Investigating AIMD Variants and TCP Algorithms for Future Data Centers

Rathee Apurv Singh
College of Data Science and Computing (CCDS)
Nanyang Technological University (NTU)
Singapore
apurv003@e.ntu.edu.sg

1. INTRODUCTION: CHALLENGES OF AIMD IN MODERN DATA CENTERS

1.1 Problem Statement

Data centers serve as the backbone of cloud computing, AI-driven workloads, and hyperscale applications. Unlike traditional wide-area networks (WANs), data center networks (DCNs) operate under unique constraints, such as high-speed (10-100 Gbps) links, ultra-low latency (microsecond RTTs), and bursty workloads. These factors challenge traditional TCP congestion control algorithms, particularly the Additive Increase Multiplicative Decrease (AIMD) mechanism, which was originally designed for low-bandwidth, high-latency internet connections.

The primary weaknesses of AIMD-based TCP in data centers include:

a. Packet Loss Sensitivity: AIMD interprets all packet loss as congestion, leading to unnecessary congestion window (CWND) reductions in lossy environments. This is problematic in data centers, where packet loss can occur due to transient queue buildups rather than persistent congestion.
b. Unfair Bandwidth Allocation: Traditional AIMD favors long-lived flows over short-lived, latency - sensitive applications. AI inference, real-time financial transactions, and cloud microservices suffer from bandwidth starvation, leading to performance degradation.
c. Slow Recovery from Congestion: After a packet loss, AIMD halves CWND, causing underutilization of high-bandwidth links. This slow growth phase is inefficient for 100 Gbps data center links.
d. Inefficiency in Multi-Tenant Cloud Networks: Data centers hot multi-tenant workloads, meaning TCP must adapt dynamically to changing traffic patterns. AIMD does not optimize bandwidth sharing across different cloud tenants, leading to congestion hotspots.
e. Lack of ECMP Awareness: Equal-Cost Multi-Path (ECMP) routing is widely used in data centers, but AIMD-based congestion control does not consider multipath dynamics. This results in poor load balancing and suboptimal link utilization.

1.2 Why Traditional AIMD-Based TCP Fails in Data Centers

Traditional TCP congestion control mechanisms were designed for long-haul WAN environments where:

a. Bandwidth is limited (10-100 Mbps vs. 100 Gbps in DCNs).
b. Congestion is persistent rather than transient.
c. RTTs are high (10-100 ms vs. microseconds in data centers).

1.3 Need for AIMD Modifications in Futuristic Data Centers

Given these limitations, data centers require adaptive, scalable, and AI-driven TCP algorithms that:

a. Dynamically adjust AIMD parameters (α, β) based on real-time congestion feedback,
b. Incorporate Explicit Congestion Notification (ECN) to react before packet loss occurs.
c. Leverage AI and machine learning for predictive congestion control.
d. Optimize TCP performance for multi-tenant, high-speed data centers.

This report explores how AIMD modifications, AI-driven congestion control, and new TCP variants enhance fairness, efficiency, and scalability in futuristic data centers.

2. LITERATURE REVIEW: INNOVATIONS BEYOND AIMD

2.1 Data Center TCP (DCTCP): ECN-Based Congestion Control

AIMD's reliance on packet loss for congestion detection results in:

a. High Buffer Occupancy, which increases queueing delays and packet loss probabilities.
b. Underutilization of bandwidth, as large-scale application experience frequent congestion window reductions.
c. Unfairness in multi-tenant environments, where long-lived flows dominate bandwidth allocation.

DCTCP addresses these issues by introducing Explicit Congestion Notification (ECN) to detect and react to congestion before packet loss occurs. DCTCP does the following:

a. Uses ECN to detect early congestion: Instead of waiting for packet drops, ECN-marked packets signal impending congestion, allowing a more graceful reduction in CWND.
b. Gradual Congestion Window Reduction: Instead of AIMD's multiplicative halving ($\beta = 0.5$), DCTCP reduces the window size proportionally to the fraction of ECN-marked packets.
c. Maintains Low Buffer Occupancy: By keeping queue sizes small, DCTCP lowers queueing delays, reducing latency-sensitive workload disruptions.

However, there are a few disadvantages as well:

a. Unfairness when competing with Traditional TCP: Because DCTCP reduces CWND more gradually than AIMD-based TCP, it can be starved by traditional TCP flows in mixed environments.
b. Requires ECN-Enabled Network Switches: Legacy hardware without ECN support cannot implement DCTCP effectively.

2.2 Adaptive Acceleration Data Center TCP (A2DTCP): Dynamic AIMD Modifications

A2DTCP extends DCTCP by introducing dynamic AIMD adjustments based on real-time congestion feedback. Unlike DCTCP, which only modifies congestion window reduction, A2DTCP adjusts both α (additive increase) and β (multiplicative decrease) dynamically. AIMD's fixed values of $\alpha = 1$ and $\beta = 0.5$ are not optimized for modern high-speed, bursty workloads, leading to slow responsiveness to congestion, inefficient bandwidth utilization, and increased packet retransmissions.

A2DTCP fixes it by:

a. Dynamically adjusting α and β: It modifies increased and decrease parameters based on real-time network conditions.
b. Reducing Unnecessary Packet Loss: Instead of aggressively cutting CWND in half, A2DTCP gradually reduces it, minimizing packet retransmission.

c. Ensuring Higher Throughput Under Congestion: By optimizing AIMD parameters dynamically, A2DTCP prevents excessive bandwidth drops during congestion events.

However, there are a few disadvantages of A2DTCP as well:

a. Increased Computational Overhead: Since A2DTCP continuously adjusts AIMD parameters, it requires more real-time processing DCTCP.
b. Fairness Issues in Multi-Tenant Networks: Some applications may get disproportionate bandwidth allocations if tuning is not balanced correctly.

2.3 Scalable TCP: AIMD Modification for High-Speed Networks

Scalable TCP replaces AIMD by replacing linear congestion window increase with a multiplicative increase. This allows it to scale better in high-speed, higher bandwidth environments. Following are its main features:

a. Uses Multiplicative Increase instead of Additive: This allows faster congestion window growth in high-bandwidth environments.
b. More efficient for Long-Lived Flows: Scalable TCP achieves higher throughput for persistent connections, improving performance for large data transfers.
c. Maintains High Link Utilization: Instead of AIMD's slow recovery process, Scalable TCP allows faster bandwidth recovery after congestion events.

However, there are a few limitations to Scalable TCP:

a. Unfair to Short-Lived Flows: Long-running flows receive more bandwidth, leading to bandwidth starvation for latency-sensitive applications.
b. More Aggressive Under Congestion: Scalable TCP can lead to queue buildup and increased delays under heavy traffic loads.

3. EXPERIMENT

Objective: The goal is to evaluate the performance and fairness of different AIMD congestion control mechanisms in a network environment where multiple TCP flows share a bottleneck. The three AIMD variants considered are:

a. Standard AIMD: the traditional congestion control mechanism used in TCP.
b. Power Function AIMD: a modified AIMD that adjusts its increase and decrease behavior using a power function.
c. Logarithmic AIMD: another variant that adapts its congestion window growth logarithmically.

Methodology: A simulated network environment was set up where three TCP flows traverse a common bottleneck. The congestion control mechanisms for each flow were implemented separately, and the network dynamics, including congestion window evaluation, throughput, queue occupancy, and fairness, were recorded over 100 Round-Trip Times (RTTs). To reflect real-world network variations, random noise was added to congestion window updates and queue occupancy to simulate unpredictable network conditions.

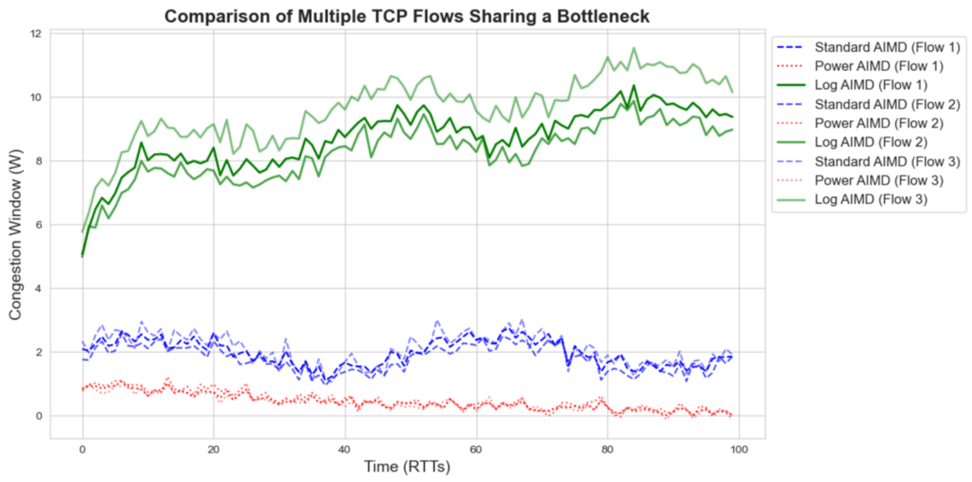
Metrics Measured:

a. Congestion Window Evolution: to observe how each AIMD variant adjusts its window size over time.

b. Throughout Evolution: to analyze the rate at which the data is successfully transmitted.
c. Queue Occupancy: to measure how the buffer size fluctuates for each flow.
d. Jain's Fairness Index: to evaluate the fairness of bandwidth allocation among competing flows.

4. RESULTS

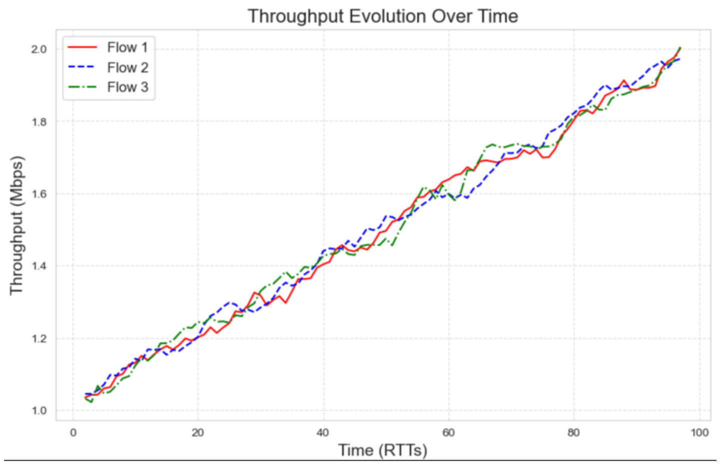
4.1 Congestion Window



a. Logarithmic AIMD achieves a much larger congestion window over time.
b. Standard AIMD stabilizes at a lower congestion window, fluctuating within a smaller range.
c. Power AIMD exhibits the slowest increase, reflecting its conservative window adjustment strategy.

These results indicate that Logarithmic AIMD is more aggressive in acquiring bandwidth, whereas Power AIMD is more cautious, leading to reduced congestion but slower transmission.

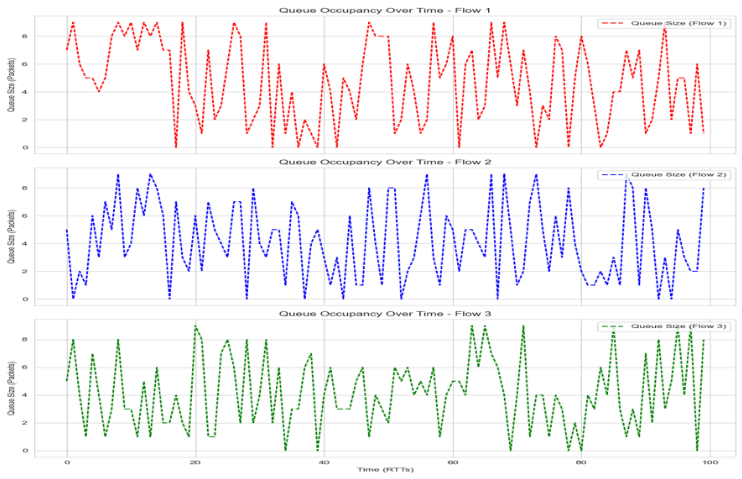
4.2 Throughput Evolution



The graph highlights:
a. A steady increase in throughput for all three flows.
b. Minimal fluctuations, suggesting that all three AIMD variants adapt efficiently to network conditions.
c. Similar overall throughput values across flows, indicating a fair distribution of bandwidth.

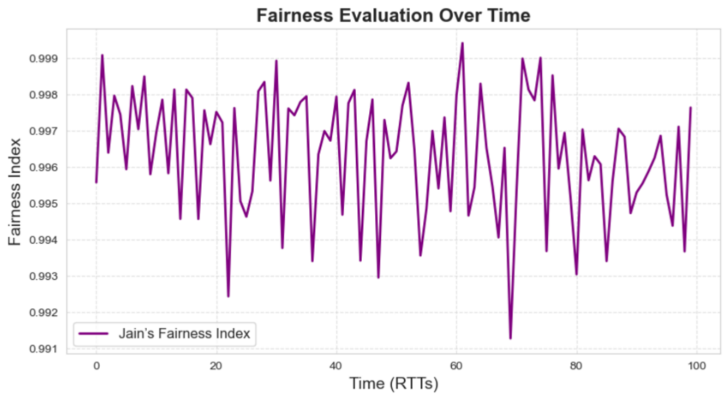
This indicates that despite differences in congestion window evolution, throughput remains balanced across the three flows.

4.3 Queue Occupancy



a. The queue occupancy graphs exhibit high variability, reflecting transient congestion periods.
b. Flow 1 (Red) has more fluctuations, which might indicate more aggressive congestion control behavior.
c. Flow 2 (Blue) and Flow 3 (Green) exhibit comparable oscillations.

4.4 Jain's Fairness Index



Jain's Fairness Index remains very close to 1, fluctuating between 0.994 and 0.999, suggesting that bandwidth allocation remains highly fair.

a. Despite the different AIMD mechanisms, no flow completely dominates the bandwidth.
b. The minor fluctuations are due to random noise and transient congestion effects.

4.5 Conclusion

This experiment demonstrated the impact of different AIMD congestion control variants on network performance, highlighting the trade-offs between aggressiveness and stability. Logarithmic AIMD achieved the highest congestion window growth, while Power AIMD exhibited a more conservative approach, leading to lower queue occupancy. Despite these differences, Jain's Fairness Index remained close to 1, confirming equitable bandwidth distribution across flows.

REFERENCES

1. Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., & Sridharan, M. (2010). Data center TCP (DCTCP). *Data Center TCP (DCTCP)*. https://doi.org/10.1145/1851182.1851192

2. Joseph, V., & De Veciana, G. (2011). Stochastic networks with multipath flow control. *Stochastic Networks With Multipath Flow Control: Impact of Resource Pools on Flow-level Performance and Network Congestion*, 61–72. https://doi.org/10.1145/1993744.1993752

3. Ha, S., Rhee, I., & Xu, L. (2008). CUBIC. *ACM SIGOPS Operating Systems Review*, *42*(5), 64–74. https://doi.org/10.1145/1400097.1400105

4. John Ousterhout. (2023). It’s time to replace TCP in the datacenter. *Stanford University*.