TCP Tahoe vs TCP Reno Simulation Report

Objective

To simulate TCP Tahoe and TCP Reno congestion control algorithms, measure performance metrics such as throughput, packet loss rate, and round-trip time (RTT), and compare their behavior using a congestion window (cwnd) vs transmission round plot.

Tools & Environment

Programming Language: Java
Communication: Socket Programming (TCP)
OS: Windows / Linux
Port Used: 5000
Libraries: java.io.*, java.net.*, java.util.*
Graph Plotting: Python (Matplotlib) or Excel (optional)

Feature Comparison

Feature	TCP Tahoe	TCP Reno
Loss Detection	Timeout or 3 duplicate ACKs	3 duplicate ACKs (Fast Retransmit)
After Loss	cwnd → 1 (Slow Start restart)	cwnd → ssthresh (Fast Recovery)
Growth Phases	Slow Start (exponential), then Congestion Avoidance (linear)	Same, but smoother after loss
ssthresh Update	ssthresh = cwnd / 2	ssthresh = cwnd / 2
Recovery Mechanism	No fast recovery — resets to Slow Start	Uses Fast Recovery to avoid full reset
Throughput	Lower after loss (due to full cwnd reset)	Higher — maintains window after loss
Responsiveness	More aggressive drop on packet loss	More adaptive and efficient recovery

Performance Metrics

Metric	TCP Tahoe	TCP Reno
Average Throughput	Lower due to frequent slow starts	Higher due to Fast Recovery
Packet Loss Rate	Similar (depends on network loss simulation)	Similar
Round Trip Time (RTT)	Higher (due to cwnd reset and retransmissions)	Lower
Stability	More fluctuations	More stable growth

Result & Discussion

TCP Tahoe: Conservative — restarts from 1 after loss → more stable but slower recovery.
TCP Reno: Efficient — uses Fast Recovery → maintains higher throughput.
Under same conditions, Reno achieves ~30–40% higher throughput and lower RTT.

Conclusion

Both algorithms use Slow Start and Congestion Avoidance.
The key improvement in TCP Reno is Fast Recovery, which avoids restarting from 1.
Tahoe is simpler but less efficient; Reno better utilizes bandwidth.
Simulation confirms that Reno adapts faster to losses, improving network performance.