CUBIC: A New TCP-Friendly High-Speed TCP Variant

Sangtae Ha, Injong Rhee and Lisong Xu

Presented by

Shams Feyzabadi

Introduction

- As Internet evolves the number of Long distance and High speed networks grows
- Bandwidth and Delay Product (BDP): The total number of packets in flight OR the congestion window size must fully utilize the bandwidth
- Standard TCPs use Additive Increase Multiplicative Decrease (AIMD) and increase their congestion window size slowly
- Example: Bandwidth 10Gbps, RTT 100ms, Packet size 1250 bytes, takes 10,000 seconds to fully utilize

Binary Increase Congestion(BIC)

- Features:
  - Increase the window size using binary search
  - Very stable
  - Highly scalable
  - Slowest window increase at saturation point
  - Fair with other TCP flows

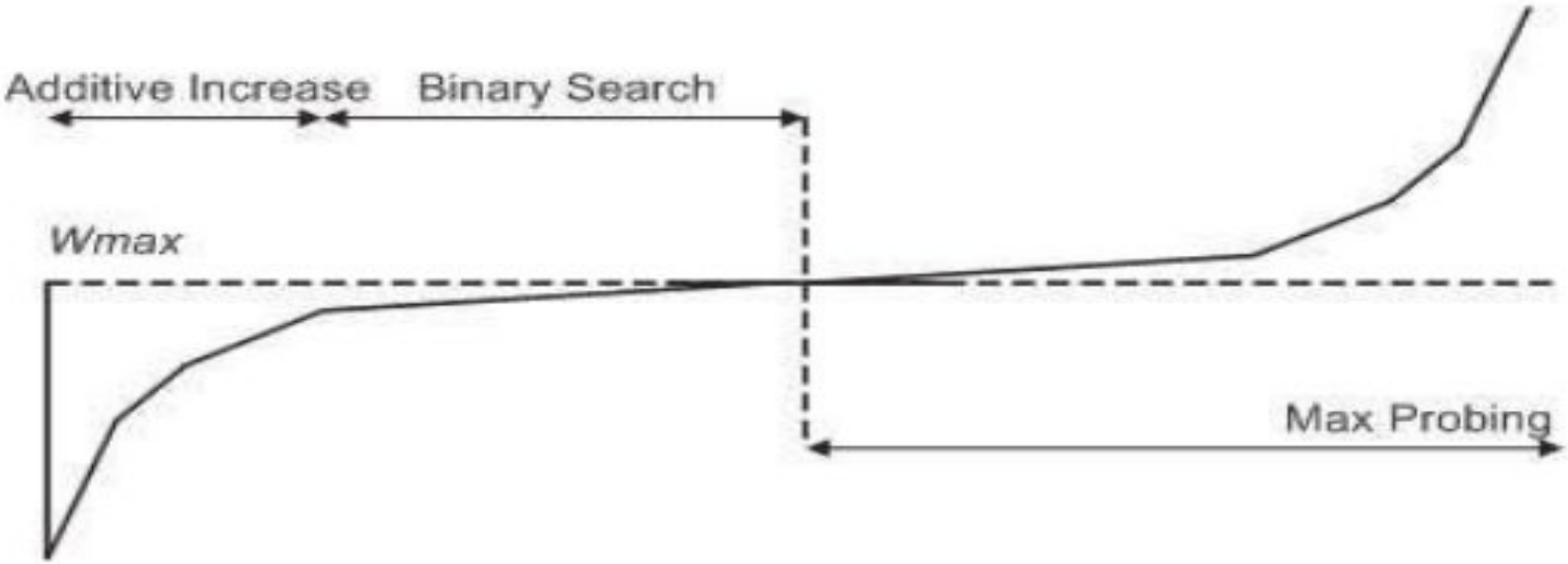
BIC Algorithm

- After a packet loss, reduces its window by a multiplicative factor of β, Default is 0.2
- The window size just before reduction is set to W_{max} and after reduction W_{min}
- The next step, it finds the midpoint using these two sizes and jump there
- If midpoint is very far from W_{min}, a constant value called S_{max} is used.
- If no loss, W_{min} is set to the new window size

BIC Algorithm(Cont.)

- The porcess continues until, the increment is less than a constant value of S_{min}
- Then it is set to the maximum window
- If no loss, new maximum must be found and enters ”max probing” phase
- Window growth function is exactly symetric to the previous part

BIC window size



(a) BIC-TCP window growth function.

BIC Problems

- BIC works very nice, but in low speed or short RTT networks is too aggressive for TCP
- Different phases like binary search increase, max probing, S_{max} and S_{min}, make its implementation very hard
- Another congestion control is required to solve these problems, while having its advantages esp stability and scalability

CUBIC

- The default congestion control on linux machines
- A decendent of BIC congestion control
- As the name represents, it uses cubic function for window growth
- It uses time instead of RTT to increase the window size

CUBIC Algorithm

- After a packet loss, reduces its window by a multiplicative factor of β by default of 0.2
- The window size just before reduction is set to W_{max}
- After it enters into congestion avoidance, it starts to increase the window using a cubic function
- The plateau of cubic function is set to W_{max}
- Size of the window grows in concave mode to reach W_{max}, then it enters the convex part

CUBIC Algorithm(Cont.)

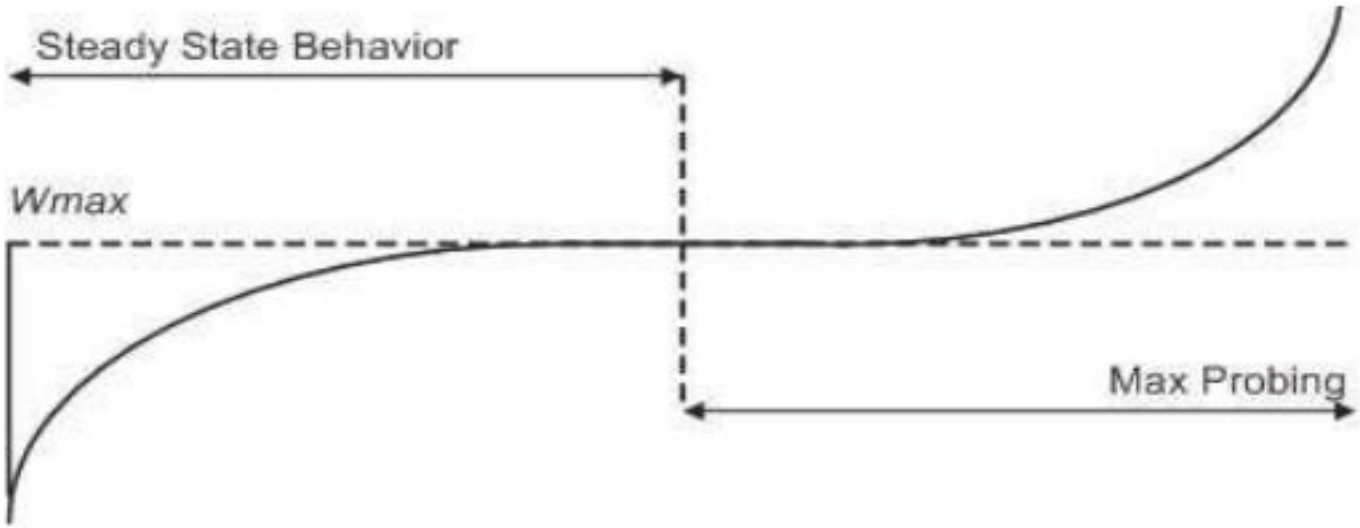
- The window growth function uses the formula below

$$W(t) = C(t-K)^3 + W_{\max}$$

$$K = \sqrt[3]{\frac{W_{\max}\beta}{C}}$$

- Where C is cubic constant, t is elapsed time from the last window reduction and K is the time period takes to get from W to W_{max} while no other loss occurs

CUBIC Window Size Figure



(b) CUBIC window growth function.

How It works

- Upon receiving ACK during congestion avoidance, it computes W(t+RTT) as congestion window
- If it is less than what standard TCP can reach then CUBIC is in TCP mode
- If it is less than W_{max} and more than W_{tcp} then CUBIC is in concave mode
- Otherwise CUBIC is in convex mode

TCP Friendly Region

- How do we find out we are in TCP mode at time t?
  - The average window size of AIMD, with additive factor of α, multiplicative factor of β and average loss rate of p will be

$$\frac{1}{RTT}\sqrt{\frac{\alpha}{2}\frac{2-\beta}{\beta}\frac{1}{p}}$$

$$W_{tcp}(t) = W_{\max}(1-\beta) + 3\frac{\beta}{2-\beta}\frac{t}{RTT}$$

  - If the CUBIC window size is less than this Wtcp it is set to Wtcp

Other regions

- In concave mode the cwnd is incremented by the following factor

$$\frac{W(t+RTT)-cwnd}{cwnd}$$

- In convex mode the formula is the same

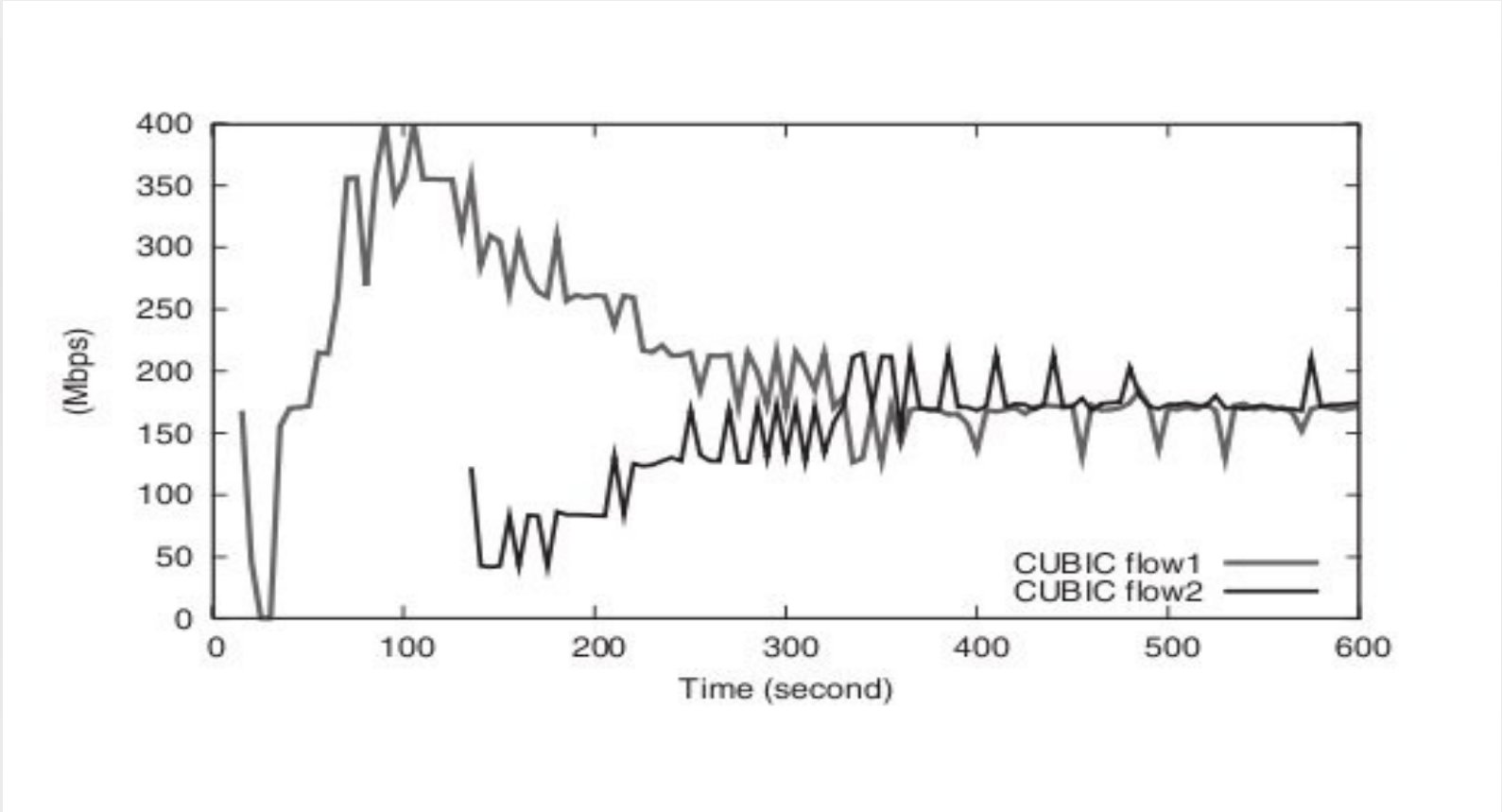
Fast Convergence

- When a new CUBIC flow starts trasmitting packets, the first flow must reduce its speed
- A heuristic is added to this protocol called ”Fast Convergence”
- When a loss occurs, before setting W_{max} CUBIC stores the previous W_{max} in W_{last_max}
- Now if $W_{max} < W_{last\_max}$, a new flow has started
- So CUBIC decreases W_{max} further

CUBIC in practice

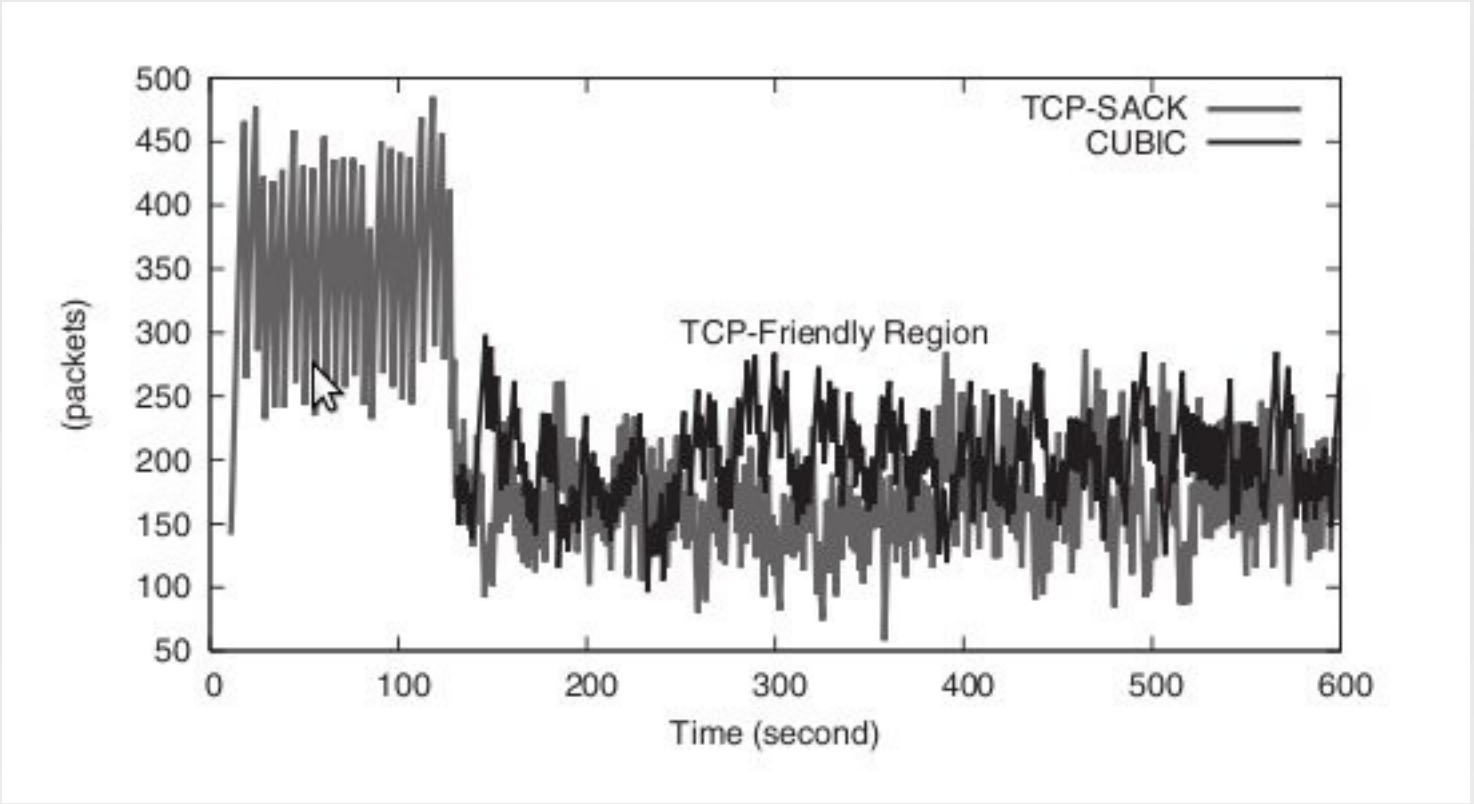
- Standard TCP works well in the following situations
  - Networks with small BDPs
  - Networks with short RTT, but not neccesarily a small BDP
- CUBIC is designed to work similarly in these conditions
- It acts fairly with other TCP flows

Throughput of two CUBIC flows

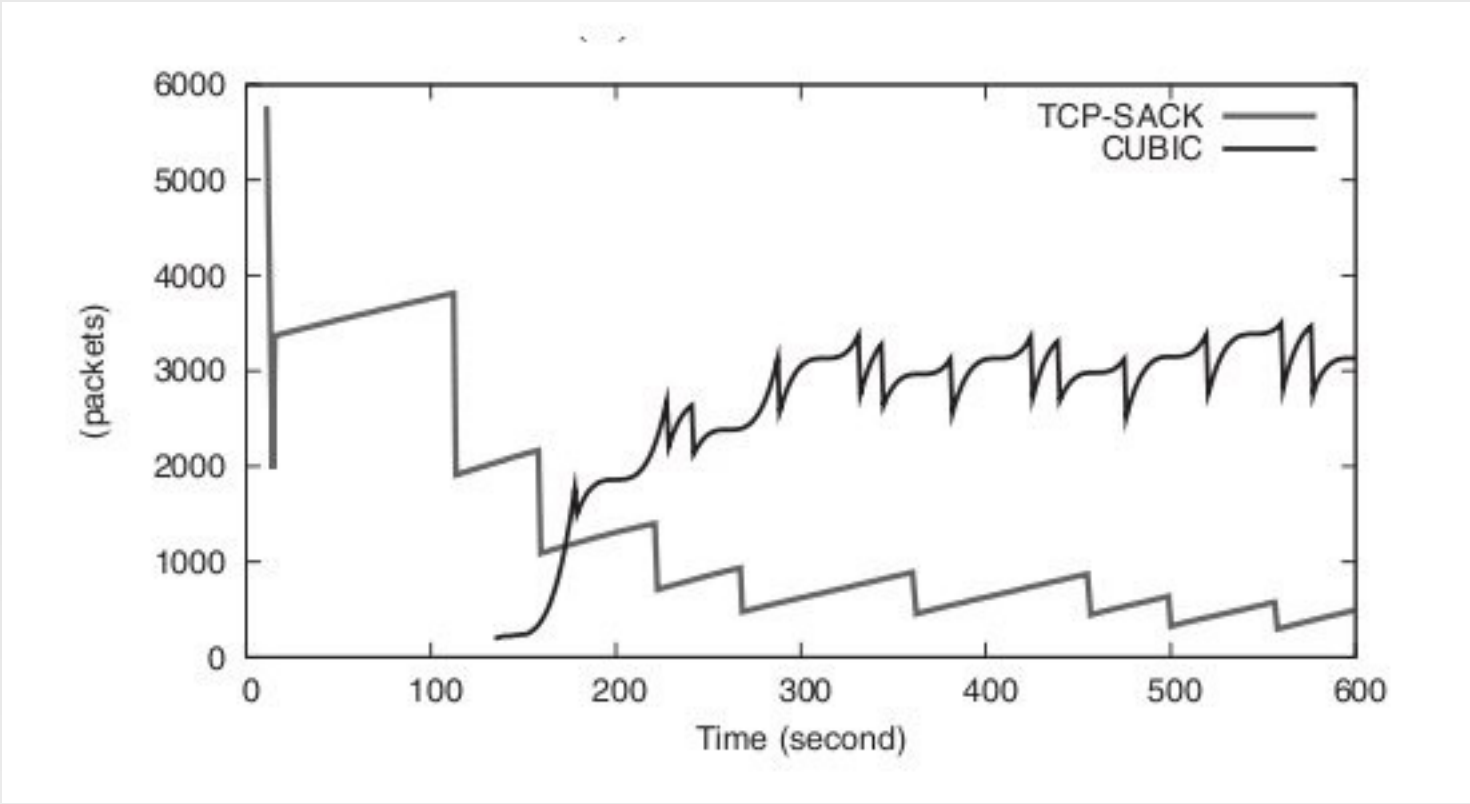


conditions:A dumbbell network configuration with a significant traffic, in both directions. Bottleneck capacity of 400Mbps. RTT 240ms. Drop tail routers

Cwnd in TCP mode(RTT=8ms)



Cwnd in CUBIC mode(RTT=82ms)



Thank you for you attention

References

- CUBIC: A New TCP-Friendly High-Speed TCP Variant, Sangtae Ha, Injong Rhee and Lisong Xu, Proc. 3rd International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet 2005)
- http://tools.ietf.org/id/draft-rhee-tcpm-cubic-02.txt
- www.wikipedia.org