Internet and Web Applications, Übung 6

Henry Haustein

Aufgabe 1: Peer-to-Peer networks

First generation:

- Centralized P2P: Central directory server is the most important communication entity It makes lists of files with their associated peers available. Decentralized file transfer
- Decentralized P2P: The decentralized network has only peers. Peers called servents (server + client)

Second generation

- Hybrid P2P: Dividing servents into SuperNodes and LeafNodes. Client software makes servents to SuperNodes by means of specific criteria (i.e. connection speed). SuperNodes are temporary servers. Every SuperNode can manage approx. 200-500 leafs. Reduce message transfer by Flow Control and Pong caching. Bootstrapping by WebCaches (directory servers) and HostCache

Third generation

- Structured P2P: Based on Distributed Hash Table (DHT). Guaranteed correct results. Quick search with logarithmic overhead $\mathcal{O}(\log(n))$

Aufgabe 2: Kademlia routing algorithm

(a) Node 1100:

- k-Bucket distance 0: 1100
- k-Bucket distance 1: 1101 (letztes Bit geändert)
- k-Bucket distance 2 and 3: 1111, 1110 (vorletztes Bit geändert)
- k-Bucket distance 4 and 7: 1010, 1001 (zweites Bit geändert)
- k-Bucket distance 8 and 15: 0110, 0011 (erstes Bit geändert)

Node 0011:

- k-Bucket distance 0: 0011
- k-Bucket distance 1: 0010
- k-Bucket distance 2 and 3: 0001, 0000
- k-Bucket distance 4 and 7: 0110
- k-Bucket distance 8 and 15: 1100, 1111

Node 0110:

- k-Bucket distance 0: 0110
- k-Bucket distance 1:
- k-Bucket distance 2 and 3:
- k-Bucket distance 4 and 7: 0011, 0010
- k-Bucket distance 8 and 15: 1001, 1100

Node 0010:

- k-Bucket distance 0: 0010
- k-Bucket distance 1: 0011
- k-Bucket distance 2 and 3: 0001, 0000
- k-Bucket distance 4 and 7: 0110
- k-Bucket distance 8 and 15: 1010, 1001

(b) $1100 \oplus 0001 = 1101 = 13 \Rightarrow$ max. Steps $\log_2(13) \approx 4$

(c) Node 1100 berechnet die Entfernung zu Node 0001. Da schon das erste Bit anders ist, schaut Node 1100 in seiner k-Bucket-List mit Entfernung 8-15 nach und findet 2 Nodes: 0110 und 0011. An beide Nodes wird eine Anfrage geschickt und beide Nodes schauen in ihren entsprechenden Bucket-Lists nach, ob sie eine Node kennen, die näher dran ist. Node 0110 sendet die Nodes 0011 und 0010 zurück, während Node 0011 die Nodes 0001 und 0000 findet und an 1100 zurück sendet. Damit kennt nun auch 1100 die Node 0001.