Exercises: DHT

- Today's goal: (implement and) run a p2p network using a distributed hash table.
- DHT: **Kademlia**
- Used in file sharing networks (BitTorrent) and blockchain (Ethereum).
- `pip install kademlia`
- Similar to Chord, but uses another distance function.
- Nodes and values live in an *m* bit space.
- `distance(node1, node2) = (uint) XOR(id(node1), id(node2))`
- Keys and values are stored redundantly over the network.
- Lookup operation takes $O(\log(n))$ steps.

Kademlia

- Each node keeps a *routing table* where lists of nodes are stored.
- Routing table has m entries
  - *i*-th entry includes those nodes that have the previous *i-1* bits equal to the node that is being sougth.
  - The further into the list, the closer the nodes in the list are.
- Lookup first using the closest nodes in the list.
- If key not present, use nodes further away.
- … until key found.

Kademlia

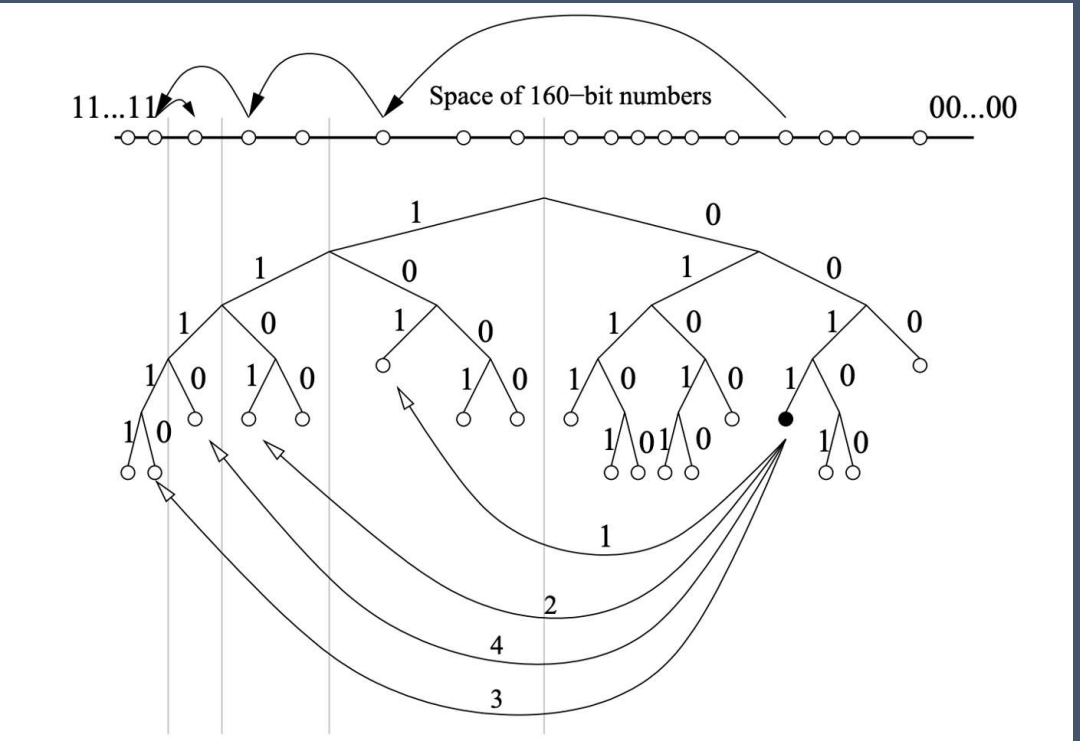


Fig. 2: Locating a node by its ID. Here the node with prefix 0011 finds the node with prefix 1110 by successively learning of and querying closer and closer nodes. The line segment on top represents the space of 160-bit IDs, and shows how the lookups converge to the target node. Below we illustrate RPC messages made by 1110. The first RPC is to node 101, already known to 1110. Subsequent RPCs are to nodes returned by the previous RPC.

Exercises

- Pull DS_Examples
- DS_Examples/p2p/p2pnode_exercise.py

1. Implement a coroutine (async) that waits for user input and implements two operations
   a) Set the value for a given key (Server.set)
   b) Get the value of a given key (Server.get)

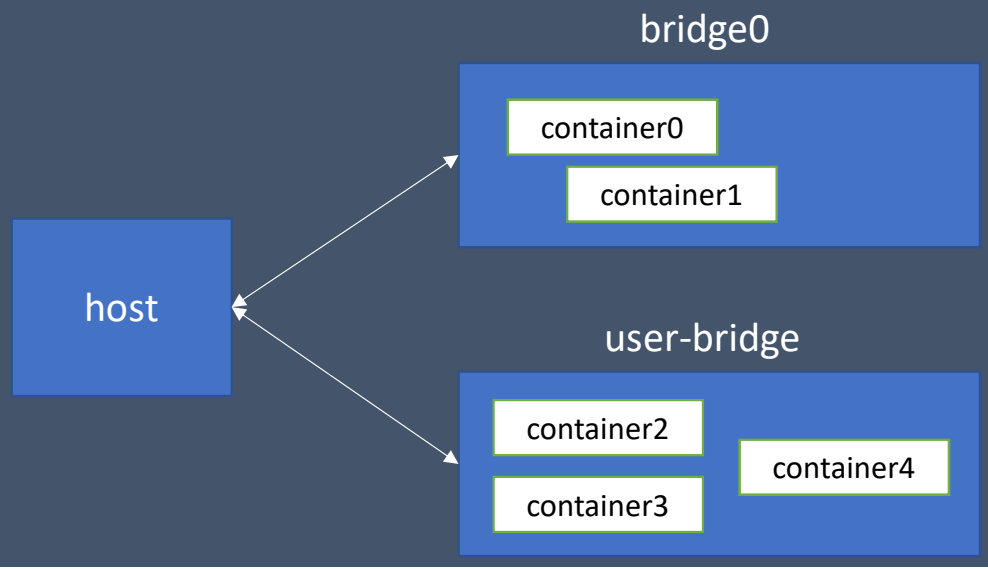
   Hint: Use the aioconsole package and the ainput function

Execises

2. Modify the program so that in case that no parameters are given for "`join`", the IP address of the neighbour will be read form the `P2P_REMOTE_HOST` environment variable
   - If this variable is empty, then a "create" and not a "join" will be made
   - **Hint:** Environment variables are stored in the os.environ dictionary
3. Write a `Dockerfile` that exposes the default port

Docker: Advanced Networking

- We will create a user network for our p2p network
- Types of networks
  - Bridge: default type
    - Containers in a bridge network can communicate between themselves, but not with containers in other bridge networks
    - User-created bridge networks provide name resolution



Docker: Advanced Networking

- Bridge networks can only communicate inside the same Docker daemon
- Overlay networks: containers can communicate with other Docker daemons
- Macvlan: can assign MAC addresses to containers
- None: container runs isolated
- Third party plugins: custom functionality

Exercises

4. Create your own p2p network (p2p_setup.md) and set and get values on the network