14-12-20

Leaky Bucket Algorithm

Dhruv Agrawal
1BM18CS028

Network Layer - Congestion Control algorithm
Traffic shaping: controls the rate at which packets are sent in network

Two types of traffic shaping
1 Leaky Bucket
2 Token Bucket

Traffic Shaping is a mechanism to control the amount and rate of the traffic sent to the network. Approach of congestion management is called Traffic shaping. Traffic shaping helps to regulate rate of data transmission and reduces congestion

So, Leaky Bucket can smoothout bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate

We use a FIFO queue to hold the packets
1 Fixed size packets
2 Variable size packets

In each clock tick, some number of packets are removed from queue to transmit over the network to get smooth ~~transmit~~ transmission at average rate

```
import java.io.*;
import java.util.*;

class LeakyBucket {
    public static void main(String[] args) {
    int no_of_queries, storage, output_pkt_size;
    int input_pkt_size, bucket_size, size_left;

    storage = 0; // initial packets in the bucket
    no_of_queries = 4; // Total number of time, bucket content is checked
```

```
bucket_size=10; // Total number of packets th
                   can be accomodated in bucket
input_pkt_size=4; // number of packets that
                     enters the bucket at a
                      time
output_pkt_size=1; // number of packets that
                      enters the bucket at a
                       time
for(int i=0; i<no_of_queries; i++)
{
    size_left= bucket_size - storage;
    if(input_pkt_size <= (size_left))
    {
        storage += input_pkt_size;
        System.out.println("Buffer size=" + storage +
          "out of bucket size=" + bucket_size);
    }
    else
    {
        System.out.println("Packet Loss ="
          + (input_pkt_size - (size_left)));

        storage= bucket_size // Full size
        System.out.println("Buffer size=" + storage
          "out of bucket size=" + bucket_size);
    }
    storage -= output_pkt_size;
  }
}
```