Peer-to-Peer Systems
Exercise 2
Published: 05.11.2013
Presentation date: 13.10.2013

Peer-to-Peer Systems – Exercise
Winter Term 2014/2015

General Remarks

Welcome to the exercise for the lecture Peer-to-Peer Systems.

Please follow the general remarks regarding the organization of the exercise.

- The lecture's website is to be found here: `http://tsn.hhu.de/teaching/lectures/2014ws/p2p.html`
- For further inquiries, please contact the lecturer under the following email address: `graffi@cs.uni-duesseldorf.de`

Problem 2.1 - Strange DHT Topologies

Let us assume a DHT, which is built the following way: Currently, n peers are in the network. The n peers are enumerated from 0 to $n-1$, each peer knows its ID. The object IDs are also natural numbers, as hash function we use:

$$h : \mathbb{N} \to \{0, \ldots, n-1\}, k \mapsto k \bmod n$$

The responsibility function M is given as: $M(i) = h(i)$. I.e. the peer with id i is responsible for the object IDs k with $h(k) = i$.

a) Quality Rating

Explain, why this is *not* a good basis for a DHT. Name at least two fundamental issue.

Solution:

A) With nodes leaving or joining, a completely new object to node assignment is given. Thus, every small change in the node set requires a large change in the mapping of the objects.

B) Enumerating the nodes, including new nodes, is not an easy task. In cases when two nodes are joining the network at similar time at different positions in the network, a negotiation one the node IDs is required.

b) Routing in Strange DHTs

Let us assume, that despite of your good arguments, the overlay exist. There are two variants for constructing the network topology:

Variant 1:

All peers are sorted according their IDs and connected to their predecessors and successors (mod n). Thus peer i is connected to peer $i-1$ and $i+1$ (mod n). Routing is done through forwarding lookup requests to the neighbor closer to the queried ID.

Variant 2:

In this variant, each peer is connected to the peer with ID 0. No further connections exist. Routing is done by forwarding lookup requests to peer 0, which forwards the lookup request further to the target peer.

A) What is the routing complexity of both variants, i.e. the (asymptotic) average number of hops until the lookup request reaches the responsible peer?

B) What is the worst-case state complexity of both variants, i.e. the the (asymptotic) maximum number of contacts, a node has to maintain.

Solution:

Routing complexity:

A) Variant 1: $O(n)$, as typical 1/4 of the ring needs to be passed.

B) Variant 2: $O(1)$, as only 2 hops are needed at most

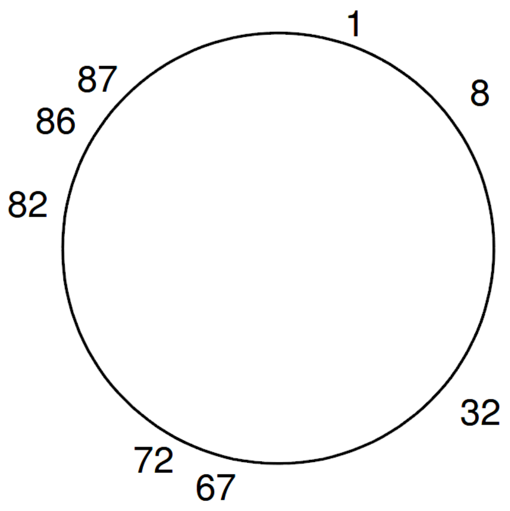
State complexity

A) Variant 1: $O(1)$, as each peer knows exactly 2 other peer.

B) Variant 2: $O(n)$, as peer 0 needs to keep contact to all other peers.

Problem 2.2 - 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) Chord Topology

How many fingers are needed if the GUID range is between 0 and 99? Which formula provides the ith finger of node n? Provide the fingers table for node 82 according to the format finger, target id and node id. Give the responsibility areas of all nodes in this Chord network according to the format (Peer ID, From, To).

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	Target ID	Node ID
0	83	86
1	84	86
2	86	86
3	90	1
4	98	1
5	14	32
6	46	67

Tabelle 1: Chord Finger Table

Peer ID	From	To
1	88	1
8	2	8
32	9	32
67	33	67
72	68	72
82	73	82
86	83	86
87	87	87

Tabelle 2: Chord Responsibility Area

b) Routing in Chord

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 peer not exceeding the lookup value. Then, Node 1 will forward the query to Node 8, which is responsible to provide the final answer. 2 steps are needed to forward the query to the final destination.

Problem 2.3 - Content Addressable Network - CAN

In this exercise, with have a look at CAN.

a) Graph Properties

Let us consider a CAN - network with an generalized ID space $[0,1)^2$. The 2-dimensional space is considered a square, you can ignore the wraparound, i.e. the space being a torus. Further assume, that the network is populated with $n = 4^i$ nodes, with $i \in \mathbb{N}$. With this in mind, the placement of the nodes is assumed to be "perfect", with all nodes having the same size of space being responsible for.

A) How large is the graph diameter, i.e. the maximum distance between two peers in hops, with respect to the number of peers n?

B) In the previous example, CAN hat a dimension of $d = 2$. Generalize d in the formula for the graph diameter and assume that the number of nodes grows correspondingly. A network with dimension d has 2^d nodes. How large is the network diameter with respect to the number of nodes n and the dimension d?

Solution:

A) The space is split in $\sqrt{n} \times \sqrt{n}$ small zones. The longest path is from one corner to the opposite corner. For that $\sqrt{n} - 1$ horizontal and $\sqrt{n} - 1$ vertical steps need to be made. Thus, the diameter is $2(\sqrt{n} - 1)$.

B) Generalizing the calculation to a d-dimensional CAN network, we have n small multidimensional zones. The longest path is again from the zone in one corner to a zone in the completely opposing corner. Along each dimension, we have $\sqrt[d]{n}$ zones. Thus the path has to go along all d dimensions to the "other" side, requiring each time $\sqrt[d]{n} - 1$ steps. In total the diameter is $d(\sqrt[d]{n} - 1)$. Asymptotically, this is the routing complexity which was named in the lecture: $O(dn^{1/d})$.

b) Defragmentation in CAN

Let us consider the cases, in which a peer leaves the CAN network and its responsibility area needs to be assigned to another node. Sketch a situation, in which the reassignment creates a complex case in which one single defragmentation step is not sufficient: A single node with several zones, out of which the neighboring zone of its smallest zone is itself split.

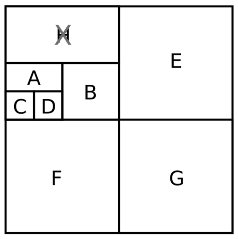
A) Draw a CAN network with such a situation, before and after the leaving of the node.

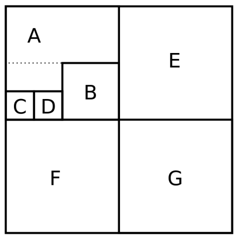
B) Draw a corresponding tree diagram of this CAN network, after the node left.

C) Give a protocol (messages and communications), that describes how step by step a consistent valid CAN state can be reached in which every peer is only responsible for one zone.

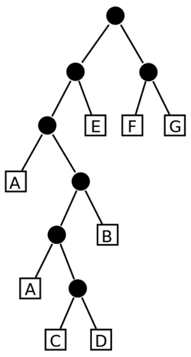
Solution:

A) CAN network





B) Corresponding CAN tree



C) Defragmentation protocol (example):
In the following we call the nodes in a tree which are have the same parent node as *counterpart*.
A contacts one of the (two in this case) counterparts of its smallest zones, in this case C or D. Please note, this step addresses a node one level deeper in the tree.
Each contacted node (C or D) checks whether its counterpart in the tree has a complete, i.e. not split, zone. This step performs on the same level in the tree.
If it is not the case, i.e. the zone of the counterpart is split, then one (of the two) neighboring nodes are contacted. Both of these possible nodes are again one level deeper in the tree. Recursion is applied here.
Eventually, a node (e.g. X) will be reached which has a counterpart (e.g. Y) which is complete. Contact details to both nodes (e.g. X and Y) will be sent to A.
A asks one of the nodes (e.g. X) to take over the zone of the other node (e.g. Y). The other node (e.g. Y) is asked to take over the smallest of A (the problematic one).
Corresponding zone information (objects and neighboring information) is transferred. Here: from A to Y and from Y to X. The corresponding neighbors are informed about the new topology.