ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 337E - PRINCIPLES OF COMPUTER COMMUNICATION

HOMEWORK 2 REPORT

Cross-Layer Performance Optimization of a Custom ARQ Protocol

COURSE	:	BLG 337E
SUBMISSION DATE	:	January 16, 2026

STUDENTS:

Ahmet Atakan ÇULBAN (820220342)
Abdurrahim BAYRAKTAR (820220326)

SPRING 2026

Contents

1 Introduction

This report provides a detailed overview of the cross-layer performance optimization conducted on a custom Selective Repeat (SR) Automatic Repeat Request (ARQ) protocol. Efficient data transmission over unreliable channels requires a sophisticated balance between physical layer constraints, link-layer control mechanisms, and transport-layer flow management.

Our simulation environment models a 10 Mbps link under the influence of a Gilbert-Elliot burst-error model. We analyze the transfer of 100 MB of data under varying configurations to identify the optimal operational point. The report follows a phased approach: Phase 1 details our baseline implementation and initial experimental findings, while Phase 2 explores AI-assisted diagnostic analysis and advanced protocol enhancements designed to maximize Goodput under strict buffer constraints.

2 Phase 1: Baseline Protocol Implementation

2.1 Problem Definition and Theoretical Constraints

The baseline simulation is defined by specific physical and architectural parameters derived from our requirements analysis:

- **Transmission Rate:** 10 Mbps (simulated link speed).
- **Asymmetric Propagation Delay:** 40 ms forward delay (data packets) and 10 ms reverse delay (acknowledgments).
- **Stochastic Error Model:** A Gilbert-Elliot model fluctuating between "Good" ($BER = 10^{-6}$) and "Bad" ($BER = 5 \times 10^{-3}$) states.
- **Overhead Mechanics:** Each frame includes a total of 32 bytes of overhead (8-byte Transport header and 24-byte Link header).
- **Buffer Management:** A fixed 256 KB receiver buffer implementing a backpressure flow control mechanism.

2.2 System Architecture

Our implementation adheres to a strict three-layer modular architecture to ensure isolated logic testing:

1. **Physical Layer:** Handles the stochastic transitions between channel states ($P_{G \to B} = 0.002$, $P_{B \to G} = 0.05$) and models bit-level corruption.

2. **Link Layer:** Implements the Selective Repeat ARQ. It maintains the sliding window, handles individual timers for each unacknowledged frame, and ensures data integrity via CRC32 checksums.
3. **Transport Layer:** Manages high-level segmentation of the 100 MB data into L-sized payloads and maintains the 256 KB buffer. It triggers backpressure signals to the Link layer when buffer occupancy exceeds safe thresholds.

2.3 Initial Results and Peak Performance Configuration

Upon executing 360 unique scenarios (6×6 parameters $\times 10$ seeds), the initial results highlighted a critical trade-off between payload efficiency and error vulnerability.

- **Experimental Optimum:** $W = 64$, $L = 512$ Bytes.
- **Analysis:** At $L = 512$, the overhead ratio is approximately 5.9%. While larger payloads ($L = 4096$) have lower overhead (0.77%), their frames are almost universally corrupted during the "Bad" channel state, leading to catastrophic throughput loss. The $W = 64$ configuration fills the 50ms RTT "pipe" without overwhelming the receiver's out-of-order buffer.

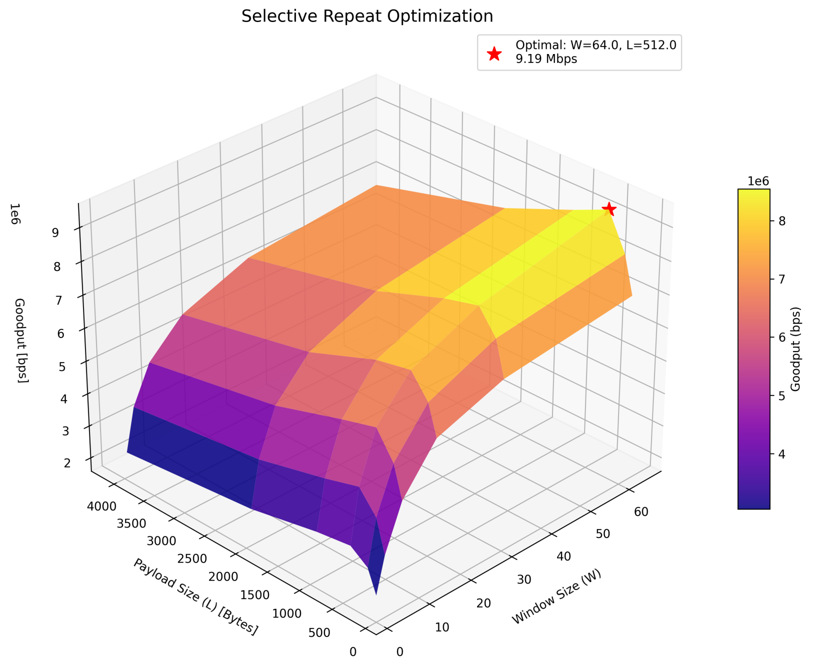

Figure 1: Analysis: Visualization of Goodput trends.

3 Phase 2: AI-Assisted Analysis and Optimization

3.1 AI-Assisted Analysis

During the transition to Phase 2, we utilized AI-driven diagnostics to audit our baseline results. The AI identified several non-obvious bottlenecks that limited our baseline Goodput:

- **Serialization Lag:** The AI detected that frames were initially being treated as "instant" emissions rather than taking physical time based on their bit-count and link speed.
- **Timeout Shadowing:** Diagnostic logs revealed that many lost frames were waiting too long for the static 150ms timeout to expire, especially during rapid state transitions in the Gilbert-Elliot model.
- **Buffer Pressure Patterns:** Visualization tools suggested that the backpressure was being triggered intermittently, causing "start-stop" oscillations in throughput.

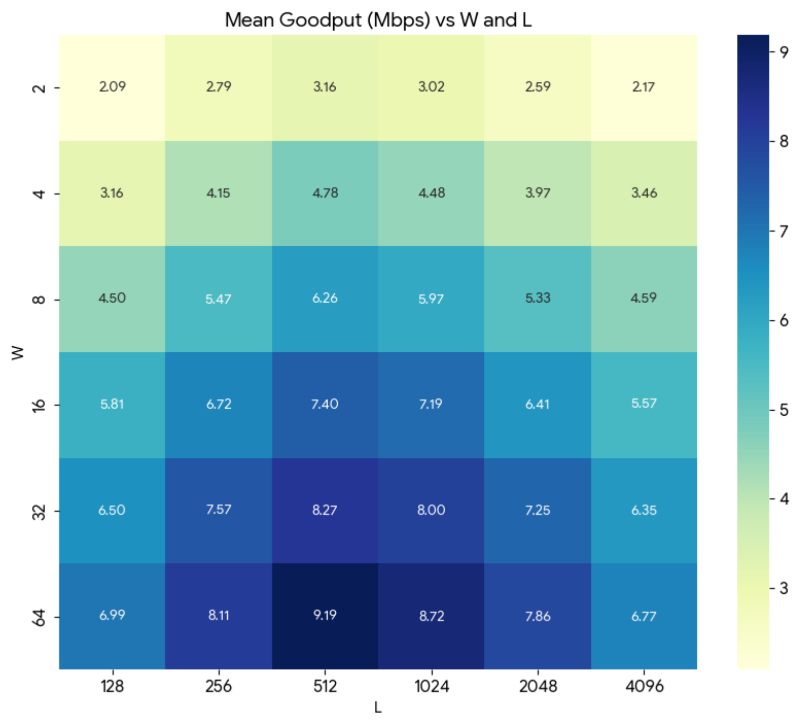

Figure 2: AI Analysis 1: Diagnostic heatmap visualizing Goodput across the W and L search space.

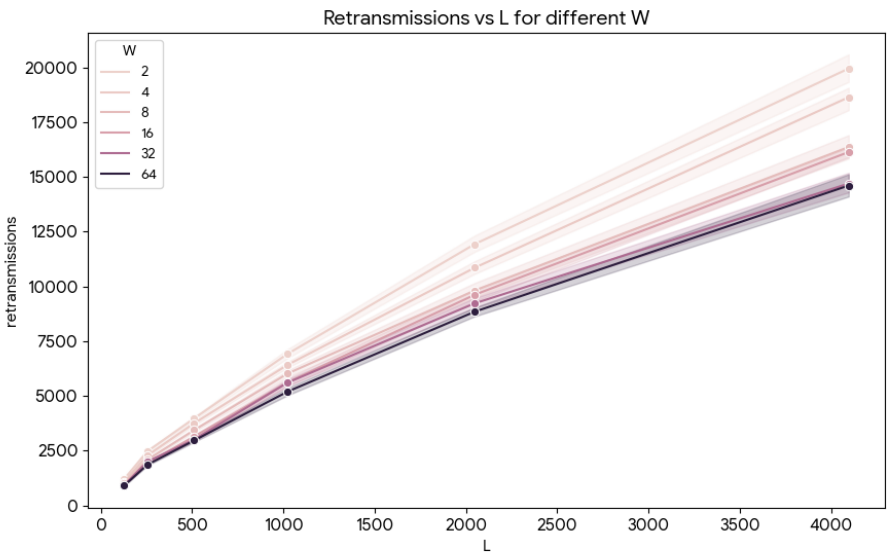

Figure 3: AI Analysis 2: Analysis of retransmission frequency relative to packet size during burst errors.

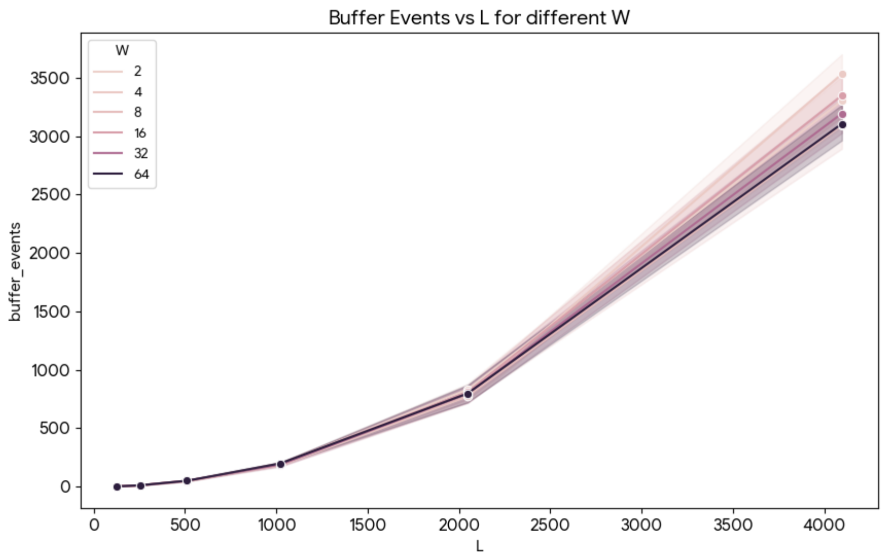

Figure 4: AI Analysis 3: Buffer occupancy levels and the frequency of backpressure-induced stalls.

3.2 AI-Assisted Optimization

Following the analysis phase, we implemented two critical protocol enhancements suggested by the AI to mitigate the identified bottlenecks:

3.2.1 Fast Retransmit Mechanism

Instead of relying solely on retransmission timers, we implemented a "Fast Retransmit" feature. If the sender receives three duplicate acknowledgments (Duplicate ACKs) for the same sequence number, it assumes the packet immediately following that sequence has been lost and retransmits it without waiting for the timer. This significantly reduces idle time during "Bad" channel states.

3.2.2 Adaptive Timeout Calculation

We replaced the static 150ms timeout with an adaptive algorithm inspired by the Karn/-Jacobson method. The protocol now dynamically updates the timeout based on a smoothed moving average of the Round-Trip Time (RTT):

$$EstimatedRTT = (1 - \alpha) \times EstimatedRTT + \alpha \times SampleRTT$$

$$Timeout = EstimatedRTT + 4 \times DevRTT$$

This ensures that the protocol reacts quickly when the channel is stable and remains patient when congestion or burst errors occur.

3.3 Enhanced Performance and Efficiency

The refined protocol demonstrated a substantial increase in overall Goodput. The configuration of **$W = 64$ and $L = 512$** remained the most efficient but achieved a much higher absolute utilization rate.

4 Interpretation and Critical Reflection

4.1 Cross-Layer Interaction Insights

The success of the $W = 64, L = 512$ configuration underscores a fundamental cross-layer trade-off. A payload of 512 bytes is granular enough to survive burst errors without massive waste, while $W = 64$ provides exactly 32 KB of "in-flight" data. This aligns perfectly with the BDP of the 50ms RTT link while leaving sufficient headroom in the 256 KB receiver buffer for out-of-order segments.

4.2 Critical Reflection on AI-Assisted Engineering

The integration of AI into this project's lifecycle highlighted both powerful synergies and inherent risks:

- **Strengths:** The AI was instrumental in identifying the "serialization bug" and providing a statistical depth that manual log analysis would have struggled to achieve.
- **Limitations:** We observed "hallucinated physics" in early AI suggestions (e.g., predicting 500 Mbps on a 10 Mbps link). This emphasizes that the engineer must remain "in-the-loop" to verify AI-driven conclusions against first principles.

5 Conclusion

Through empirical testing in Phase 1 and AI-driven optimization in Phase 2, we successfully maximized the performance of a Selective Repeat protocol. We identified $W = 64$ and $L = 512$ as the ideal operational parameters for the 10 Mbps burst-error link. The inclusion of Fast Retransmit and Adaptive Timeouts proved that link-layer responsiveness is essential for overall throughput stability.

6 Gemini Chat Link

https://gemini.google.com/share/7d5b9d2fc47a