**Cloud Computing Fall 2019**

**Reading Assignment2 - Greeshmika Korrapati (U00932594)**

**Summary:**This paper predominantly depicts a solution for basic complication of identifying location of node that has required data. When compared to other peer to peer protocols this approach associate’s keys with respective data and stores this key-data pair at node where key is mapped. Chord changes effectively as node enters and leaves system although system is constantly switching. Chord is scalable and distributed lookup approach.

**Approaches and Challenges:**

**Balancing Load:**

Most of other approaches are basically using equal loads but chord balances load by keeping track of key-data and respective nodes table, also keys are apportioned on nodes which creates a natural balancing of load.

**Decentralization:**  
This implies that there is no predominance for particular node when compared to others but this leads to increase complications in locating keys but this problem is addressed by building information of routing nodes with help of O(log N) and when details are not updated execution declines gradually.

**Scalability:**  
Chord searches a key from its hashing table and finds out node with help of performing log operation on number of nodes, so this does not take much effort and cost since it maintains information of nodes that join and leave system. Overall this helps searching performance of system and scalability of system.

**Availability:**

Chord updates its hashing table with newly joined keys, failed keys and it can track a lost key and will be able to get data from key even when there is abrupt change. This helps network available highly.

**Flexible naming:**

Chord has a flat key-space, so this permits application selects their ways to map their names to keys of chord, this basically gives choices for applications for utilizing their own conventions. This helps application to avoid additional name translation protocols for mapping in chord.

**Strengths:**

The major strengths of this approaches are:

1.It has minimal complicated routing process which allows node in network to just know about log N number of nodes instead of knowing every node in the network

2. Required data can be easily identified as it has distributed hashing table data can be obtained by matching its key and node from the table

3.It also reduces time for querying even though there is continuous flow of data as it stores even failure nodes data.

4. Scalability and load balance is obtained by maintaining routing tables and continuous hashing.

5. Simulations are conducted with variations to check exactness and performance of chord.

**Weakness:**

There are few drawbacks that are not explained in detail or provided proper solutions in this paper. They are:

1.As efficiency of routing depends upon information of O(log N) nodes information but when it is not updated properly performance highly decreases. As in network it is usual that node joining and leaving will be very frequent so updating data would be a considerable issue. Though it is taken care partially there is no complete solution provided to avoid this issue.

2.Another drawback is about malicious chord node which author discussed in future works, but security is a predominant factor to be considered and in peer-to-peer mechanism it would be more frequent for such attacks so it would be more efficient if he had provided solution for this in paper.

3.Network topology and location are not taken into consideration when performing routing. This might cause some performance delay.