A Control Theoretical Approach to a Window-based Flow Control Mechanism with Explicit Congestion Notification

Hiroyuki Ohsaki

Graduate School of Engineering Science
Osaka University, Japan
oosaki@ics.es.osaka-u.ac.jp

Contents

- Introduction
- Analysis of window-based flow control mechanism based on TCP Vegas
- Simulation results
- ECN (Explicit Congestion Notification) in TCP Vegas
- Conclusion

TCP (Transmission Control Protocol)

- **Packet retransmission mechanism**
  - Retransmit lost packets in the network
- **Congestion avoidance mechanism**
  - A window-based flow control mechanism
- Several versions of TCP
  - TCP Tahoe
  - TCP Reno
  - TCP Vegas

TCP Reno

- Implemented in BSD UNIX
- Widely used in the current Internet
- Use **packet loss** as feedback information
  1. Source host continuously **increases** window size
  2. Packet loss occurs at the bottleneck router
  3. Source host detects packet loss by duplicate ACK
  4. Source host **reduces** its window size to 1/2
- Packet loss is **inevitable**

TCP Vegas

- Advantages over TCP Reno
  - A new retransmission mechanism
  - An improved **congestion avoidance mechanism**
  - A modified slow-start mechanism
- Uses **measured RTT** as feedback information
  1. Source host measures RTT for a specific packet
  2. Source host estimates **severity of congestion**
  3. Source host changes window size
- Packet loss can be **prevented**

Congestion avoidance of TCP Vegas

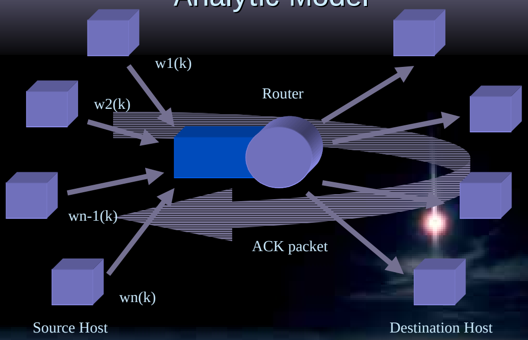
- Source host maintains the minimum RTT: τ
- Source host measures the actual RTT: *r(k)*

$$d(k) = \frac{w_n(k)}{\tau} - \frac{w_n(k)}{r(k)}$$

- Window size is changed based on *d(k)*

$$w_n(k+1) = \begin{cases} w_n(k)+1 & \text{if } d(k) < \alpha \\ w_n(k)-1 & \text{if } \beta < d(k) \\ w_n(k) & \text{otherwise} \end{cases}$$

Analytic Model



System State Equations

- Window size: $w_n(k)$
  - δ: **a control parameter** that determines the amount of increase/decrease in window size

$$w_n(k+1) = \max\left(w_n(k) + \delta(\gamma - d(k)), 0\right)$$

- The number of packets in the buffer: $q(k)$

$$q(k+1) = \min\left(\max\left(N\,w(k) - B w(k), 0\right), L\right)$$

Stability Analysis

1. Derive **the fixed point**: (w^*, q^*)
2. **Linearize** the system around the fixed point
3. Obtain conditions for the system to be **locally exponentially stable**

$$\delta > 0$$

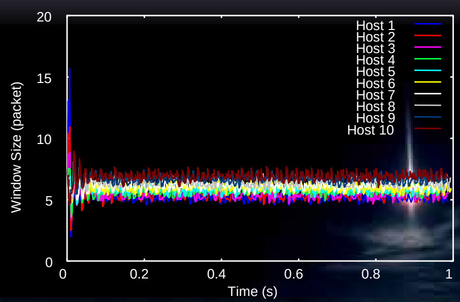
$$\frac{\delta(B - \gamma N)}{(B + \gamma N)\tau} + 2 > 0$$

$$\frac{B\delta}{(B + \gamma N)} < 1$$

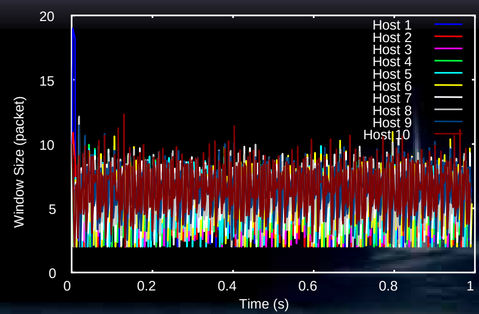
Simulation Parameters

- Packet length 1000 byte
- Number of connections 10
- Bottleneck bandwidth 20 packet/ms
- Propagation delay 1 ms
- Control parameter γ 3 packet
  - Determine buffer occupancy per connection
- Control parameter δ 0.4, 2.0, 3.0
  - 0.4 (optimal), 2.0 (stable) , 3.0 (unstable)

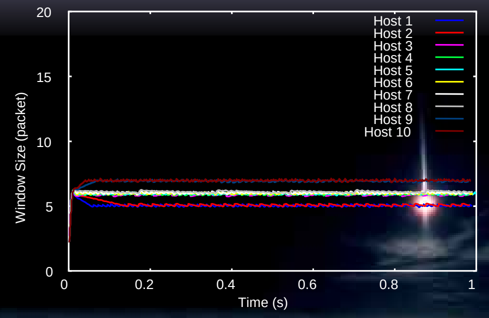
Stable Behavior (δ=2.0)



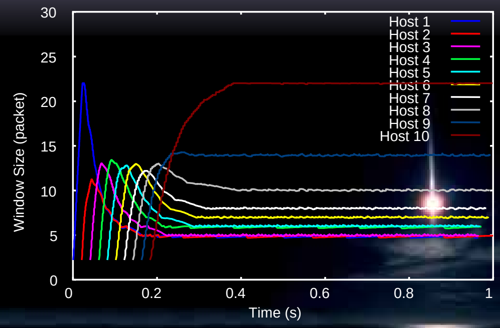
Unstable Behavior (δ=3.0)



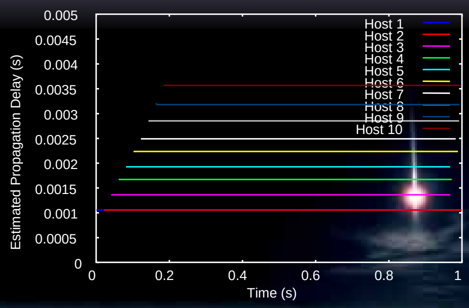
Optimal Behavior (δ=0.4)



Staggered Activation (δ=0.4)



Estimated Propagation Delay



Drawbacks of TCP Vegas

- **Incorrect measurement** of propagation delay
  - All routers have separate buffers for all connections
  - Offered traffic load is very low
  - Throughput becomes **proportional to the measured propagation delay**
- **No scalability** for the number of connections
  - Buffer occupancy is proportional to the number of connections

$$w^* = \tau\left(\frac{B+\gamma N}{N}\right) \qquad q^* = \gamma N \tau$$

ECN in TCP Vegas

- **ECN (Explicit Congestion Notification)**
  - Explicitly notify source hosts of congestion ocurrence
- Two types of implementation
  - ICMP Source Quench packet
  - TOS (Type of Service) field in IP packet header
- When ECN message is received...
  - TCP Vegas seems not to operate correctly
  - Window size **should be decreased**

Proposed Scheme

- Use TOS filed of the IP packet
- Calculate the **ratio of ECN messages**

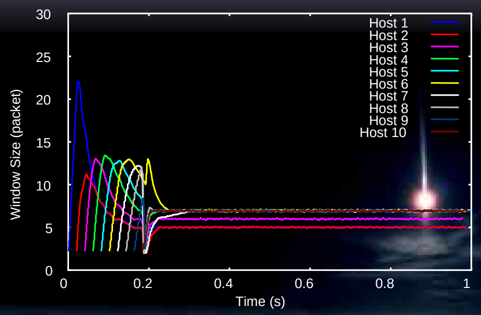
$$e(k) = \frac{N_e}{N_a}$$

- Control window size to **e(k) → 0**

$$w_n(k+1) = \max\left(w_n(k) + \delta(\gamma - d(k)) - w_n(k)e(k), 0\right)$$

- Different from discussion in IETF

Case of Proposed Scheme (δ=0.4)



Conclusion

- Window-based flow control mechanism based on TCP Vegas
- Stability analysis
- Simulation results
- Illustrate drawbacks of TCP Vegas
  - Incorrect measurement of the propagation delay
  - No scalability for the number of connections
- Solution using ECN mechanism
  - Fairness among connections are greatly improved