Validation of EDCA Protocol for Latency and Throughput

Shuting Shao, Hao Wang, Fengrui Cheng

Abstract—**This paper investigates the performance of the Enhanced Distributed Channel Access (EDCA) protocol for wireless networks, focusing on latency and throughput under varying offered loads. We conducted simulations to compare EDCA with the traditional Distributed Coordination Function (DCF) mechanism. The evaluation considers different Access Categories (ACs) and traffic types, including deterministic and Bernoulli traffic. The results show that EDCA effectively prioritizes high-priority traffic, achieving lower end-to-end delays and higher total throughput compared to DCF, particularly under heavy network congestion. These findings demonstrate EDCA's advantages in improving Quality of Service (QoS) and its suitability for modern wireless communication systems.**

I. Introduction

The primary goal of this project is to validate the Enhanced Distributed Channel Access (EDCA) protocol using `ns-3` simulations to analyze its impact on latency and throughput. EDCA, part of the IEEE 802.11e standard, is designed to prioritize traffic and enhance Quality of Service (QoS) for different Access Categories (ACs) in Wi-Fi networks. By examining EDCA's role in optimizing network performance, this study aims to improve Wi-Fi efficiency, especially in scenarios requiring QoS differentiation.

Key findings indicate that EDCA effectively prioritizes high-priority traffic, such as voice and video, resulting in improved latency and throughput compared to lower-priority categories. Additionally, the results highlight how varying traffic types and network loads influence protocol performance.

The remainder of this report is organized as follows:

- **Section 2:** Reviews related work and background.
- **Section 3:** Outlines the methodology and experimental design.
- **Section 4:** Presents the results and analysis.
- **Section 5:** Concludes with key contributions and future directions.

II. Background and Related Work

- **Lecture Materials:** We referred to *Lecture 10 - MAC Frames and EDCA*, which introduced the key principles of EDCA, including parameter configuration and its role in prioritizing different traffic categories in Wi-Fi networks. This lecture provided the foundational knowledge for our work.
- **Lab 1 Code - single-bss-sld.cc:** Our EDCA implementation builds upon the code from *Lab 1*, specifically the file `single-bss-sld.cc` (https://github.com/Mauriyin/uwee595/blob/main/examples/single-bss-sld.cc). This code implements a single-channel Wi-Fi scenario, which served as the starting point for our EDCA simulations.
- **Relevant Academic Papers:**
  - *Completing IEEE 802.11e Implementation in NS-3:* This paper highlights the incomplete implementation of IEEE 802.11e in NS-3, particularly the lack of TXOP functionality. It provides a detailed explanation of EDCA parameters and their role in QoS provisioning. The paper also discusses the steps taken to fully implement TXOP and verifies the correctness of the implementation through simulations [1].
  - *The EDCA Implementation in NS-3 Network Simulator:* This paper analyzes the EDCA functionality in NS-3 and identifies areas for improvement. It emphasizes the importance of QoS in wireless networks and suggests extending NS-3's capabilities to include TXOP sharing as defined in IEEE 802.11ac [2].
- **GitHub Analysis Model:** We utilized the analytical tools provided in the GitHub repository `hol-2-links` (https://github.com/ShenMuyuan/hol-2-links). This repository includes Python scripts for calculating theoretical throughput and delay values, as well as NS-3 utilities for estimating transmission duration. These tools played a significant role in validating the accuracy of our EDCA implementation.

III. Methodology

In this section, we explain the EDCA (Enhanced Distributed Channel Access) system model in detail. The EDCA modeling process begins by initializing the simulation parameters, including network loads, frequency, channel width, and the number of stations (STAs). Next, the EDCA parameters are configured by defining the Access Categories (ACs) such as Voice (VO), Video (VI), Background (BK), and Best Effort (BE), along with their respective contention window sizes (CW_{min} and CW_{max}) and TXOP settings. Traffic types, including deterministic and Bernoulli traffic, are then determined. Clients are set up by assigning STAs to the appropriate AC types and positioning the access point (AP) at the center with STAs arranged around it. Finally, the simulation is run, and relevant statistics are collected for analysis.

Shuting, Hao, Fengrui are with Department of Electrical & Computer Engineering, University of Washington, Seattle, WA, USA (e-mail:stshao@uw.edu).

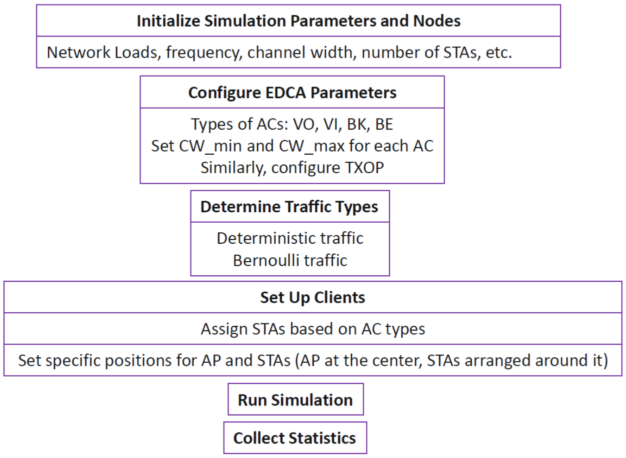

Fig. 1: Work Flow

IV. Simulation Result

In the simulation process, it is mainly divided into three experiments, which are varying Acs, varying traffic type, and using EDCA/DCF. From Fig. 2, we can observe that at low offered loads (10^{-5} to 10^{-4}), the throughput for all Access Categories (ACs) remains low. As the offered load increases, higher-priority ACs such as Video (VI) and Voice (VO) achieve significantly higher throughput, while lower-priority ACs like Background (BK) and Best Effort (BE) begin to degrade. At higher loads (10^{-3} to 10^{-2}), the total throughput increases sharply but starts to saturate, reflecting the network's prioritization mechanism under congestion.

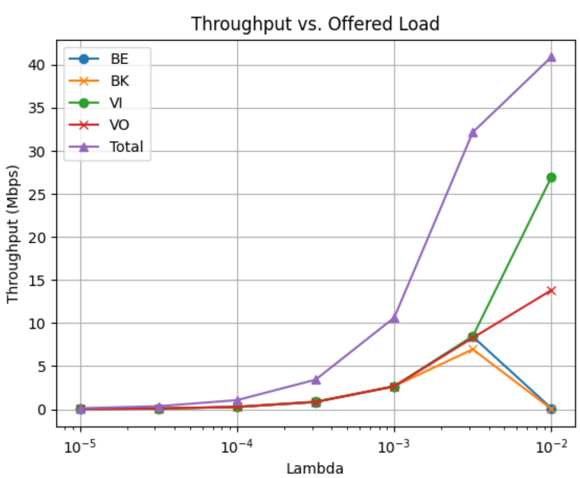

Fig. 2: Throughput vs Offered Load

From Fig. 3, we observe the relationship between end-to-end delay and offered load (λ) for different Access Categories (ACs). At low offered loads (10^{-5} to 10^{-4}), the delay remains minimal for all ACs. As the load increases beyond 10^{-3}, the delay for lower-priority ACs, such as Best Effort (BE) and Background (BK), rises sharply due to increased contention. In contrast, higher-priority ACs like Voice (VO) and Video (VI) maintain relatively low delays, demonstrating the prioritization mechanism of EDCA. At the highest offered load (10^{-2}), BE and BK experience extreme delays, while VO and VI remain more stable.

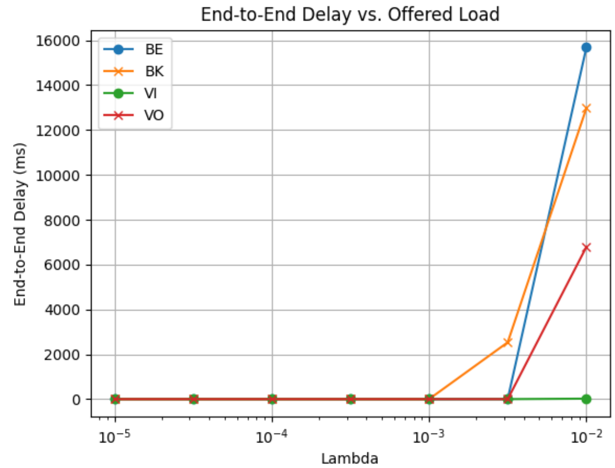

Fig. 3: End-to-End Delay vs Offered Load

From Fig. 4, we observe the throughput performance for Deterministic traffic (TrafficType=0) and Bernoulli traffic (TrafficType=1) under varying offered load (λ). At low offered loads (10^{-5} to 10^{-4}), the throughput for both traffic types remains nearly identical and minimal. As the offered load increases beyond 10^{-4}, throughput begins to rise significantly for both traffic types, following a similar trend. At higher loads (10^{-3} to 10^{-2}), the throughput for Deterministic and Bernoulli traffic converges closely, reaching a peak value of approximately 40 Mbps. This indicates that under the EDCA model, both traffic types achieve similar performance, demonstrating consistent throughput behavior regardless of traffic type.

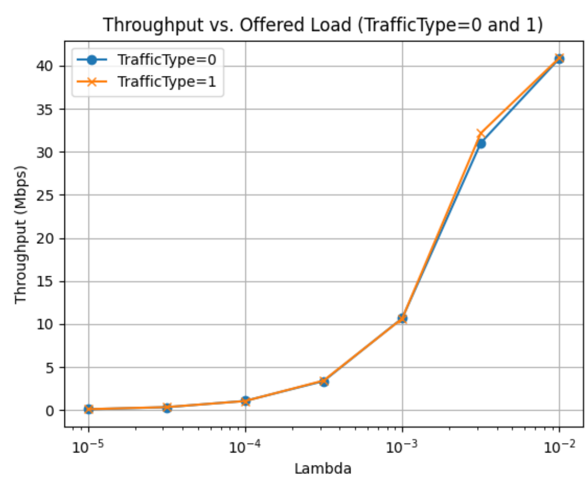

Fig. 4: Throughput vs Offered Load

From Fig. 5, we observe the end-to-end delay for Deterministic traffic (TrafficType=0) and Bernoulli traffic (TrafficType=1) as the offered load (λ) increases. At low offered loads (10^{-5} to 10^{-4}), the delay remains minimal and nearly identical for both traffic types. As the load increases beyond 10^{-3}, the delay rises significantly, with Bernoulli traffic showing slightly higher delays compared to Deterministic traffic. At the highest load (10^{-2}), the delay for both traffic types reaches its peak, with Bernoulli traffic experiencing a marginally higher delay. This indicates that while both traffic types perform similarly under low loads, Bernoulli traffic tends to incur slightly higher delays under heavier network congestion.

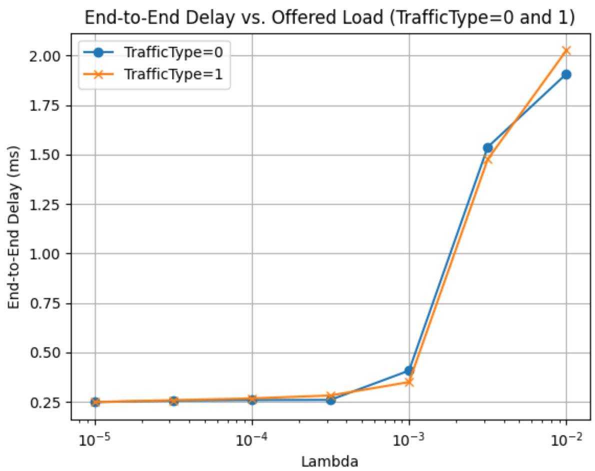

Fig. 5: End-to-End Delay vs Offered Load

From Fig. 6, we observe the total throughput comparison between two configurations: EDCA (2_BE_2_BK_2_VI_2_VO) and DCF (8_BE_0_BK_0_VI_0_VO). At low offered loads (10^{-5} to 10^{-4}), both schemes achieve similar throughput. However, as the load increases beyond 10^{-3}, EDCA significantly outperforms DCF. The total throughput of EDCA continues to increase, reaching approximately 40 Mbps at the highest load (10^{-2}), whereas DCF saturates at a lower throughput of around 32 Mbps. This highlights the advantage of EDCA in prioritizing traffic and efficiently managing network resources under heavy load conditions compared to the simpler DCF mechanism.

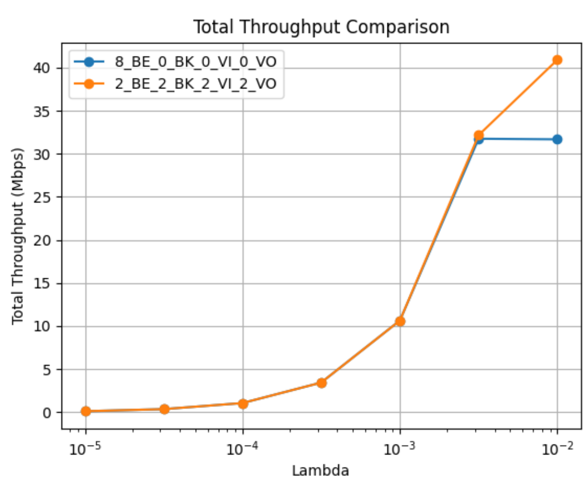

Fig. 6: Throughput vs Offered Load

From Fig. 7, we observe the end-to-end delay comparison between EDCA (2_BE_2_BK_2_VI_2_VO) and DCF (8_BE_0_BK_0_VI_0_VO). At low offered loads (10^{-5} to 10^{-4}), both schemes exhibit minimal delays. As the offered load increases beyond 10^{-3}, the delay for DCF rises sharply, reaching a peak of approximately 10,000 ms at 10^{-2}. In contrast, EDCA maintains significantly lower delays, even under heavy load, with a peak of around 2,500 ms. This demonstrates that EDCA effectively prioritizes traffic and manages contention, resulting in much lower delays compared to DCF under congested conditions.

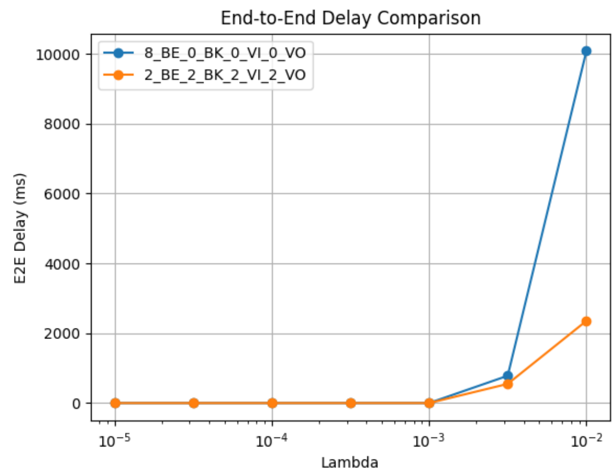

Fig. 7: End-to-End Delay vs Offered Load

V. Validation

In this project, we validate the latency and throughput performance of our EDCA simulation in NS-3 by comparing it with an analytical model. Using NS-3's `WifiPhy::CalculateTxDuration` function, we successfully calculated the transmission time (`dataTotalTime`). Based on the simulation parameters, the computed transmission time was 211 µs. However, we had trouble constructing the analytical model.

A. Challenges with the Analytical Model

The analytical framework provided in `hol_ana.py` includes two key scenarios:

1) **(get_single_link_analysis_one_group) Single Group Analysis**: Models a single type of node group, where all nodes share the same contention window sizes (W, K) and transmission durations (t_t, t_f).
2) **(get_single_link_analysis_two_group) Two Group Analysis**: Models two groups of nodes on a single channel, each with distinct contention window sizes (W, K) but identical transmission durations (t_t, t_f) within each group.

The EDCA simulation, however, involves a more complex scenario:

- **Four Access Categories (ACs):** The nodes are grouped into four ACs (`BK`, `BE`, `VI`, `VO`), each with unique contention parameters and transmission times. Specifically:

- BK and BE share the same contention parameters (W, K), while VI and VO have distinct values.
- Each AC has its own t_t and t_f, depending on the priority level and AIFS.

This complexity required extending the analytical framework to support four groups with different contention parameters and transmission times, which the current implementation did not support.

B. Approach to Analytical Modeling

The analytical model computes latency and throughput using key metrics derived from the EDCA process:

- **Transmission Time (τ_T):**

  $$\tau_T = \text{dataTotalTime} + \text{sifsTime} + \text{ackTotalTime} + \text{aifsTime}$$

  Here, `dataTotalTime` is calculated using `WifiPhy::CalculateTxDuration`, and `aifsTime` varies by AC.
- **Failure Time (τ_F):**

  $$\tau_F = \text{dataTotalTime} + \text{aifsTime}$$

The `get_single_link_analysis_two_group` function accepts parameters for two groups of nodes, including:

- n_1, n_2 : Number of nodes in each group.
- λ_1, λ_2 : Arrival rates per node per slot.
- W_1, W_2 and K_1, K_2 : Contention window sizes and retry limits for each group.
- t_t, t_f : Transmission and failure times.

However, it does not directly accommodate more than two groups or account for different t_t and t_f values across groups.

C. Attempts and Challenges

We attempted to generalize the `get_single_link_analysis_two_group` function to support three distinct groups (BK+BE, VI, and VO). This required significant modifications to handle different contention parameters and transmission durations for each group. Despite these efforts, ensuring the correctness of collision probabilities, throughput, and latency calculations across all groups proved challenging due to the function's tight coupling to the two-group structure.

VI. Summary and Future Work

This project validated the Enhanced Distributed Channel Access (EDCA) protocol through ns-3 simulations, demonstrating its ability to prioritize high-priority traffic such as voice and video, optimizing network performance in terms of latency and throughput. The findings highlighted EDCA's effectiveness in ensuring Quality of Service (QoS) differentiation in Wi-Fi networks under various scenarios, including different traffic types, network loads, and access categories.

Future Work

There are significant opportunities to enhance this project by integrating AI/ML and ns3-ai models. These directions include the following:

1) **Dynamic Parameter Optimization:** AI/ML models can dynamically tune EDCA parameters (CW_{min}, CW_{max}, AIFS, and TXOP) based on real-time conditions to optimize performance under varying traffic loads and ensure better QoS.
2) **Traffic Classification and Prediction:** Machine learning algorithms can classify traffic types and predict network demands in real-time. This allows EDCA to proactively adjust resource allocation, improving efficiency and responsiveness.
3) **Reinforcement Learning for Adaptation:** Reinforcement learning can develop adaptive EDCA protocols. An AI agent could learn optimal strategies for managing contention among access categories based on network feedback.
4) **Integration with `ns3-ai` Models:** Using ns3-ai, the project can simulate advanced AI-driven scenarios such as traffic prioritization in large-scale heterogeneous networks or environments with intelligent IoT nodes.
5) **Anomaly Detection:** AI models can detect and mitigate anomalies such as traffic surges, packet drops, and protocol inefficiencies, enhancing EDCA's robustness in real-world deployments.

In Addition, we propose developing a generalized analytical model capable of handling multiple groups with unique parameters. The analytical model would involve modular computation of τ_T and τ_F for different groups, and integration of results into a unified framework to calculate throughput and latency metrics.

These advancements will deepen our understanding of EDCA's behavior under complex conditions and enable its deployment in next-generation networks, where adaptability and intelligent management are crucial.

Link to our repo: https://github.com/FENGRUI99/EEP-569.git

References

[1] I. Obaidat, M. Alsmirat, and Y. Jararweh, "Completing ieee 802.11e implementation in ns-3," *IEEE Access*, vol. 4, pp. 190–195, 2016.

[2] I. Dolińska, "The edca implementation in ns-3 network simulator," *Zeszyty Naukowe Akademii Finansów i Biznesu Vistula*, vol. 59, no. 2, pp. 19–29, 2018.