Exercise
for Lecture "P2P Systems"

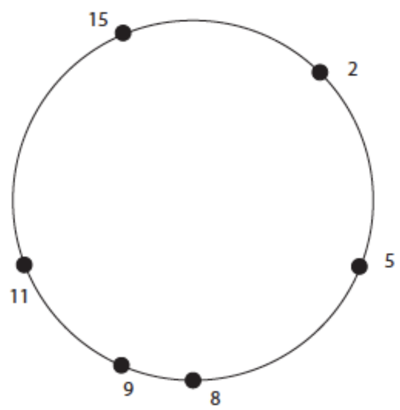
Prof. Dr. David Hausheer
Dipl.-Wirtsch.-Inform. Matthias Wichtlhuber, Leonhard Nobach, M. Sc., Dipl.-Ing. Fabian Kaup, Christian Koch, M. Sc., Dipl.-Wirtsch.-Inform. Jeremias Blendin

TECHNISCHE UNIVERSITÄT DARMSTADT

Summer Term 2015
Exercise No. 2
Published at: 28.04.2015, Submission date: 26.05.2015
Submission only via the Moodle platform in PDF, plain text, or JPG/PNG.
Contact: `[mwichtlh|lnobach|fkaup|ckoch|jblendin]@ps.tu-darmstadt.de`
Web: `http://www.ps.tu-darmstadt.de/teaching/p2p/`
– Example Solution –

Problem 2.1 - Pastry Addressing

The following address space ring for this problem is given. The markers represent the available peers in the Pastry DHT. (Note: Although hexadecimal notation is common in Pastry, in this figure, we use decimal notation for simplification.)



A) How many bits does an identifier (NodeID) have in this address space, and what is the largest possible address?

Solution: Since the address space must be a power of 2, the (minimal) number of bits in this example is 4 ($15 \leq 2^4$) and the largest possible address is $2^4 - 1 = 15$.

B) Let the DHT parameter b to be configured to 2. How many digits are then used for node and item ids? What would be the according string representations of the nodes in the above example?

Solution: With $N = 16$ being the size of the address space:
Number of digits per id: $log_2(N)/b = log_2(16)/2 = 2$
String representations (by representing the original decimal number with the base of $2^b = 4$): $2_{10} \rightarrow 02_4$, $5_{10} \rightarrow 11_4$, $8_{10} \rightarrow 20_4$, $9_{10} \rightarrow 21_4$, $11_{10} \rightarrow 23_4$, and $15_{10} \rightarrow 33_4$.

Problem 2.2 - Pastry Routing Complexity

Given is a Pastry DHT with the following properties:

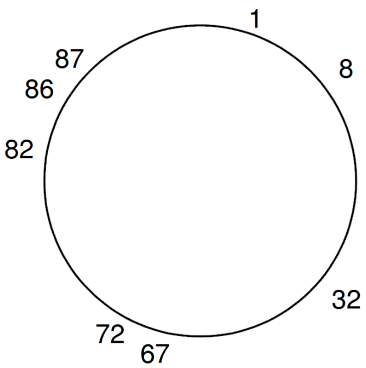
$\lvert L \rvert = 0$,	the leaf set is empty.
$N = 2^k, k \in \mathbb{N}$,	the number of nodes in the DHT is a power of 2.
b,	is an arbitrary constant value.

Pastry's routing algorithm converges in not more than $\log_{2^b} N$ hops. Prove the theorem. *Hint:* consider the size of the set of possible target nodes during the routing process.

Solution: The message is forwarded to a node who shares a prefix that is at least one digit (b bits) longer than the prefix that the key shares with the current node. If a message is forwarded using the routing table only ($\lvert L \rvert = 0$), the set of nodes whose ids have a longer prefix match with the key is reduced by a factor of at least 2^b in each step. Routing ends when there is only one node left in the set: $\frac{N}{2^{bi}} = 1 \iff \log_{2^b} N = i$, where i is the maximum number of hops to be performed.

Problem 2.3 - Chord network

Consider the Chord network shown in the figure. In this network, 8 nodes participate having the following Globally Unique Identifiers (GUIDs): 1, 8, 32, 67, 72, 82, 86, 87.



A) How many fingers are needed if the GUID range is between 0 and 99? Which formula provides the i-th finger of node n? Provide the fingers table for node 82. Which is the responsibility area of node 82 in this Chord network?

Solution:

7 fingers are needed to cover the identifier space ($2^7 = 128 > 100$, which is the size of the address space). The ith finger for node n is given by $f_n(i) = Successor(n+2^i)$ (alternatively it can be $f_n(i) = (n + 2^i) mod 100$). The finger table is given in the following table:

finger	node
0	86
1	86
2	86
3	1
4	1
5	32
6	67

Node 82 is responsible for the identifier space (73, 82).

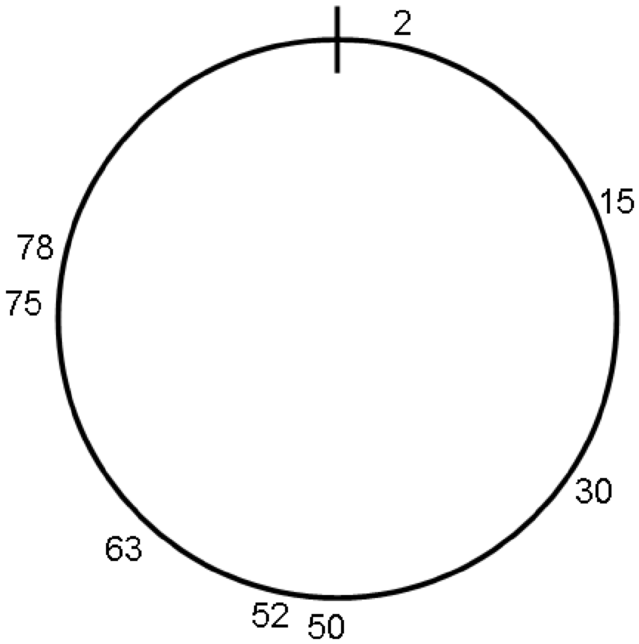
B) Node 82 is performing a lookup request with input value 7. How many steps are needed assuming that the network is stabilized? Show the followed path until the destination.

Solution:

Node 82 will forward the query to Node 1, since it is the closest one in node 82's finger table not exceeding the lookup value. Node 1 does not have any node in its finger table closer to but below value 7. Because of that, Node 1 is the final hop, although not responsible for the content: Node 1 answers to Node 82 that Node 8 (Node 1's immediate successor) is responsible for value 7.

Problem 2.4 - Chord network

Consider the Chord network with an identifier space ranging from 0 to 99 as shown in the figure. In this network, 8 nodes participate having the following Globally Unique Identifiers (GUIDs): 2, 15, 30, 50, 52, 63, 75, 78.



A) Please derive a formula for calculating the the ith finger in the routing table at node n.

Solution:

The ith finger for node n is given by $f_n(i) = Successor(n + 2^i)$ (alternatively it can be $f_n(i) = (n + 2^i) mod 100$).

B) How many fingers are needed if the GUID range is between 0 and 99?

Solution:

7 fingers are needed to cover the identifier space ($2^7 = 128 > 100$, which is the size of the address space).

C) Provide the finger table for node 50.

Solution:

7 fingers are needed to cover the identifier space ($2^7 = 128 > 100$, which is the size of the address space). The finger table is given in the following table:

finger	node
0	52
1	52
2	63
3	63
4	75
5	2
6	15

D) Which is the responsibility area of node 50 in this Chord network?

Solution:

Node 50 is responsible for the identifier space $]30, 50]$.

E) Node 50 is performing a lookup request with input value 16. How many steps are needed assuming that the network is stabilized? Show the followed path to the destination.

Solution:

Node 50 will forward the query to node 15, since it is the closest peer in Node 50's finger table not exceeding the lookup value. Node 15 is the final hop, as it has no closer peer not exceeding the lookup value in its finger table. Node 15 will directly answer Node 50 that node 30 (Node 15's successor) is responsible for the key 16. One hop is necessary to look up the responsible node.

Problem 2.5 - Structured Overlay Networks

A) Choose the right answer:

		TRUE	FALSE
i)	Search complexity of O(1) is not possible in a DHT.	☐	☒
ii)	DHTs use different address spaces for data and nodes.	☐	☒
iii)	Introducing *Virtual Servers* for load balancing helps reducing the number of nodes without any load.	☒	☐
iv)	Chord automatically replicates stored application data items among each node's list of successors.	☐	☒

B) Which of the both mechanisms (Replication and Redundancy) is beneficial if the data items stored in the DHT are (i) small or (ii) large?

Solution:

small items: replication is better (low overhead)

large items: redundancy is better (efficient usage of bandwidth and storage)

C) Certain mechanisms in DHTs (which?) require availability of many different hash functions. How can such functions be derived from an initial hash function $h_0(\cdot)$ such as *SHA-1*?

Solution: Examples are load balancing mechanisms (e.g. *power of two choices*) and replication mechanisms that store replicas on different unrelated nodes.
Given an arbitrary input string s and concatenation function $\circ$:
Naive approach: $h_{i+1}(s) = h_i(s \circ 1)$ for $i \geq 0$
More advanced approach: $h_{i+1}(s) = h(h_i(s) \circ 1)$ for $i \geq 0$
...feel free to invent/find more ...

D) How shall an identifier (NodeID) be created in a structured overlay network? How can a malicious peer exploit arbitrary NodeID creation?

Solution: NodeIDs must be unique (distinct ids for distinct nodes) and should be equally distributed over the address space (to achieve good load balancing).
NodeIDs should be also verifiable, e.g. using a hash of the node's IP address or public key. Otherwise a malicious node could take over the responsibility of a desired item by generate a NodeID close to the item's ID.

E) Explain why using pointers with the *Power of Two Choices* load balancing algorithm implies an increased overall load for the system?

Solution:

- More than one node has to be informed about pointers on inserting data (at least one additional message per pointer).
- In case of joining/leaving peers, additional messages have to be exchanged to update pointers if changes in the nodes' responsibility areas occur.

F) Explain the steps of Chord's "Stabilize" function in case a node A does not know about its newly joined successor in the ring. Assume that all predecessor relations are up-to-date.

Solution:

- Node A asks its currently known successor about its current predecessor (this should be A itself, assuming no join).
- Node A compares the answer with its own ID, detects that they differ and updates its successor.
- During the next stabilize operation (it is executed periodically), either an additional new node will be discovered or the successor's predecessor is A itself and this part of the ring is consistent.