Bangladesh University of Business & Technology (BUBT)

Assignment for Mid-term Examination

Course Code **:** CSE 319

Course Title **:** Computer Network

Assignment no : 01

Submitted By

Name : Abdullah Al Hill Baki
ID : 21225103341
Intake : 49
Section : 08
Program :B.Sc. in CSE

Submitted To

Dr. Khandoker Nadim Parvez
Associate Professor

Department of CSE
BUBT

Date of Submission: 2nd November ,2024

…………………………………..
Signature of Teacher

TCP congestion control is designed to allow each source to assess the available network capacity, helping it decide how many packets it can safely send at once. In wireless TCP congestion control, this concept is adapted to handle data flow in TCP connections over wireless networks, where connections face unique challenges like signal interference, variable latency, and increased packet loss. Traditional TCP congestion control interprets packet loss as a signal of network congestion, triggering a slowdown in data transmission to prevent network overload. However, in wireless networks, packet loss often occurs due to issues like signal fading or interference rather than congestion.

Wireless TCP congestion control methods modify traditional TCP algorithms to account for the distinctive characteristics of wireless environments. In these settings, packet loss may stem from factors such as signal fading, interference, and mobility, which do not necessarily indicate congestion. Some prominent techniques in wireless TCP congestion control include:

TCP Westwood+ :

TCP Westwood+ refines traditional TCP by estimating available bandwidth based on the rate at which acknowledgment (ACK) packets are received. Upon detecting packet loss, Westwood+ calculates available bandwidth and adjusts the congestion window to sustain throughput, instead of drastically reducing it.

Advantages:

- Better Bandwidth Utilization: Westwood+ can effectively utilize the network by accurately estimating available bandwidth, leading to improved data transfer rates over unreliable wireless connections.

- Reduced Sensitivity to Packet Loss: By distinguishing between packet loss due to wireless errors and actual congestion, Westwood+ avoids unnecessary reductions in the congestion window, enhancing throughput.

- Energy Efficiency: Westwood+ conserves power in mobile devices by preventing needless retransmissions, which is especially helpful in battery-limited environments.

Disadvantages:

- Bandwidth Estimation Errors: In rapidly changing environments, bandwidth estimation can be inaccurate, resulting in suboptimal congestion window adjustments.

- Increased Processing Overhead: Continuous monitoring and estimating bandwidth add computational load, potentially increasing both energy usage and processing demands.

- Complexity in Implementation: The requirement for dynamic estimation and adjustment of the congestion window makes Westwood+ more complex to implement than simpler congestion control methods.

TCP Westwood+ is specifically designed to handle the unique characteristics of wireless networks. It estimates available bandwidth after packet loss, adjusting the congestion window as needed instead of simply halving it, as in TCP Reno. This approach makes it well-suited for networks with high packet loss and fluctuating bandwidth, such as mobile and satellite connections. While well-adapted to wireless environments, Westwood+ may not fully utilize bandwidth in extremely dynamic or congested conditions, but it is highly effective in preventing bandwidth underutilization due to packet loss unrelated to congestion.

Selective Acknowledgment (SACK) TCP:

SACK TCP enables the receiver to provide detailed feedback on which packets have been successfully received, even if they arrive out of order. By acknowledging specific, non-continuous packets, SACK minimizes redundant retransmissions, allowing the sender to resend only the missing segments.

Advantages:

- Efficient Loss Recovery: SACK allows the sender to retransmit only the lost segments, reducing unnecessary retransmissions and conserving bandwidth.

- Increased Throughput: By retransmitting only the missing segments, SACK maintains higher data flow and speeds up recovery from packet loss.

- Improved Compatibility: SACK can work alongside various TCP extensions, making it adaptable to different TCP implementations and protocols.

Disadvantages:

- Increased ACK Overhead: SACK introduces additional complexity in ACK packets, as more information is needed to specify received and missing segments.

- Requires SACK Support on Both Ends: For SACK to work effectively, both sender and receiver must support the SACK option, which may not always be possible.

- Complexity in Implementation: Implementing SACK involves modifying the acknowledgment process, increasing protocol complexity.

SACK allows the receiver to notify the sender about specific lost packets, enabling retransmission of only the missing data. This approach reduces retransmission of successfully delivered packets, enhancing efficiency over noisy wireless links. SACK is particularly useful in networks with frequent packet loss or reordering, such as high-interference environments. However, SACK can struggle in heavily congested or burst wireless networks, where multiple packet losses often occur.

Split-TCP (Indirect-TCP):

In Split-TCP, the TCP connection is divided into two distinct segments at a base station or proxy. This proxy splits the connection into a "wired" and a "wireless" segment, each with independent TCP congestion control.

Advantages:

- Improved Performance on Wireless Links: By isolating the wireless segment, Split-TCP ensures that issues on the wireless link do not affect the entire connection.

- Enhanced Error Handling: Packet retransmissions can be managed locally by the base station or proxy, reducing latency and speeding up recovery from packet loss.

- Flexibility for Network Conditions: Each segment can use optimized TCP settings, allowing better adaptation to diverse conditions on the wired and wireless segments.

Disadvantages:

- Security and Privacy Concerns: Dividing the connection at a proxy can compromise end-to-end encryption, leading to potential privacy issues.

- Dependency on Intermediary Node: This method depends on the base station or proxy, which can be a single point of failure and may create a bottleneck.

- Additional Latency: Managing traffic for both segments may cause delays, especially if the base station is overloaded.

Split-TCP divides the connection into wired and wireless segments, optimizing each part separately. The wireless segment manages losses locally, while the wired segment remains unaffected by wireless issues. It is particularly effective in environments with a combination of wired and wireless segments, such as Wi-Fi or satellite connected to wired infrastructure.

Snooping TCP (Snoop Protocol):

The Snoop Protocol operates at the base station, monitoring TCP packets as they pass between sender and receiver. It caches unacknowledged packets, and if it detects a lost packet (indicated by duplicate ACKs), the base station locally retransmits it on the wireless link, preventing congestion window reduction.

Advantages:

- Localized Packet Recovery: By managing retransmissions at the base station, Snoop conserves bandwidth and improves speed by avoiding unnecessary end-to-end retransmissions.

- Stable Transmission Rate: Prevents unnecessary shrinking of the sender’s congestion window, maintaining a stable data rate even when the wireless link is unreliable.

- Compatibility with Standard TCP: Snoop works with existing TCP implementations without requiring changes at the sender or receiver.

Disadvantages:

- Requires Infrastructure Modification: Implementing Snoop requires changes to base stations, which can be costly and complex.

- Not Suitable for Encrypted Data: Snooping relies on inspecting packet headers, making it incompatible with encrypted traffic.

- Limited Scope: Effective only in networks with fixed infrastructure (e.g., cellular networks) and less suitable for ad hoc or highly mobile wireless environments.

Freeze-TCP:

Freeze-TCP enables mobile devices to instruct the sender to "freeze" the congestion window during anticipated short disconnections (e.g., during cell handoff). The sender pauses data transmission, avoiding unnecessary retransmissions and congestion window reduction.

Advantages:

- Reduced Retransmission Cost: By pausing transmission during short disconnections, Freeze-TCP conserves bandwidth by avoiding unnecessary retransmissions.

- Energy Conservation: Reduces energy consumption by avoiding retransmissions when the receiver is temporarily disconnected, which is beneficial for mobile devices.

- Maintains High Throughput: Prevents unnecessary congestion window reductions, allowing the sender to resume data transfer quickly at full speed once reconnected.

Disadvantages:

- Delay in Data Transmission: Freezing the connection may introduce delays, especially if disconnections are frequent.

- Dependency on Accurate Signal Prediction: Requires reliable detection of potential disconnections, which is challenging in environments with variable signal quality.

- Limited Application: Best suited for networks with predictable short disconnections, like cell handoffs, and less effective in unstable networks.

Freeze-TCP is ideal for mobile networks with intermittent connectivity, such as those involving devices or vehicles moving between cells. By pausing the TCP session during expected disconnections, it minimizes throughput loss and avoids unnecessary congestion window reductions.

TCP Reno:

TCP Reno is a traditional congestion control protocol that uses an additive-increase, multiplicative-decrease (AIMD) approach. It gradually increases the congestion window additively until it detects packet loss, which is taken as a sign of congestion. When packet loss occurs, TCP Reno reduces the congestion window by half (multiplicative decrease). This strategy helps control congestion but presents challenges in wireless networks due to higher packet loss rates and varying signal quality.

Advantages:

- Resource-Efficient: Requires low processing and memory overhead, making it lightweight and simple to implement.

- Quick Implementation: Well-established and compatible with legacy systems, allowing easy deployment.

- Stable in Moderate Networks: Performs reliably in networks with moderate congestion or lower bandwidth.

- Adaptive to Short-Lived Connections: Effective for short sessions, such as web browsing or low-data applications.

Disadvantages:

- Sensitivity to Non-Congestion Loss: Treats all packet loss as congestion, leading to unnecessary reductions in throughput in wireless networks where loss may be unrelated to congestion.

- Inconsistent Throughput in High-Loss Networks: Frequent window reductions in lossy environments cause unstable performance.

- Underutilizes High-Bandwidth Links: Has difficulty fully utilizing high-speed wireless links, such as LTE or 5G.

- Inadequate for Bursty Traffic: Slow to adjust to traffic bursts, resulting in poor performance in dynamic wireless settings.

- Limited Fairness in Mixed Environments: Struggles to compete with more aggressive TCP variants like TCP CUBIC in shared environments.

Best Use Case:

TCP Reno works best in low-bandwidth, moderately stable wireless connections with infrequent packet loss. However, due to slow recovery from losses, it is less ideal than more advanced methods for high-loss or high-speed wireless networks.

TCP CUBIC:

TCP CUBIC is a modern congestion control protocol optimized for high-bandwidth, long-distance connections. It uses a cubic growth function instead of linear growth, allowing it to recover bandwidth more quickly after packet loss. TCP CUBIC aggressively increases the congestion window after loss, aiming to regain the previous high window size (Wmax) and probe for additional capacity.

Advantages:

- Higher Throughput in High-Bandwidth Networks: Efficiently utilizes available bandwidth by increasing the congestion window aggressively.

- Reduced Time in Slow Start: CUBIC's cubic growth reduces the duration spent in slow start, reaching high data rates more quickly.

- Better Performance on Long Connections: Maintains stable throughput over extended periods, making it ideal for large data transfers.

- Fairness Across TCP Variants: Coexists effectively with other TCP protocols, enhancing fairness in mixed environments.

Disadvantages:

- Complex Implementation: Requires more advanced algorithms and resources, adding complexity.

- Resource Intensive under High Load: May demand more CPU and memory, impacting performance under heavy load.

- Oscillation in Dynamic Conditions: The cubic function can cause fluctuations in variable network environments.

- Sensitivity to Misconfiguration: Requires careful tuning to avoid suboptimal performance.

- Less Effective in Highly Congested or Lossy Networks: Performance can degrade in networks with frequent congestion or packet loss.

Best Use Case:

TCP CUBIC is well-suited for high-speed wireless networks with large data transfers, such as LTE or 5G networks. Its aggressive window growth efficiently utilizes high-bandwidth wireless connections and enables faster recovery from packet loss.

TCP Vegas:

TCP Vegas adopts a proactive approach to congestion control, using round-trip time (RTT) to gauge network congestion and adjust the sending rate accordingly. It establishes a baseline minimum RTT, and by comparing expected versus actual throughput, TCP Vegas can adjust the congestion window to maintain stability and avoid packet loss.

Advantages:

- Proactive Congestion Avoidance: Adjusts sending rates based on RTT, preventing congestion before packet loss occurs.

- Efficient Bandwidth Utilization: Reduces retransmission overhead by avoiding congestion-related losses, resulting in higher efficiency.

- Lower Latency: Maintains a low-latency environment, which is beneficial for applications requiring timely data transfer.

- Stable Throughput: Provides a smooth, stable throughput by minimizing fluctuations and oscillations.

- Responsiveness for Real-Time Applications: Quickly reacts to network changes, making it ideal for real-time communications.

Disadvantages:

- Dependency on RTT Accuracy: Relies on precise RTT measurements, which can be challenging in dynamic conditions.

- Configuration Complexity: More complex than simpler TCP variants, requiring careful setup and tuning.

- Ineffective in Short-Lived Connections: Performs less efficiently on short connections, as time is needed to establish an accurate baseline.

- Limited Deployment: Not widely adopted, which can lead to compatibility issues across platforms.

- Challenges in Congested or Bursty Networks: Performance may suffer in highly congested or bursty networks.

Best Use Case:

TCP Vegas is suitable for environments where maintaining stable throughput and low latency is essential, such as real-time communication networks with moderate to stable traffic conditions.

TCP BBR (Bottleneck Bandwidth and Round-trip Time):

TCP BBR (Bottleneck Bandwidth and Round-trip Time) is a congestion control mechanism that maximizes throughput by estimating the bottleneck bandwidth and RTT without relying on packet loss as an indicator of congestion. It dynamically adjusts the sending rate based on measurements of available bandwidth and latency, aiming to keep the network pipeline full without causing congestion.

Advantages:

- Maximizes Throughput Efficiently: Optimizes data flow by estimating bottleneck bandwidth, providing higher efficiency.

- Reduced Latency: Avoids congestion-based packet loss, benefiting applications that need rapid response times.

- Adaptable to Network Variability: Quickly adapts to changes in network conditions, maintaining consistent throughput.

- Scalable in High-Speed Environments: Works well in high-speed networks and data centers, achieving significant throughput improvements.

- Stable in Diverse Conditions: Reliable performance across different environments, minimizing the impact of network fluctuations.

Disadvantages:

- Complex Implementation: Requires advanced bandwidth estimation, adding complexity to configuration and resource demands.

- Sensitivity to Bandwidth Overestimation: Overestimating available bandwidth may lead to congestion, which can impact performance.

- Less Predictable in Highly Variable Conditions: May exhibit unpredictable behavior in rapidly changing networks.

- Limited Support on Legacy Systems: Compatibility with older systems can be limited, affecting its applicability across networks.

- Resource-Intensive: Monitoring and adjustments require significant CPU and memory resources.

Best Use Case:

TCP BBR excels in high-speed, stable environments with large data flows, such as data centers or cloud networks. Its ability to dynamically adjust to bottleneck bandwidth and RTT makes it ideal for scenarios requiring high throughput and low latency, especially in environments that can handle its resource demands.

DCTCP (Data Center TCP):

DCTCP is a congestion control protocol designed specifically for data center environments. It employs Explicit Congestion Notification (ECN) to proactively manage congestion by adjusting the sending rate based on real-time feedback from the network, which helps reduce queue lengths and improve overall latency.

Advantages:

- Optimized for data center traffic patterns: Tailored for applications in data centers that require low latency and high throughput.

- Efficient queue management: Minimizes queue length, which leads to reduced delays in data transmission within the data center.

- Proactive congestion management via ECN: Utilizes ECN to signal congestion before packet loss occurs, enhancing network efficiency.

- High throughput for large data transfers: Performs exceptionally well in high-speed environments with substantial data flows, typical of data centers.

- Low packet loss rates: Maintains low packet loss rates, which is crucial for applications sensitive to latency and data loss.

Disadvantages:

- Requires ECN-capable infrastructure: Relies on the availability of ECN for effective congestion signaling, which may not be present in networks outside of data centers.

- Limited to data centers: Primarily designed for use within data center environments, making it less effective in typical internet or wide-area network settings.

- Complex setup and tuning: Requires careful configuration and fine-tuning for optimal performance, especially in heterogeneous data center environments.

- Potential fairness issues in mixed traffic: May struggle with fairness when managing a mix of data center-specific traffic and general internet traffic.

- Resource constraints under high congestion: Performance can degrade significantly if the network experiences sustained high levels of congestion.

Comparison of TCP Westwood+ and TCP CUBIC:

TCP Westwood+:

- Best for general wireless environments characterized by variable bandwidth and frequent packet loss.

- Strengths:

  - Estimates available bandwidth to prevent underutilization.

  - Ideal for mobile and lossy wireless networks due to its adaptability to fluctuating conditions.

TCP CUBIC:

- Best for high-speed wireless networks, such as LTE and 5G.

- Strengths:

  - Utilizes aggressive window growth to maximize the utilization of available bandwidth.

  - Effective in stable, high-throughput scenarios, providing better performance in environments where bandwidth is ample.

Overall summary:

TCP Westwood+ and TCP CUBIC offer complementary advantages. Westwood+ excels in adapting to variable network conditions, making it suitable for lossy and mobile environments, while CUBIC is designed for maximizing throughput in high-speed networks, ensuring efficient bandwidth use in stable conditions. Together, they provide a balance of performance and adaptability in diverse networking situations.

In conclusion, various TCP congestion control algorithms, including TCP Reno, TCP CUBIC, TCP Vegas, TCP BBR, DCTCP, and TCP Westwood+, each offer distinct advantages and disadvantages tailored to different network environments.

TCP Reno remains a foundational protocol, effective in moderately stable networks but limited in its responsiveness to high loss scenarios. TCP CUBIC, on the other hand, excels in high-speed networks, effectively utilizing available bandwidth through aggressive window growth. TCP Vegas provides a proactive approach to congestion management by monitoring round-trip time, though it may underperform in short-lived connections and requires accurate RTT measurements.

BBR represents a significant advancement by optimizing throughput based on bottleneck bandwidth and round-trip time, yet it demands sophisticated implementation and careful resource management. DCTCP is specifically designed for data centers, leveraging ECN to manage congestion proactively and ensure low packet loss, but its effectiveness is limited outside of this environment.

Finally, TCP Westwood+ is particularly suited for variable wireless conditions, where it dynamically estimates available bandwidth to adapt to changing network states. The choice of congestion control algorithm should be informed by the specific requirements of the application and the characteristics of the network environment. By understanding the strengths and limitations of these protocols, network designers and operators can make informed decisions to optimize performance and enhance user experiences across diverse networking scenarios.