# Advance Computer Networks (CS G525)


Virendra S Shekhawat
Department of Computer Science and Information Systems

BITS Pilani
Pilani Campus

innovate achieve lead

BITS Pilani
Pilani Campus

# First Semester 2018-2019
# Lecture: 12-14 [10-14 Sept 2018]

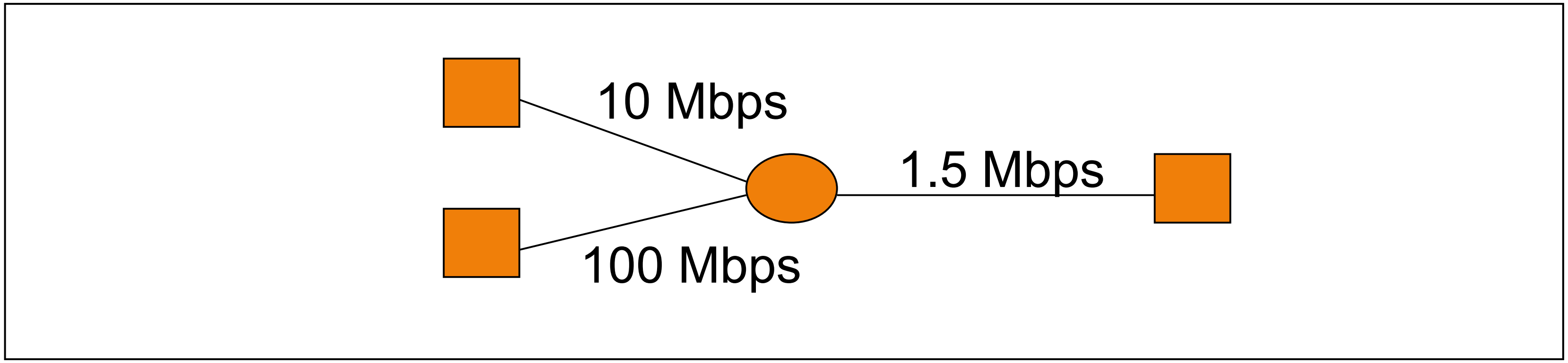
# Next Module Topics

- Internet congestion control principles
  - End to End based mechanisms
    - e.g., TCP
  - Network assisted congestion control
    - e.g., RED, FQ, CSFQ etc.
- New proposals on end to end congestion control
    - e.g., QUIC, BBR
- Multi-path TCP design and implementation
- Role of traffic routing for Internet congestion control
  - Inter-domain and Intra-domain routing protocols
- Multicast Routing
  - IP Multicast and Application level Multicast routing

# Agenda

- Internet Congestion Control Principles, Congestion Avoidance and Control
  - TCP as an example
- Compulsory Reading:
  - Congestion Avoidance and Control [Jacobson 1988]

# What is Congestion…?




- Why is it a problem?
  - Different sources compete for resources inside network
  - Sources are unaware of current state of resource
  - Sources are unaware of each other
  - In many situations will result in < 1.5 Mbps of throughput (congestion collapse)

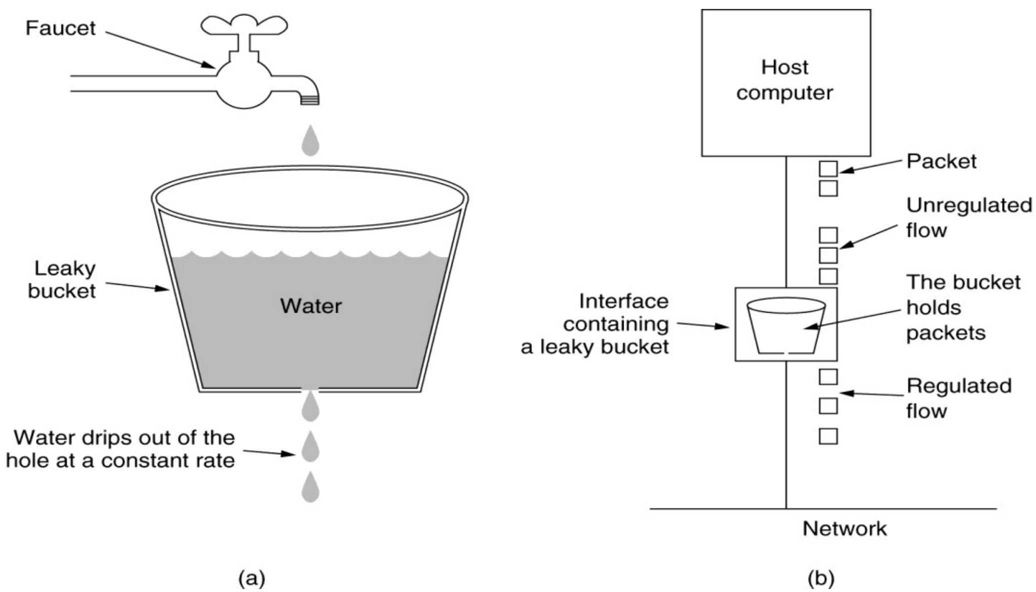
# Solutions

- Open Loop
  - Preventing congestion (Implemented at end points)
  - Policy and scheduling decision making is not based on the current state of the network

- Closed Loop
  - Detect, Feedback and Correct
  - Explicit or Implicit Feedback

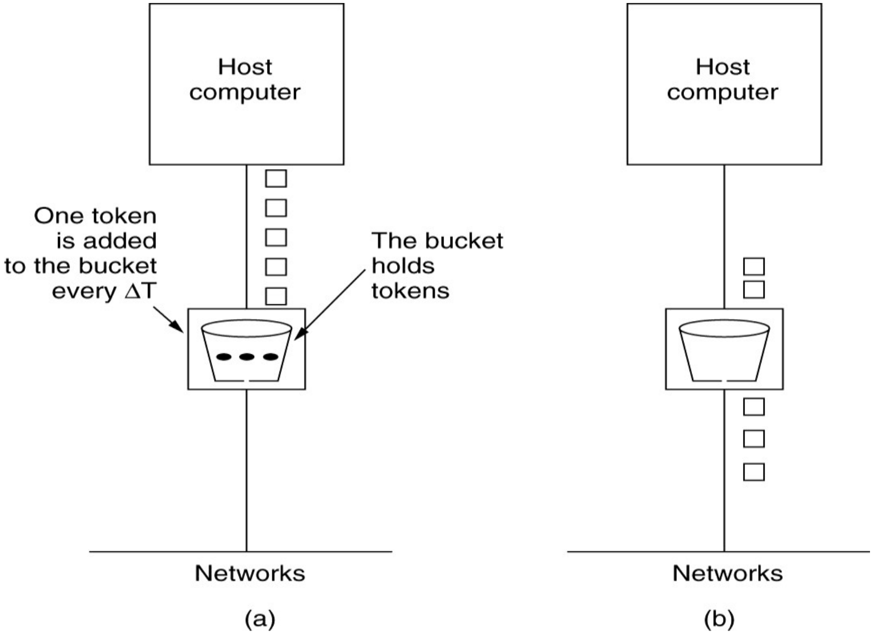
# Open Loop Solutions [1]

- Leaky Bucket
  - When a packet arrives, if there is a room in the queue it is joined the queue; otherwise, it is discarded
  - At every (fixed) clock tick, one packet is transmitted unless the queue is empty



# Open Loop Solutions [2]

- Token Bucket
  - Tokens are added at a const rate
  - A host had to capture a token to transmit a packet
  - A host can capture and save up tokens (up to the max. size of the bucket) in order to send larger burst later



# Closed Loop Solutions [1]

- Warning bit
  - A router set a special bit (e.g., ECN) to warn the source when congestion is detected.
  - The sender monitors the number of ACK packets it receives with the warning bit set and adjusts its transmission rate accordingly.
- Issues
  - Non cooperative sources…?

# Closed Loop Solutions [2]

- Choke Packet
  - Router send a choke packet to the source in case line is too busy
  - e.g., ICMPs “Source quench packet”
- Issues
  - Choke packet itself delayed…then…?
  - What about non-cooperative sources…?

# How congestion handled so far…?

- Different mechanisms at different layers
  - Transport
    - Retransmission, Out of order packet caching, Time out, Acknowledgement etc.
  - Network
    - Virtual Circuit , Packet discard policy, Packet queuing service policy,  Routing algorithm
  - Data link Layer
    - Similar to TP layer

# Where to Prevent Congestion?

- Can end hosts prevent congestion?
  - **Yes**, but must trust end hosts to do the right thing
    - e.g., sending host must adjust amount of data it puts in the network based on detected congestion

- Can routers prevent congestion?
  - No, not all forms of congestion
  - Doesn't mean they can't help
    - Sending accurate congestion signals
    - Isolating well-behaved sources from ill-behaved sources

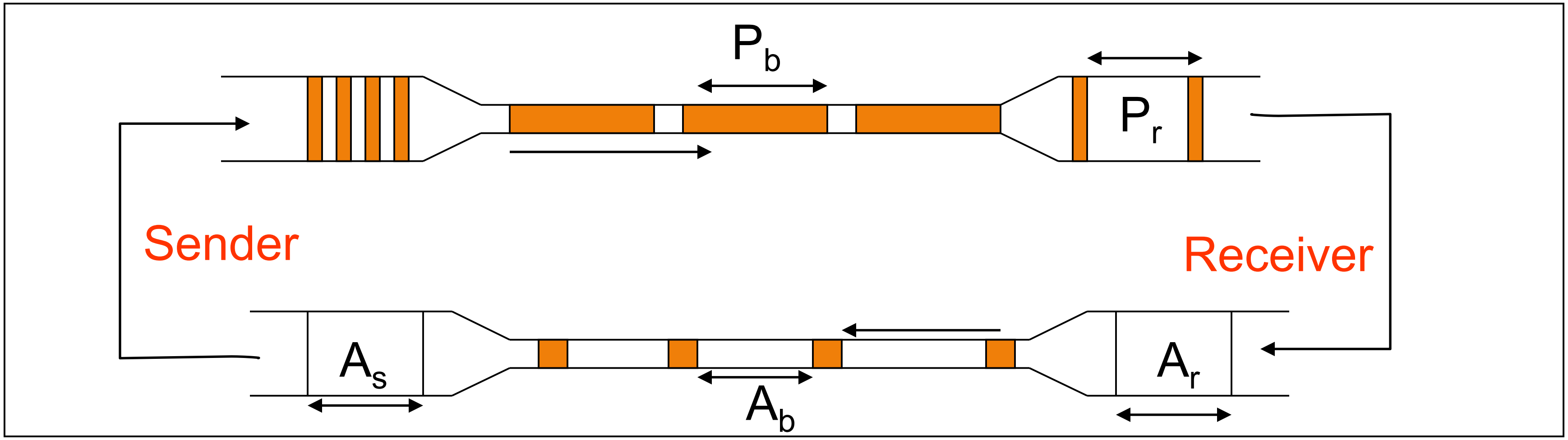
# TCP Congestion Control

- Motivated by ARPANET congestion collapse

- Basic principles
  - Self clocking
  - Reaching steady state quickly (Slow Start)
  - Packet conservation
  - AIMD

# Packet Pacing: Self Clocking

- Congestion window helps to "pace" the transmission of data packets
- In steady state, a packet is sent when an ack is received
  - Data transmission remains smooth, once it is smooth



# Conservation of Packets

- Packet conservation Principle
  - In equilibrium (**a full window of data in transit**) a new packet isn't put into the network until an old packet leaves

- Why Conservation Fails…?
  - The connection doesn't get to equilibrium, or
  - A sender injects a new packet before an old packet has exited, or
  - The equilibrium can't be reached because of resource limits along the path

# Congestion Avoidance

- If loss occurs when cwnd = W
  - Network can handle 0.5W ~ W segments
  - Set cwnd to 0.5W (multiplicative decrease)
- Upon receiving ACK
  - Increase  cwnd by (1 MSS)/cwnd
- Implements AIMD
- Question
  - Why TCP Fails to avoid congestion?

# Congestion Collapse Causes…[1]

- Spurious retransmissions of packets still in flight
  - How can this happen with **packet conservation**?
  - **Solution:** Better timers and TCP congestion control
- Undelivered packets
  - Packets consume resources and are dropped elsewhere in the network
  - **Solution:** congestion control for ALL traffic

# Congestion Collapse Causes…[2]

- Packet fragmentation
  - Mismatch of loss and retransmission units
  - Solutions
    - Make network drop all fragments of a packet (early packet discard in ATM)
    - Do path MTU discovery

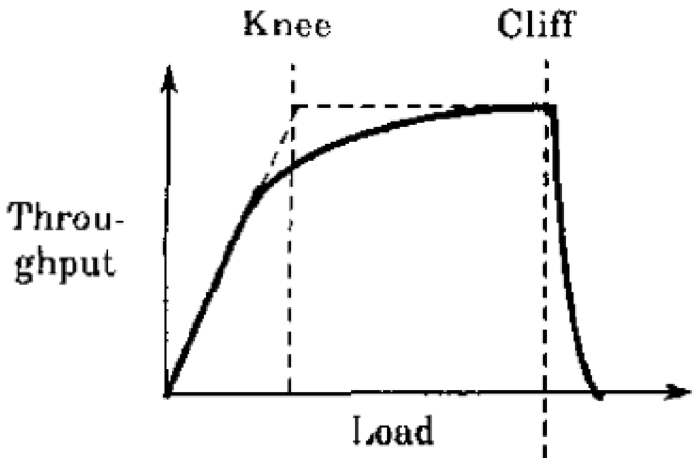
- Control traffic
  - Large percentage of traffic is for control
    - Headers, routing messages, DNS, etc.

# Next…

- Analysis of AIMD Congestion Control Algorithm
  - Compulsory Reading:
    - Analysis of Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks [Chiu 1989]
- How does TCP fits in AIMD...?
- Research Issues/Open questions in congestion control

# Congestion Control Objectives

- Key to congestion avoidance is the "control function" used to increase or decrease their sending window
  - Distributedness
  - Efficiency: $X_{knee} = \Sigma x_i(t)$
  - Fairness: $(\Sigma x_i)^2/n(\Sigma x_i^2)$
  - Convergence: control system must be stable and reach to goal state from any starting state quickly



# Basic Control Model

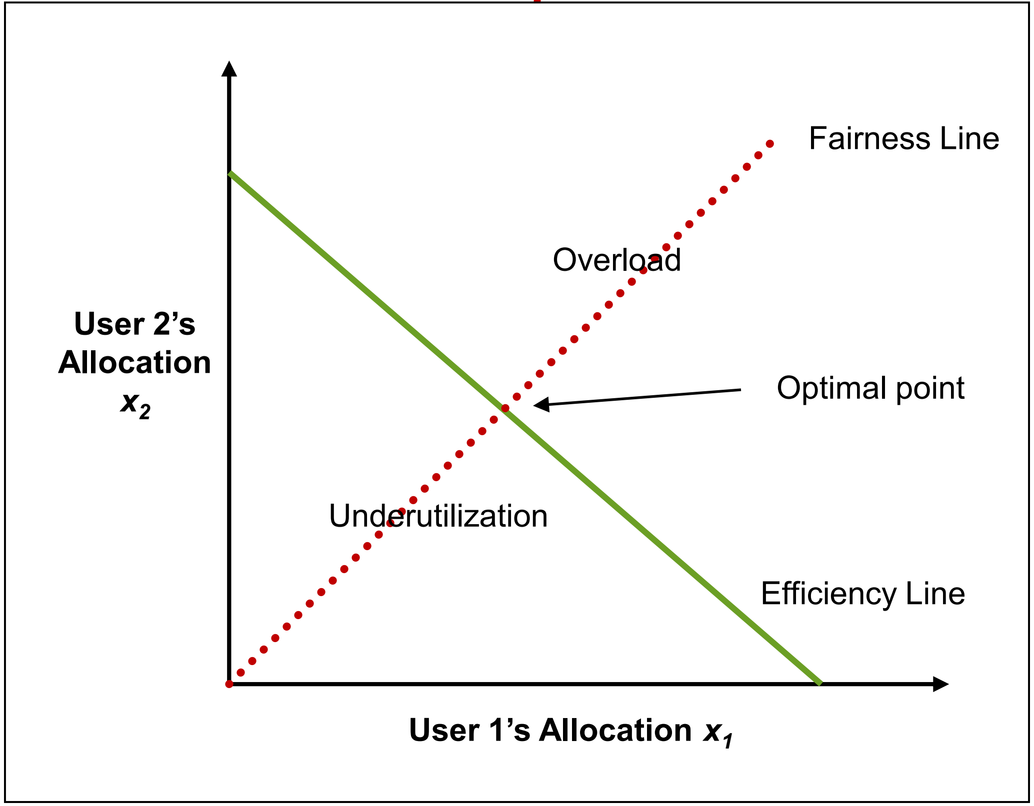
- Reduce window when congestion is perceived
  - How is congestion signaled?
    - Either mark or drop packets
  - When is a router congested?
    - Drop tail queues – when queue is full
    - RED queues - average queue length at some threshold
- Increase window otherwise
  - Probe for available bandwidth – how?

# Linear Control

- Many different possibilities for reaction to congestion and probing
  - Examine simple linear controls
  - Window(t + 1) = a + b*Window(t)
  - Different $a_i/b_i$ for increase and $a_d/b_d$ for decrease
- Supports various reaction to signals
  - Increase/decrease additively
  - Increased/decrease multiplicatively
  - Which of the four combinations is optimal?

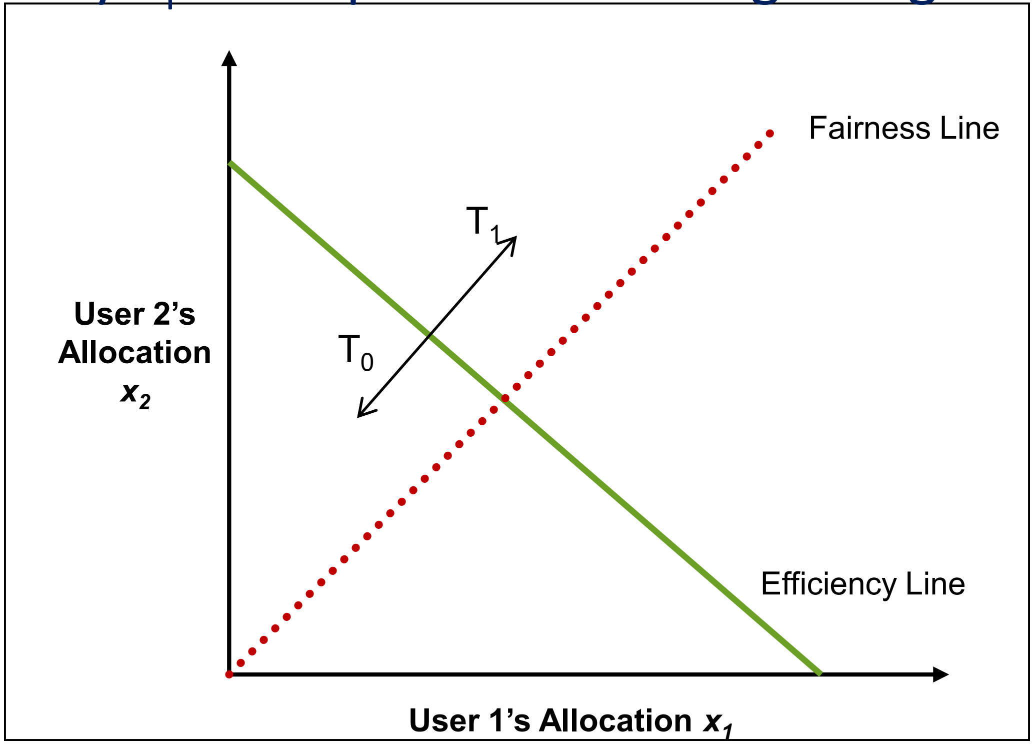
# Phase plots (Vector Representation)

- What are desirable properties?
- What if flows are not equal?



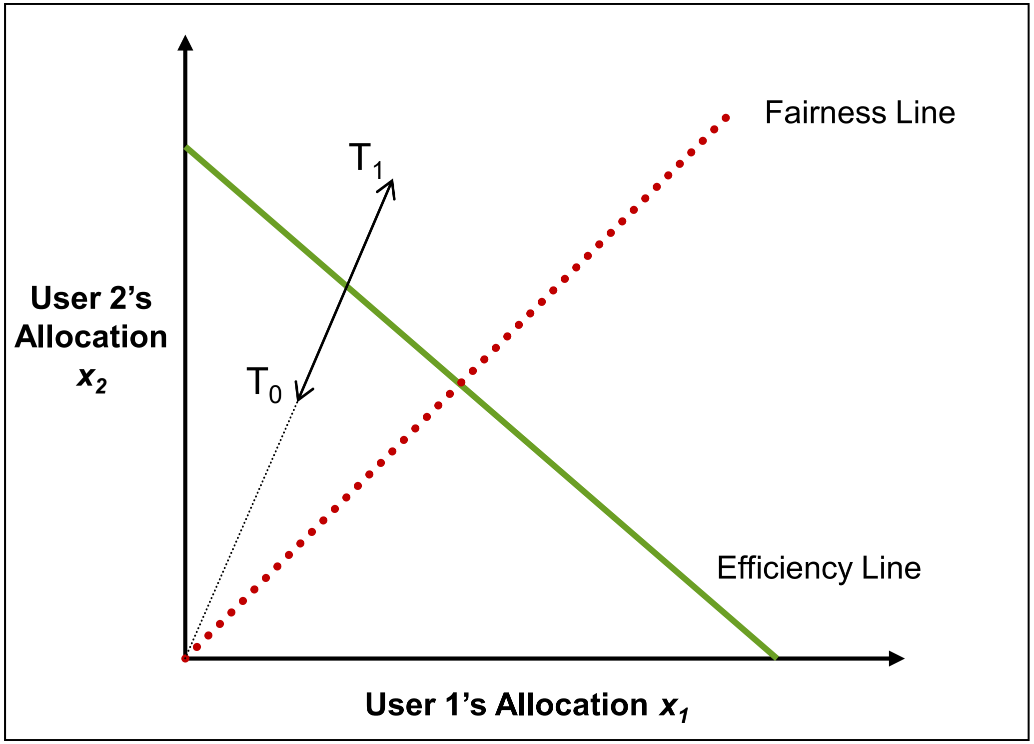
# Additive Increase/Decrease

- Both $X_1$ and $X_2$ increase/decrease by the same amount over time
  - The additive increase policy of increasing both users' allocations by $a_I$ corresponds to moving along a $45^0$ line



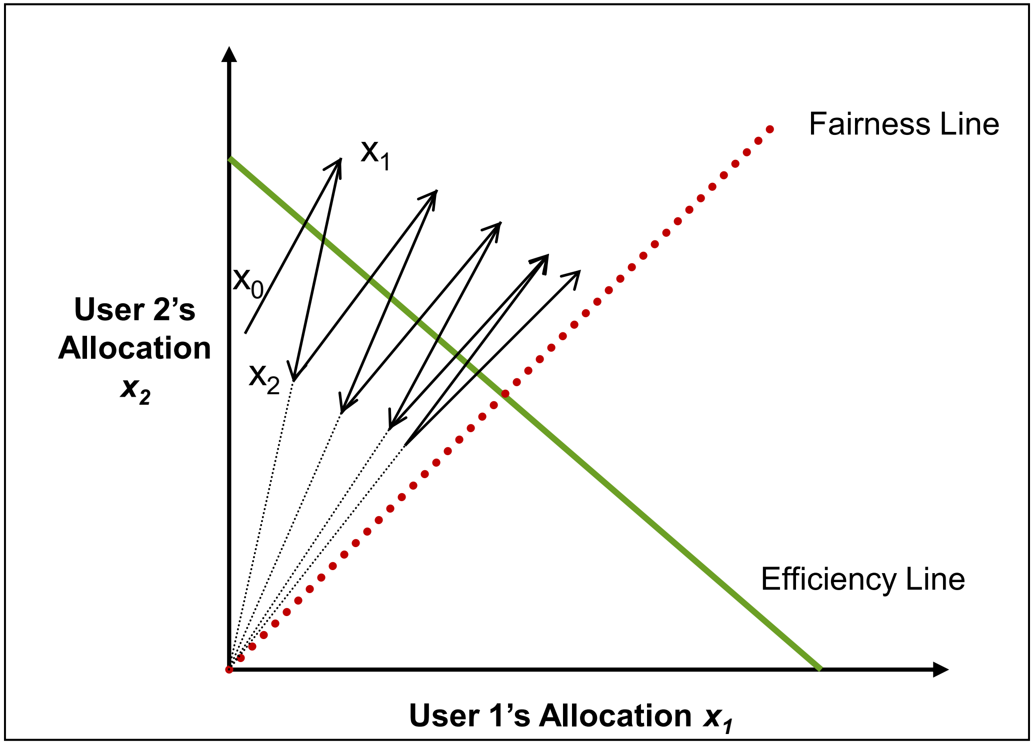
# Multiplicative Increase/Decrease

- Both $X_1$ and $X_2$ increase/decrease by the same factor over time
  – Extension from origin – constant fairness



# What is the Right Choice?

- Constraints limit us to AIMD
  - AIMD moves towards optimal point



# Practical Considerations

- Algorithm should be hardware/software independent
  - Scaling parameters are not easily gathered in an automatic fashion in a complex system
- Control should be chosen for the widest range of values of system parameters
  - Linear parameters are chosen over non-linear parameters due to dependency of non linear parameters on system

# Further Issues

- How does delayed feedback affect the control?
  - Delayed feedback becomes less useful!
- Binary feedback limitations?
  - Additional feedback may cut down the oscillations!
- Is considering current number of users worthwhile?
  - Users come and go dynamically!
  - If number of users are bounded then it is easy. How?
- Impact of asynchronous operation?
  - Current assumption is that everything is synchronized!

# Question…?

- Does TCP Converge to Optimal Point…?

# Problems with TCP (Standard)

- Flow Startup
- Misbehaving Senders and Receivers
- Corruption Loss
- Packet Size
  - TCP doesn’t take packet size into account when responding to losses

# TCP Performance

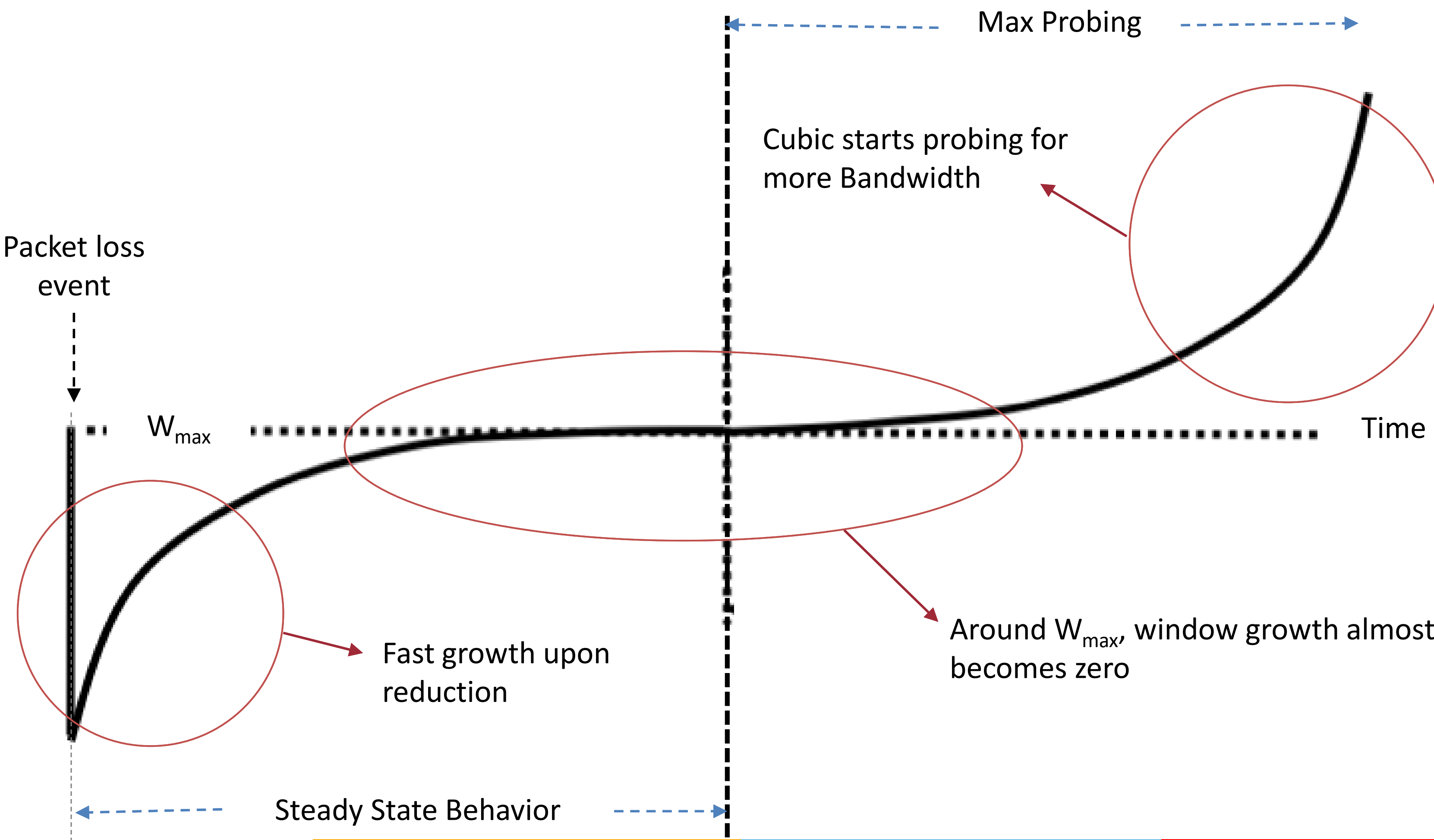
- Question: Can TCP saturate a link?
- Congestion control
  - Increase utilization until… link becomes congested
  - React by decreasing window by 50%
  - Window is proportional to → BW (Rate) * RTT
- Doesn't this mean that the network oscillates between 50 and 100% utilization?
  - Average utilization = 75%??
  - No…this is *not* right!

# Cubic TCP

- Congestion window is cubic function of time

- $cwnd = C(t - K)^3 + W_{max}$ $$K = \sqrt[3]{W_{max}\beta / C}$$
  - $W_{max}$ = cwnd before last reduction
  - β multiplicative decrease factor
  - C scaling factor
  - t is the time elapsed since last window reduction

# Congestion Window: Cubic TCP



# TCP Cubic Advantages

- Good RTT fairness
  - Growth dominated by time t (last congestion event), competing flows have same value of time t, after synchronized packet loss
- Real-time dependent
  - Does not depend on ACK's like TCP Reno
- Scalability
  - Cubic increases window to $W_{max}$ (or its vicinity) quickly and keeps it there longer
- Drawback - Slow Convergence
  - Flows with higher cwnd are more aggressive initially
  - Prolonged unfairness between flows

# Thank You!