Reading Report #12
Paper: Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications
Student: Shuo Yang

The strengths of Chord include *highly efficient search*, *good load balancing* and *completely decentralized design*. I would also like to point out some disadvantages of the design of Chord.

First, Chord does not consider physical location of nodes in their search ring design. Each node directs its lookup request based on its routing table (or finger table), without searching for the desired data in the nodes that are in its geographical range first. This may increase lookup time and waste network bandwidth significantly, and make the performance of Chord much less predictable. A node in Boston may have its successor node in Seattle, where it could have chosen a node in New York as its successor if Chord took geographic location into design consideration. To fix this drawback, we could redesign Chord to considers physical location of nodes present in the network while still maintaining other Chord's properties in order to minimize the real distance messages travel.

Another disadvantage of Chord is its asymmetric lookup, that is, lookup from a node to another node could take a significant different number of hops than vice versa. Considering a very large Chord ring where node q precedes node p. If node p wants to look up node q, it has to travel over the complete ring to reach to p, while it only takes one hop for p to reach q if p is able to lookup backwards to its preceding nodes. To address this inefficiency, we could have two finger tables, one for looking up succeeding nodes, another for looking up preceding nodes. In doing this node in the Chord ring will have the capability of searching both clockwise and counterclockwise. Upon receiving a lookup request, a node should first determine which direction produces a shorter path to the target node before sending out lookup requests.

Because Chord is basically a distributed hash table, it can only support exact match queries. But in reality, keyword-based queries is more popular, Chord has no way to support this. Chord also does not take advantage of the popularity of files being searched. One possible solution is to rank the popularity of each file and let the nodes in the ring cache those nodes with highly ranked files to speed up search. The final disadvantage for Chord is that all nodes must have IDs, thus Chord does not allow anonymous nodes to join the ring. This brings concern about security. What if some node does not want others to know its IP? Chord has no way to hide the identify of the nodes.