not reach a node having the data. Too large incurs high communication costs.



Unstructured Overlay, Data Dissemination and Retrieval

Random-Walk:

- **Ask a randomly to chosen neighbor**,
- If v does not have the data, **it forwards the request to one of its randomly chosen neighbors**, and so on.
- Much less network traffic, yet it may take much longer before a node is reached that has the requested data.
- An issuer can start **n random walks simultaneously**
- **Time-To-Live**



Classes of Overlay Networks: **Features**

- **Structured**
  - + efficient data location
  - - longer join and leave procedures (reconfiguration)
  - - less robust in high-churn environments
  - Properties are generally provided deterministically

- **Hybrid and/or multi-level:**
  - Custom features
  - Usually large state and high load on supernodes

- **Unstructured**
  - + support more complex queries
  - + faster join procedure
  - + usually more tolerant to churn
  - + good for data dissemination,
  - + bad for data location
  - Properties are provided probabilistically

Services generally provided by Overlay network protocols

- Join (leave) the network
- Discover nodes
- Identification and Addressing
- Routing
- Data Dissemination and Retrieval
- Overlay management

Peer to Peer (P2P)

P2P systems comprise **self-organized equal** and **autonomous** entities (**peers**) aiming to **share distributed resources** in a given network.

No centralized control

Peers are **both client and servers, each node carries out the same tasks**

Local knowledge: nodes only know a small set of other nodes

Robustness: several nodes may fail with little or no impact

High scalability: **high aggregate capacity, load distribution**

Low-cost: storage and bandwidth are contributed by users

Peer-to-Peer: Applications

- **Data storage**: IPFS, BitTorrent, etc.
- **Distributed Ledger**: Ethereum, IOTA, etc.
- **Cryptocurrency**: Bitcoin, etc.
- **Security**: Onion routing, anonymous content storage, etc.
- Publish/subscribe, and many others

Identifiers

Identify a resource (data) in a distributed system: **Globally Unique IDentifier (GUID)**

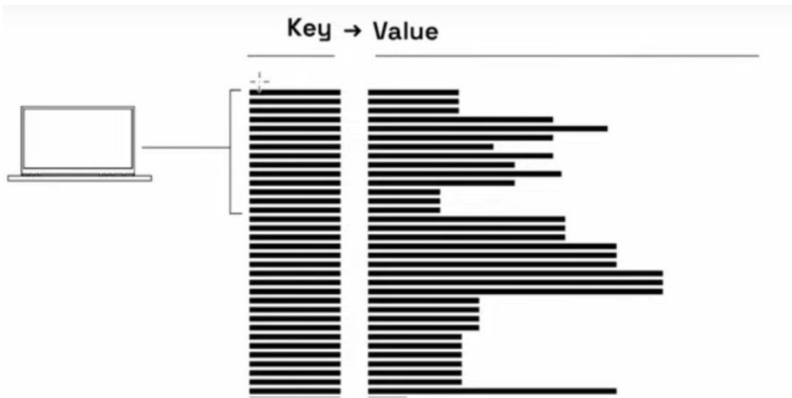
- Usually generated by applying a **secure hash function** to some (or all) of the resource's state

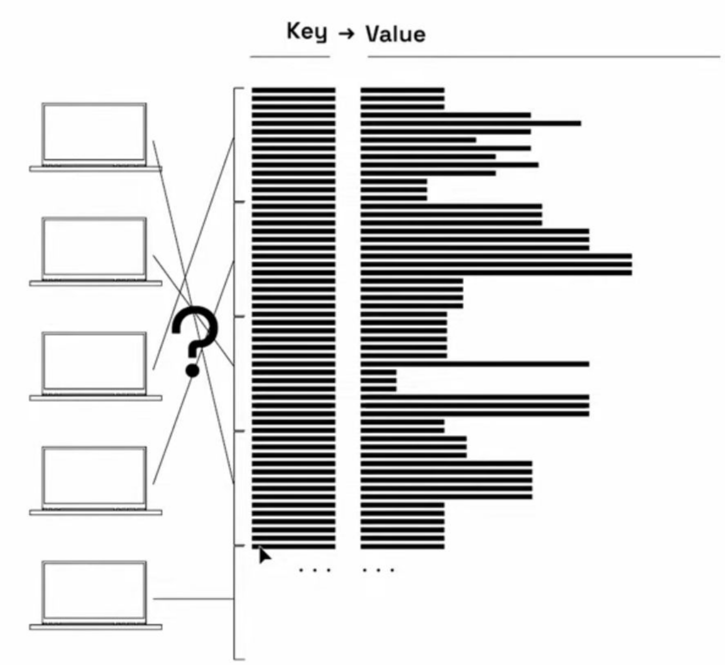
Identify peers

- Generate a random number, compute an identity by applying a secure hash function

There exists many other alternatives for both

Distributed Hash Table (DHT)





DHT

Distributed hash tables are a class of decentralized distributed algorithms that **offer a storage-lookup service**

DHTs **store (key, value) pairs**, and they support the **lookup** of the value associated with a given key

The **keys and values are distributed** in the system, and the DHT system must ensure that the nodes have sufficient information of the global state to be able to forward and process lookup requests properly

Data object (or value) location information is placed deterministically, at the peers with identifiers corresponding to the data object's unique key.

Linear Hashing and LH*: support table expansion

DHT

desirable property: find data locations within a bounded number of overlay hops

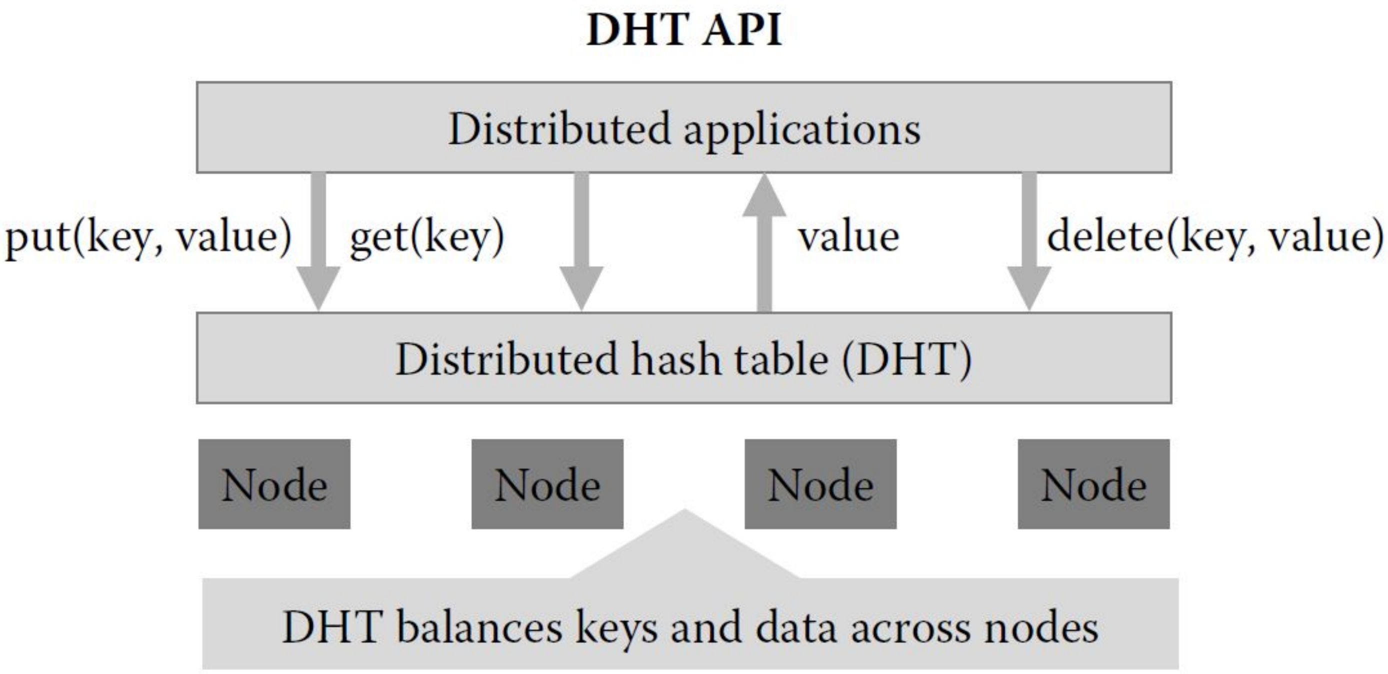
The **DHT algorithm is responsible for distributing the keys and values** in such a way that efficient lookup of the value corresponding to a key becomes possible

Since peer nodes may come and go, this requires that **the algorithm be able to cope with changes in the distributed system**.

The locality of data plays an important part in all overlays, since they are executed on top of an existing network, typically the Internet

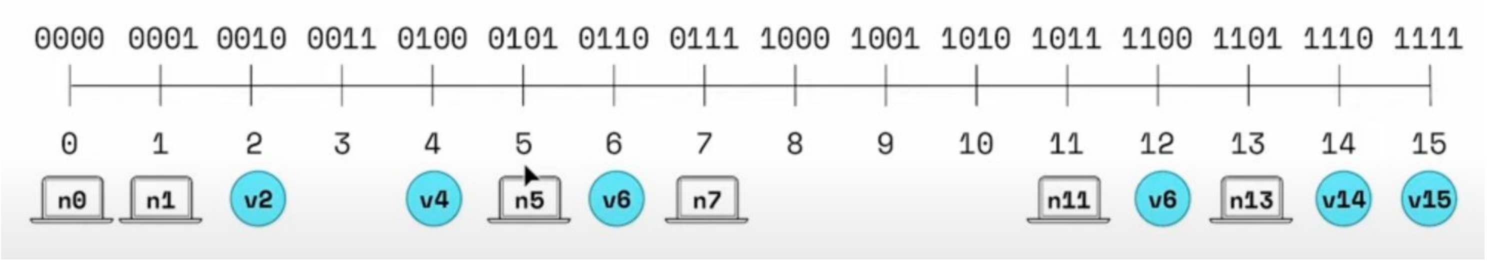
The overlay should **take the network locations of the peers into account** when deciding where data is stored, and where messages are sent, in order to **minimize networking overhead**

DHT


DHT API
Distributed applications
put(key, value)
get(key)
value
delete(key, value)
Distributed hash table (DHT)
Node
Node
Node
Node
DHT balances keys and data across nodes

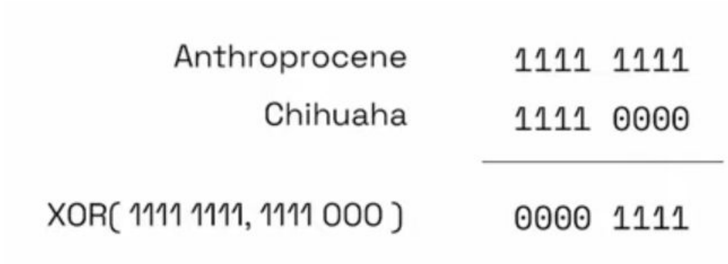
Kademlia

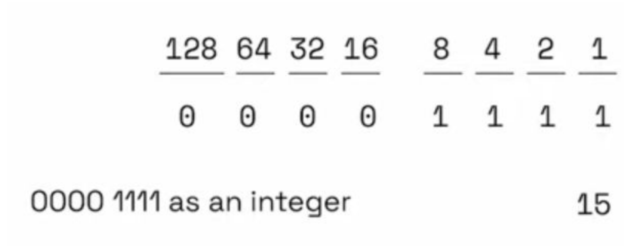
- One the mostly deployed solution for DHT, used for example in Ethereum and IPFS
- Structured Overlay Network
- Both Identifiers and Keys are 160 bit
- Several refinements have been proposed



Kademlia

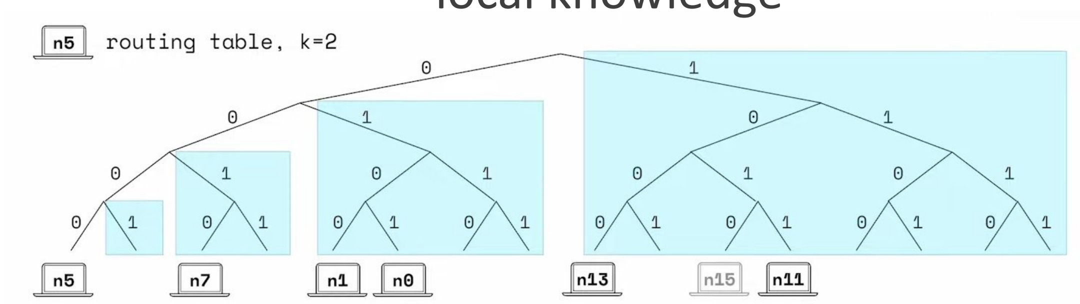
- Based on XOR distance





In the **routing table** each peer stores information (Node ID, UDP port, IP address) about other **peers located at the distance from 2i to 2i+1** from itself (0 < i < 160), organized in lists (**k-buckets**)

local knowledge



$\log_n$ - hop routing

References

- M. V. Steen, A. S. Tanenbaum - Distributed Systems - V4.01 - *Section: 2.4 Symmetrically distributed system architectures* (https://www.distributed-systems.net/index.php/books/ds4/)
- Bitcoin's dynamic peer-to-peer topology (https://doi.org/10.1002/nem.2106)
- Information Propagation in the Bitcoin Network (https://doi.org/10.1109/P2P.2013.6688704)
- [EXTRA] Kademlia, explained (https://youtu.be/1QdKhNpsj8M?si=mJutPnILIYgqR8HY)

An overlay network is a network build on existing phisical network. The links are said to be logical or virtual, as they are usually indipendent of the underlying network links and topology. This means that 2 direct neighbors in the overlay, may be separated by several hoops in the underlay. There are several classes: structured, means that nodes are organized in a specific structure like a ring; the schema is deterministic, so the position of the nodes is known a priori. While the unstructured has a random topology and the schema is impossible to predict. In the first class we have an efficient routing algorithms in locate data, but mantaining the structure is challenging with the frequent node joins. The most common example is DHT that provides the capacity to map any given key to a node that is active at the moment.
In the second class we have semplicity in the implementation and in the mantainance thanks to the lack of organization rules. Also is flexible to network changes. The routing is not efficient especially in large network.