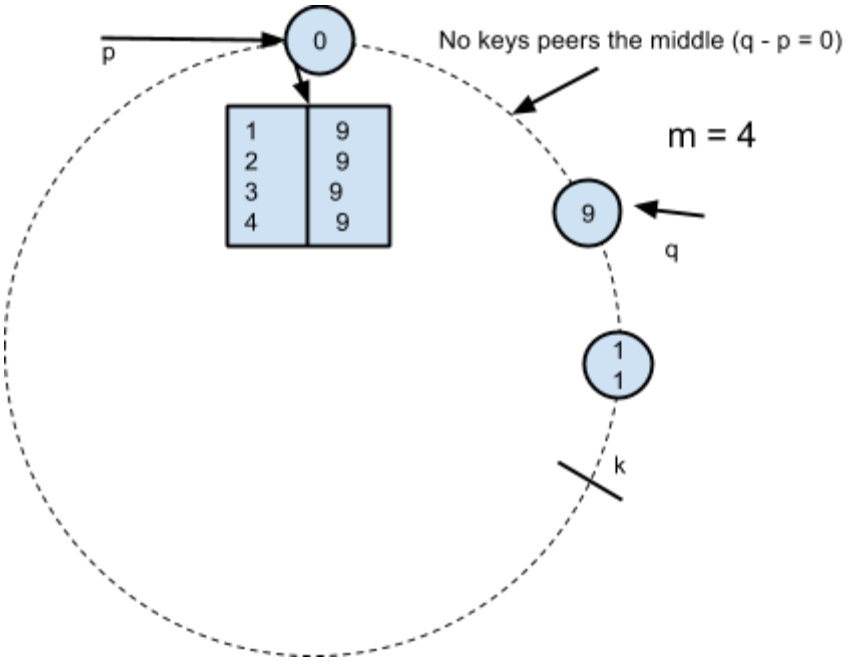
Yash Malik (20379044)
Piyush Gadigone (20366910)

Assignment 2

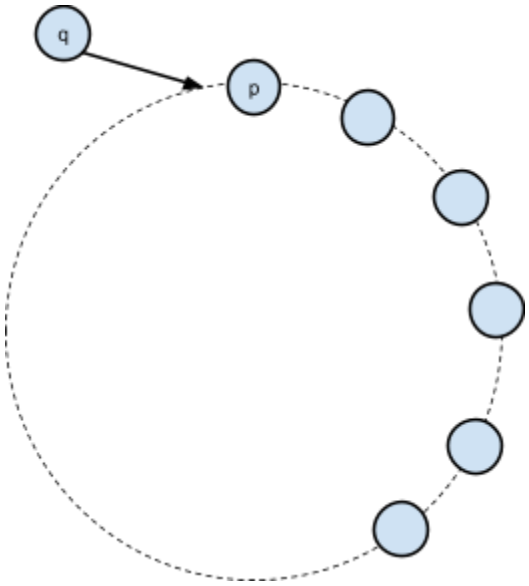
1. Disprove $q - p \geq (k - p) / 2$



$q - p \geq (k - p) / 2$ (*given*)
$0 \geq (k - p) / 2$, *because no peers between q and p*
$k - p = 2$, *as 2 peers between k and p, thus*
$(k - p) / 2 > 0$, *thus proven.*

2.

(a) The worst case number of finger tables that need to be updated is n. Consider the following scenario:



Consider a Chord DHT where all the peers have keys less than 2^(m-1) i.e. the peers are on right half of the circle. In this case, the last entry in the finger table for all the peers point to p as the last entry is succ(peer + 2^(m-1)).

Let the nth peer (q) be inserted right before p. In this case, the finger tables for all n-1 nodes have to be updated because their last entry is changed to point to q from p.

Yash Malik (20379044)
Piyush Gadigone (20366910)

(b) In this case, each peer keeps track of only one peer that is succ(peer + 1). Hence, when we add an nth peer, only the (n-1)th peer's entry will have to be updated i.e. only 1 finger table will have to be updated.

3. For an m-bit identifier, consider number of peers = 2^m. Take a peer p and look for p's predecessor i.e. k = p - 1. As proved in the lecture slides, the number of jumps to peer k will be log(n) = log(2^m) = m. Hence, for every m >= 1 we can find a case where the worst case is atleast a linear function of m.

4. Consider an nxn matrix where the coordinates go from (1,1) to (n,n). We have to map this using an invertible function to a Chord DHT of size n^2 - 1. Let *x* and *y* denote the coordinates from the matrix. An invertible function that can be used is f(n) = (x-1)n + y-1.

For ex: (1,1) would map to Chord DHT of (1-1)n + 1-1 = 0
(n,n) would map to Chord DHT of (n-1)n + n-1 = n^2 - 1

You can used this same function to get back the original coordinates from Chord DHT by using 1 <= x <= n and 1 <= y <= n as the constraint and solving for an x and y.

5.
Given that P and Q are both neighbors of R, the probability that P and Q are neighbors of each other is given by :

$$Pr(P \text{ and } Q \text{ are neighbors}) = \frac{\#\text{ of ways for } P \text{ to pick } Q \text{ as a neighbor}}{\#\text{ of ways for } P \text{ to pick } c-1 \text{ neighbors}} = \frac{(N-3)\ Choose\ (N-2)}{(N-2)\ Choose\ (C-1)}$$

6.
Bully
Considering the situation where there are $n = 2^m$ nodes in the Chord DHT system. To practice the worst case scenario, say that the peer with the smallest priority initiates an election. Then, the total cost of messages in the first round of election is $(n-1) + (n-1)$. Here the first term comes from the node with the lowest priority sending the ELECTION message to all other peers with higher priority, and all of them responding with OK (all are of higher priority). Then, the peer with the second smallest priority sends the ELECTION message to n-2 peers and all n-2 peers respond with OK. In the end, the peer with the highest priority sends a COORDINATOR message to n-1 peers. Using this, we get the following terms:

$$\begin{aligned} Total\ Messages &= [(n-1)+(n-1)] + [(n-2)+(n-2)] + [(n-3)+(n-3)] + \ldots + [(n-n)+(n-n)] + n - 1 \\ &= 2(n)\left(\frac{n(n-1)}{2}\right) + n - 1 \\ &= n(n-1) + (n-1) \end{aligned}$$

$= (n + 1)(n - 1)$
$= (n^2 - 1)$

$Cost = Total\ Messages * c\ (where\ c\ is\ the\ cost\ of\ one\ message) = c(n^2 - 1)$
This, in a situation where $n = 2^m$, the worst case cost is quadratic to the number of peers = O(n)

Ring
The worst case in Ring Election is when all the n peers send a message at the same time. Each peer attaches its ID to the message increasing the cost by c. Then the message returns back to the initiator with all process ids. Next, The initiator then sends a COORDINATOR message containing n process ids around the network which goes through the entire circle.

Cost of ELECTION message = c(1 + 2 + … + n) * n = cn*(n(n+1)/2)
Cost of COORDINATOR message = cn^3 as the message of size n, and n nodes are sending the messages and each of the n nodes is making a copy and forwarding the message.
Total cost = cn^3 + cn*(n(n+1)/2) = cn^2*(3n+1)/2

In this solution we assume that the cost of the message is not constant, it increases linearly as more id's are added to the message.

7.
When a peer p, initiates an election message, it sets a local flag, `ElectionSent` to `true`. It then forwards a list with its priority to the next peer. When p receives an election message from a peer with smaller priority than itself, it checks whether its local `ElectionSent` is set to true. If so, it drops the message, otherwise, it forwards the message after adding it's own priority in the list. When p receives its own election message, it sets the `ElectionSent` flag back to false. This algorithm ensures that unnecessary messages from peers with a lower priority are dropped.