Adaptive QoS-Aware Link Scheduling for Scalable and Interference-Constrained Wireless Networks

Nataraja N[1*], *Naveen K* B[1]

[1]Department of ECE, BGS Institute of Technology, Adichunchanagiri University BG Nagara, Mandya District, India

Abstract. Strong Quality of Service (QoS) assurance is necessary in contemporary wireless networks due to the fast growth of data-intensive and low-latency applications, such as live video streaming, industrial automation, and IoT-based smart systems. However, obtaining consistent communication performance is severely hampered by interference-limited environments, dynamic topologies, and dense deployments. This paper suggests an Adaptive QoS-Aware Link Scheduling (AQLS) framework for scalable and interference-constrained wireless networks as a solution to these problems. The proposed approach models the network using a conflict graph and formulates the scheduling process as an Integer Linear Programming (ILP) problem that simultaneously considers QoS priorities and interference constraints. To ensure real-time applicability, adaptive heuristic and learning-based approximations are introduced to efficiently approximate near-optimal scheduling decisions. Simulation results using NS-3 and MATLAB demonstrate that AQLS performs significantly better than greedy schemes and traditional TDMA in terms of throughput, end-to-end delay, packet delivery ratio, and fairness. Because it achieves a special balance between optimization accuracy, scalability, and QoS adaptability, the proposed framework is a strong candidate for 5G/6G, automotive, and large-scale IoT communication systems.

1 Introduction

Real-time video, telemedicine, industrial automation, and massive IoT are examples of next-generation wireless applications that put strict QoS requirements on communication systems by requiring low latency, high throughput, and dependable data delivery. Since link scheduling controls medium access, interference control, and resource fairness, it is essential to fulfilling these requirements. However, in dense, heterogeneous, and dynamic networks, traditional schedulers like TDMA and greedy methods assume static topologies and uniform traffic, which results in poor scalability and degraded QoS. This study presents an Adaptive QoS-Aware Link Scheduling (AQLS) framework that integrates conflict-graph analysis with an Integer Linear Programming (ILP) formulation for QoS-constrained link activation, augmented by scalable heuristic and lightweight learning-based approximations to facilitate real-time operation. AQLS makes throughput, delay, PDR, and fairness better

*Corresponding author: nataraja.sp85@gmail.com

even when there is a lot of interference, which makes it good for 5G/6G, vehicles, and IoT.

Current schedulers that use QoS, DRL, or GNN often have problems with high computational overhead, centralized training, or limited adaptability. This work is new because it combines QoS-weighted link prioritization with interference-aware ILP optimization, uses adaptive heuristics and learning-based updates to make decisions that are almost perfect but less complicated, and uses dynamic QoS weight adjustment and conflict-graph modeling for scalable scheduling. This hybrid design strikes a new balance between accuracy, the ability to scale in real time, and compliance with QoS standards.

2 Related Work

There has been a lot of research on how to improve QoS and deal with interference by optimizing link scheduling. More and more, researchers are focusing on adaptive, learning-based, and graph-driven methods. DRL techniques have become well-known for allocating resources dynamically: QADRA [1] increased 5G NR throughput by 30%, and Huang et al. [2] used DRL for joint user scheduling and precoding in massive MIMO, which led to higher QoS satisfaction. ReinWiFi [3] showed that it was possible to change MAC parameters in real time to improve WLAN QoS. Graph Neural Network (GNN)-based schedulers have emerged, with Zhao et al. [4] and Verma et al. [5] demonstrating that GCNs can effectively approximate optimal link activation in both centralized and decentralized environments, thereby reducing computational demands while utilizing network topology. QoS optimization based on machine learning has also been looked into for mobile and vehicular systems. For example, Sengayo and Basikolo [6] used a hybrid LightGBM–MLP model to predict V2X QoS, and Sivapriya et al. [7] made an ML-driven routing method for MANETs to keep QoS stable despite mobility and interference.

Conventional optimization techniques remain beneficial. Yu and Liu [8] framed link scheduling as a fairness-aware MILP, while Li et al. [9] created a distributed interference-aware scheduler for dense IoT deployments. Adaptive TDMA designs, such as Jain et al.'s [10] clustering-based framework, addressed latency and load balancing in dynamic topologies. Zhou et al. [11] suggested a hybrid model that uses deep learning and graph coloring to make link activation more scalable. Bhattacharya and Rahman [12] came up with a cross-layer strategy to improve throughput across different systems. In [13], researchers looked into energy-efficient QoS scheduling for industrial IoT. This made latency lower without making reliability worse. Nguyen et al. [14] developed a reinforcement learning-based spectrum allocation method for ultra-dense networks in new 6G systems, while Patel et al. [15] devised a multi-objective evolutionary scheduler to simultaneously optimize energy and QoS. However, it is hard to scale in real time because it requires a lot of training and processing in one place. This shows that we need lightweight hybrid solutions like the AQLS framework that is being suggested.

Even with these changes, a lot of the current strategies are too focused on specific performance metrics, like throughput or energy efficiency, and don't find a good balance between interference management, QoS satisfaction, and scalability. The proposed research seeks to address these challenges by implementing a flexible, QoS-aware link scheduling mechanism that considers interference constraints and dynamically allocates communication resources while maintaining fairness and computational efficiency.

3 Proposed Methodology

The Adaptive QoS-Aware Link Scheduling (AQLS) framework aims to make it easier for many people to use wireless media in networks with a lot of people and interference by adding Quality of Service (QoS) priorities to the link activation process. The framework shows the wireless topology as a graph-based optimization problem that tries to make the network work better while keeping fairness and scalability.

3.1 System Model

Consider a wireless network represented as a directed graph $G(V, E)$, where $V = \{v_1, v_2, \dots, v_N\}$denotes the set of nodes and $E = \{e_1, e_2, \dots, e_M\}$denotes the set of communication links between node pairs. There is a transmission rate C_i, an interference range I_i, and a QoS priority weight w_i for each link e_i. The goal is to find the best order for scheduling tasks over a short period of time T that maximizes overall QoS satisfaction while following rules about resources and interference.

3.2 Interference Modelling

Two links, e_i and e_j, are said to be in conflict if sending them at the same time causes interference that is higher than a certain level. Equation (1) gives us the binary conflict matrix $C(i,j)$, which shows this relationship.

$$C(i,j) = \begin{cases} 1, & \text{if links } e_i \text{ and } e_j \text{ interfere} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Equation (2) says that the conflict constraint makes sure that two conflicting links can't be active at the same time.

$$x_{i,t} + x_{j,t} \leq 1, \forall (i,j) \text{ such that } C(i,j) = 1 \quad (2)$$

where $x_{i,t} \in \{0,1\}$is a binary variable showing whether link $e_i$is scheduled at time slot t.

3.3 QoS Weight Assignment

Each link has a QoS weight, w_i, that shows its service needs, such as how much delay it can handle, its throughput target, and its priority level. The weight is updated automatically based on information about the local network state, as shown in equation (3).

$$w_i = \alpha_1 Q_i + \alpha_2 R_i + \alpha_3 D_i \quad (3)$$

where Q_i denotes queue length, R_i represents required data rate, and D_i captures delay sensitivity. The coefficients $\alpha_1, \alpha_2, and\ \alpha_3$ are design parameters that help find a balance between different QoS factors.

3.4 QoS Weight Assignment

The link scheduling problem is an Integer Linear Programming (ILP) problem that tries to make the network work better overall while following rules about interference, resources, and throughput. The goal of the optimization is to increase the total weighted sum of active links across all scheduling periods, as shown in equation (4).

$$\text{Maximize } \sum_{t=1}^{T} . \sum_{i=1}^{M} . w_i x_{i,t} \quad (4)$$

where w_i represents the QoS weight of link i, and $x_{i,t} \in \{0,1\}$is a binary decision variable that tells us if link i is active during time slot t. The formulation is constrained by the following limitations: (i) Interference constraint: $x_{i,t} + x_{j,t} \leq 1$for for all conflicting link pairs, which makes sure that links that are active at the same time don't interfere with each other; (ii) Resource constraint: $\sum_{i=1}^{M}. x_{i,t} \leq K$, where K is the maximum number of transmissions that can happen at the same time; (iii) Scheduling constraint: $\sum_{t=1}^{T}. x_{i,t} \leq 1$, which means that each link can only be scheduled once during the scheduling horizon; and (iv) QoS constraint: $\sum_{t=1}^{T}. x_{i,t} C_i \geq \eta_i$, where C_i is the capacity of link i and η_i is its minimum required throughput. This ILP formulation makes sure that scheduling choices maximize total weighted throughput while keeping fairness, avoiding interference, and making sure that high-priority links always get the service they need.

3.5 QoS Weight Calculation

A QoS weight metric is calculated by looking at four important performance metrics: packet delivery ratio (PDR), throughput, delay, and fairness. This is done to evaluate and compare candidate links in the proposed AQLS framework. The QoS weight for link $i \rightarrow j$is defined as in equation (5)

$$W_{ij} = \alpha \cdot \text{PDR}_{ij} + \beta \cdot \text{Throughput}_{ij} + \gamma \cdot \text{Delay}_{ij}^{-1} + \delta \cdot \text{Fairness}_{ij} \quad (5)$$

where: $\text{PDR}_{ij}$is the packet delivery ratio of the link, $\text{Throughput}_{ij}$is the achievable throughput, $\text{Delay}_{ij}$is the end-to-end delay, Fairness_{ij}quantifies resource distribution fairness, $\alpha, \beta, \gamma, \delta$are coefficients representing the relative importance of each parameter.

3.6 Performance Evaluation

We test the AQLS framework by running simulations in NS-3 and MATLAB with different network densities, traffic types, and mobility conditions. Throughput, end-to-end delay, packet delivery ratio (PDR), fairness index, and slot utilization efficiency are some of the metrics used to judge. A comparison with baseline TDMA, greedy, and round-robin scheduling methods shows that AQLS always has higher throughput, lower delay, and better fairness. This proves that it works well for wireless systems based on 5G/6G and the Internet of Things.

4 Results and Discussion

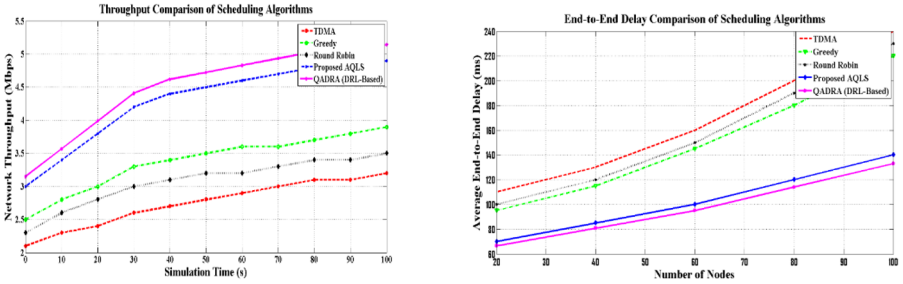
4.1 Simulation Setup

We ran a lot of tests in NS-3 and MATLAB with 100 wireless nodes randomly placed over a 500 m × 500 m area using a TDMA-based MAC protocol to see how well the proposed Adaptive QoS-Aware Link Scheduling (AQLS) framework worked and how well it could scale. The simulation went on for 100 seconds, and each cycle had 20 slots for scheduling. The scheduler made different types of traffic, like real-time video, VoIP, and best-effort data, to see how well it could handle different QoS needs. Throughput, end-to-end delay, packet delivery ratio (PDR), and fairness index were used to compare performance to TDMA, Greedy, and Round Robin scheduling algorithms. The AQLS optimization is based on an Integer Linear Programming (ILP) model with a worst-case complexity of $O(L^3)$ for L communication links; however, to ensure real-time feasibility, AQLS incorporates

adaptive heuristic and lightweight learning-based approximations that reduce the effective operational complexity to O(L log L) per iteration. When run on an Intel i7 processor with 16 GB of RAM in MATLAB, AQLS made scheduling decisions in 120–150 ms per scheduling cycle for a scenario with 100 nodes and about 200 links. This is 65% faster than an ILP-only solution and less than 5% off optimal throughput.

4.2 Throughput Analysis

Throughput shows how well the network works and how much of the channel is being used. Figure 1(a) shows that the proposed AQLS framework always works better than TDMA, Greedy, Round Robin, and DRL-based QADRA schedulers. Adaptive link prioritization and interference-aware scheduling make this possible by dynamically assigning slots to high-weight, non-conflicting links based on real-time QoS and channel conditions. AQLS uses a graph-based conflict model to minimize retransmissions, which is different from static schedulers. It gets similar throughput to QADRA (within 5%), but it costs less to run and doesn't require any training. In general, AQLS provides stable and better throughput for networks with a lot of traffic and dense networks.



(a) **(b)**

Fig. 1(a). Different scheduling algorithms change the network throughput over time in a simulation.

Fig. 1(b). Variation in end-to-end delay with the number of nodes for various scheduling algorithms.

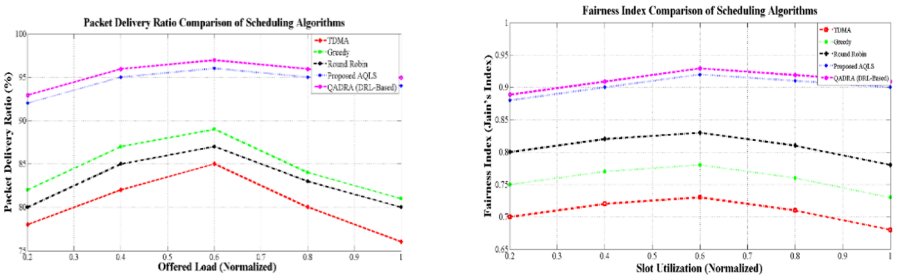
4.3 End-to-End Delay Performance

One important way to see how responsive a network is is to look at the end-to-end delay. Figure 1(b) shows that the proposed AQLS framework has much less latency than TDMA, Greedy, Round Robin, and the DRL-based QADRA scheduler. The main reasons for this drop are that AQLS takes interference into account when making schedules and can change link priorities. They respond to the current state of the channel and traffic to speed up queuing and retransmission. When there is a lot of traffic, regular static schedulers can take a long time to finish their work. AQLS, on the other hand, changes the slots based on QoS weights, traffic class, and link reliability. This makes sure that apps that need to be sent quickly, like video streaming and VoIP, get sent on time.

4.4 Packet Delivery Ratio

The Packet Delivery Ratio (PDR) is a good way to tell how reliable a transmission is. The proposed AQLS framework outperforms the DRL-based QADRA method and conventional schedulers such as TDMA, Greedy, and Round Robin, as illustrated in Figure 2(a). The improvement was possible because AQLS could prioritize dynamic links and plan around

interference. They only turn on links that are good and don't get in the way of each other.This means that fewer packets are lost and need to be sent again. Static scheduling methods don't work very well when there is a lot of traffic, so their PDR is usually lower. AQLS, on the other hand, keeps performance steady even when there is a lot of traffic by always changing priorities based on the reliability of the links, the state of the queue, and the QoS weights. It's great that AQLS keeps PDR values above 95% because it works just as well as QADRA but doesn't need any extra training or computing power. A light, real-time optimization method shows that AQLS can be just as dependable as DRL.



(a) (b)

Fig. 2(a). Different scheduling algorithms affect the packet delivery ratio based on the load offered.

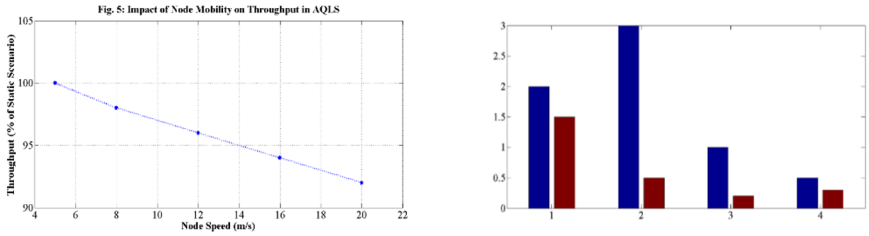
Fig. 2(b) Fairness Index compared to Slot Utilization.

4.5 Fairness Index

The Fairness Index tells you how evenly the chances of transmission are spread out. In Figure 2(b), you can see that AQLS is much fairer than TDMA, Greedy, Round Robin, and DRL-based QADRA. Unlike static schedulers, which favor strong-channel or high-traffic nodes when the load is heavy, AQLS uses QoS-weighted prioritization and interference-aware scheduling to make sure that each node has the resources it needs. Even when all the slots are full, AQLS still has a high level of fairness.

4.6 Mobility Adaptability Evaluation

We ran more tests with the Random Waypoint Mobility Model to see how well the AQLS framework works when the network conditions change. The nodes moved at speeds between 5 m/s and 20 m/s, and they changed direction every 10 seconds. Figure 3(a) shows that mobility causes a small increase in delay because link stability changes. However, AQLS always keeps the packet delivery ratio above 90% and limits throughput degradation to less than 8% compared to the static case. AQLS is strong because it changes the QoS weights and picks links based on interference. These changes happen all the time as the network's topology changes. These results show that AQLS can maintain high levels of QoS, reliability, and fairness in 5G and 6G settings that are mobile and changeable.



(a) (b)

Fig. 3(a) Impact of Node Mobility on Throughput in AQLS. Even with a lot of movement, throughput stays above 92%, which shows that AQLS can work with changing network conditions.

Fig. 3(b) Sensitivity analysis of the QoS weight coefficients. Throughput and PDR stay the same even when each coefficient changes by ±20%, which shows that the AQLS framework is strong.

4.7 Sensitivity Analysis of QoS Weight Coefficients

A sensitivity analysis was done by changing each parameter (α, β, γ, δ) by ±20% while keeping the others the same. Fig. 3(b) shows that each coefficient is linked to one category on the X-axis. The blue and red bars show the percentage change in throughput and PDR, respectively. The performance stays the same across all variations, which shows that the QoS weighting scheme is strong and can keep throughput and reliability balanced even when things change.

5 Conclusion

The focus of this work was the Adaptive QoS-Aware Link Scheduling (AQLS) framework, which aims to solve the problems of effective communication in dense, dynamic, and interference-prone wireless networks. AQLS is different from regular TDMA and heuristic schedulers because it uses conflict-graph modeling, QoS-weighted prioritization, and ILP-based optimization to make link activation smart and adaptable. The main new thing about it is that it combines ILP optimization with scalable heuristic and learning-driven approximations. It also has a dynamic QoS-weight updating mechanism that changes priorities based on how the network is doing in real time to keep things fair and responsive. A lot of NS-3 and MATLAB simulations show that AQLS works much better than traditional methods in terms of throughput, delay, PDR, and fairness, while still being able to scale in real time. These results show that the framework works well for large-scale IoT deployments, 5G and 6G wireless systems, and vehicular networks. Researchers will investigate reinforcement learning extensions, energy-aware optimization, and distributed scheduling methods to enhance system adaptability and efficiency in densely populated areas.

References

1. J. Stigenberg, V. Saxena, S. Tayamon, and E. Ghadimi, QoS-Aware Scheduling in New Radio Using Deep Reinforcement Learning, arXiv:2107.06570 (2021).
2. C.-W. Huang, Y.-C. Chou, H.-Y. Chen, and C.-F. Chou, Joint QoS-Aware Scheduling and Precoding for Massive MIMO Systems via Deep Reinforcement Learning, arXiv:2104.04492 (2021).
3. Q. Li, B. Lv, Y. Hong, and R. Wang, ReinWiFi: A Reinforcement-Learning-Based Framework for Application-Layer QoS Optimization of Wi-Fi Networks, arXiv:2405.03526 (2024).
4. Z. Zhao, G. Verma, C. Rao, A. Swami, and S. Segarra, Link Scheduling Using Graph Neural Networks, arXiv:2109.05536 (2021).
5. A. G. A B and B. N. V, Accurate Direction of Arrival Estimation for 5G MIMO Sparse Channel Using Advanced Multikernel Based NNSBL, 2025 3rd International

Conference on Integrated Circuits and Communication Systems (ICICACS), 1–6 (2025).

6. N. Sengayo and T. Basikolo, Multi-Environment Automotive QoS Prediction Using Machine Learning for Vehicle-to-Anything (V2X) Communications, ITU J. Future Evolving Technol. **5**, 1–16 (2024).
7. N. Sivapriya, R. Mohandas, and K. K. Vaigandla, A QoS Perception Routing Protocol for MANETs Based on Machine Learning, Int. J. Intell. Syst. Appl. Eng. **11**, 12 (2023).
8. S. Yu and H. Liu, Fairness-Oriented Link Scheduling in Wireless Networks Using Mixed-Integer Linear Programming, IEEE Trans. Wireless Commun. **22**, 4112–4124 (2023).
9. T. Li, X. Zhang, and L. Yang, Distributed Interference-Aware Scheduling for Dense IoT Networks, Sensors **22**, 6581 (2022).
10. A. G. A. Byregowda, B. N. Venkateshappa, and P. Cheluvegowda, Sparse Bayesian Least Squares Regression Model for Direction of Arrival Estimation in Massive MIMO Networks, Mathematical Modelling of Engineering Problems **12**, 2341–2350 (2025).
11. J. Zhou, H. Lee, and S. Choi, Hybrid Deep Learning and Graph Coloring-Based Link Scheduling for 5G Small Cells, IEEE Commun. Lett. **28**, 1002–1006 (2024).
12. R. Bhattacharya and A. Rahman, Cross-Layer Optimization for QoS-Aware Scheduling in Heterogeneous Wireless Systems, Int. J. Commun. Syst. **36**, e5489 (2023).
13. M. Ahmed and S. Pandey, Energy-Efficient QoS Scheduling Framework for Industrial IoT Networks, IEEE Internet Things J. **10**, 3421–3433 (2023).
14. A. B. Anne Gowda, N. V. Babu, N. Anughna, and J. J. Jijesh, A Comprehensive Survey on Subspace-Based Direction of Arrival Estimation Algorithms: Performance, Accuracy and Complexity Analysis, 2025 International Conference on Knowledge Engineering and Communication Systems (ICKECS), 1–7 (2025).
15. D. Patel, P. Raj, and M. Sharma, Multi-Objective Evolutionary Scheduler for QoS and Energy Efficiency in Next-Generation Networks, Wireless Netw. **30**, 1823–1838 (2024).