**Dissertation**

# Protocols and Algorithms for reliability-constrained Quality-of-Service Routing in IEEE 802.11 (WiFi) Networks in Contention Mode

Thesis approved by the Department of Computer Science
of the University of Kaiserslautern (TU Kaiserslautern)
for the award of the Doctoral Degree

**Doctor of Engineering (Dr. -Ing.)**

to

**Kiran Mathews**

| | |
|---|---|
| Date of Defense: | September 05, 2022 |
| Dean: | Prof. Dr.-Ing. Jens Schmitt |
| *PhD Committee* | |
| Chair: | Prof. Dr.-Ing. Stefan Deßloch |
| Reviewers: | Prof. Dr.-Ing. Reinhard Gotzhein |
| | Prof. Dr.-Ing. Jens Schmitt |

D 386

# Abstract

Several applications have emerged and benefited from the recent advancements in wireless communication technologies. In the case of industrial automation, the wireless networks substituted wired networks to control and monitor the production systems and the factory environment. In such use cases, a common requirement is communication reliability. Technologies based on IEEE 802.15.4, such as WirelessHart and ZigBee developed for industrial applications, offer deterministic guarantees using reservation-based medium access. However, it is becoming more challenging for these technologies to guarantee their sufficiently predictable behavior, as the number of consumer electronics equipped with wireless communication technologies operating in the 2.4 GHz ISM band shared by IEEE 802.15.4 is increasing day by day.

Meanwhile, developments in WiFi technology opened the opportunity to use WiFi for industrial applications. Compared to the technologies based on IEEE 802.15.4, WiFi offers significantly higher transmission rates, and the off-the-shelf commodity WiFi hardwares are available at a low cost. However, when using a contention-based technology such as WiFi for industrial applications, additional measures are required to guarantee the specified statistical reliability.

This thesis lays the foundations for developing a multi-hop wireless control network using off-the-shelf IEEE 802.11 (WiFi) hardware operating in contention mode that can satisfy the specified reliability requirements of the applications. In a multi-hop wireless network, the communication reliability between the nodes depends on the routes determined by the routing protocol and managing these routes. We introduce a novel Quality-of-Service (QoS) routing protocol for contention-based wireless technologies such as WiFi that prioritizes reliability as the QoS requirement for route selection. The proposed routing protocol relies on different aspects of the network to determine and manage the routes. For instance, it requires algorithms and protocols to monitor and measure link quality, available bandwidth, or medium overload. Further, the determined routes require certain statistical link properties for the successful operation of the routes. We develop and evaluate different protocols, algorithms, and metrics to monitor and measure different aspects of the network in this thesis.

*"I constantly see people rise in life who are not the smartest, sometimes not even the most diligent, but they are learning machines,"*

— Charlie Munger

# Acknowledgments

This thesis would not be possible without several people's help and constant support. I express my deepest gratitude to my supervisor, Prof. Dr. Reinhard Gotzhein. Shortly after joining the master's program at the Technical University of Kaiserslautern, I was fortunate to be mentored by Prof. Dr. Gotzhein throughout my master's studies. It helped me to adapt to a different education system than I had experienced and a portal to scientific research. After completing my master's studies, I had further opportunities to learn and contribute to the scientific community as a Ph.D. candidate under his supervision. During my research, his input during our regular meetings helped me significantly. Further, I thank him for proofreading my scientific publications, especially parts of this thesis. I couldn't imagine a better supervisor.

I thank Prof. Dr. Jens Schmitt for being a co-supervisor of this thesis, and for his helpful insights during my research, particularly during the meetings discussing the research works regarding the statistical link properties. I sincerely thank Prof. Dr. Stefan Deßloch for being the chairman of my Ph.D. defense.

I was fortunate to be a part of the Networked Systems Group at the Technical University of Kaiserslautern with helpful colleagues. I thank Dr. Christopher Kramer, Christopher Kohlstruck, Dr. Markus Engel, and Paulo Aragao for helping during my research, especially with experiments conducted in the basement, and for proofreading my papers. Thanks to our group secretary, Anja Gerber, for helping me with the administrative processes and administrative paper works. A special thanks to their partners, or sometimes them, for the home-made cakes.

I thank Dr. Matthias Schäfer of the DISCO research team at the Technical University of Kaiserslautern for his help and support during my early days at the university. A special thanks to Vipin Madavanunni, who provided me with much technical information regarding WiFi drivers.

I am always grateful to my family for all of the achievements I have had in my life. I thank my parents (Mathews and Rosily) and my little sister, Keerthy, for their lifelong support and motivation. I thank my best friend and wife, Jeethu Jose, for her constant support and motivation through my good and bad times. Last but not least, I thank my dear friends and everyone who helped me through this journey of my life. Thank you.

# Contents

**List of Figures** xvii

**List of Tables** xx

**List of Abbreviations** xxi

**1 Introduction** 1

1.1 Problem Statement . . . 2
1.2 Approach . . . 4
1.3 Contributions and the outline . . . 6

**2 $r_{min}$-Routing :- A Protocol for Statistically Reliable Routing for Contention-based Wireless Networked Control Systems** 9

2.1 Introduction . . . 10
2.2 $r_{min}$-routing . . . 10
2.2.1 State of the art . . . 11
2.2.2 Motivation . . . 12
2.2.3 $r_{min}$ routing metric . . . 15
2.2.4 Route Selection . . . 20
2.2.5 Experiments . . . 23
2.2.6 Conclusion . . . 31
2.3 Further contributions . . . 32
2.4 Implementation framework and the hardwares . . . 33
2.4.1 WiPS Framework . . . 34
2.4.2 Hardwares used in evaluations . . . 35
2.5 Summary . . . 37

**3 Statistical Foundations for Improving the Reliability of Wireless Links through Retransmissions** 39

3.1 Introduction . . . 40
3.2 State of the art . . . 42
3.3 Statistical Methods . . . 44
3.3.1 Gilbert-Elliot Model . . . 44
3.3.2 Runs-Test . . . 46
3.3.3 Chi-square test . . . 48

3.4 Experimental Assessment . . . 50
3.4.1 Experiment Setup . . . 50
3.4.2 Synthetic data . . . 54
3.4.3 Real link data from a collision-free environment . . . 64
3.4.4 Environment with foreign traffic . . . 68
3.5 Conclusion . . . 79

**4 Applying Statistical Methods to Identify *Link Boosting* Capability of Wireless Links 81**
4.1 Introduction . . . 82
4.2 Link Boosting . . . 82
4.2.1 Methodology . . . 84
4.2.2 Evaluation techniques . . . 88
4.3 Experimental Assessment . . . 89
4.3.1 Evaluation using Synthetic data . . . 89
4.3.2 Evaluation using Real world link data . . . 105
4.4 Measuring the "margin of safety" required to tolerate variation in statistical properties of a link . . . 109
4.4.1 Foundations . . . 110
4.4.2 Influence of sample size . . . 113
4.4.3 Influence of target reliabilities . . . 115
4.5 Conclusion . . . 119

**5 Time Token Bucket Algorithm 121**
5.1 Introduction . . . 122
5.2 Medium Management Layer . . . 123
5.3 Time Token Bucket Algorithm (TTBA) . . . 125
5.3.1 State of the art . . . 126
5.3.2 Operation of the TTBA . . . 129
5.3.3 Dynamic token refilling to introduce randomness . . . 138
5.4 Experiments . . . 141
5.4.1 Evaluating the traffic monitoring metric, *unusable wasted assigned medium occupancy ratio* . . . 141
5.4.2 Evaluating the *dynamic refilling* mechanism . . . 150
5.5 Algorithms and protocols using the functionalities of the MML . . . 163
5.5.1 TTBA for Cooperative Fair Medium Sharing . . . 163
5.5.2 Route selection based on the medium conditions . . . 164
5.6 Conclusion . . . 164

**6 OB-EWMA: A Link Metric for Reliability-constrained Routing in Wireless Networks 167**
6.1 Introduction . . . 168
6.2 State of the art . . . 168
6.3 Outlier Bounded Exponential Weighted Moving Average (OB-EWMA) 170
6.3.1 Motivation . . . 170
6.3.2 Definition of OB-EWMA . . . 171
6.4 Link metrics used in evaluation . . . 175
6.4.1 Window Mean with EWMA (WMEWMA) . . . 175
6.4.2 FaLQE . . . 176
6.4.3 F-LQE . . . 177
6.4.4 *min*-EWMA . . . 179
6.5 Experimental Evaluation . . . 179
6.5.1 Properties of PDR distribution . . . 180
6.5.2 Metric Evaluation . . . 182
6.5.3 Routing Experiments . . . 189
6.6 Conclusion . . . 193

**7 Conclusions 195**
7.1 Summary . . . 196
7.2 Limitations and Future Works . . . 198

**List of own publications 201**

**Bibliography 203**

**Curriculum Vitae 211**

# List of Figures


2.1 Multiple paths between node $v_1$ and $v_4$ . . . . . 12
2.2 A sample route . . . . . 13
2.3 A 3-hop path $p = \langle e_1, e_2, e_3 \rangle$ . . . . . 16
2.4 Multiple paths between $v_s$ and destination node $v_d$. . . . . 21
2.5 Topology used in the experiments . . . . . 24
2.6 Routes calculated by multiple metrics between from 2 to 6 . . . . . 26
2.7 Success ratio of routes selected by each metric from node 7 to other nodes in the network. The red dotted lined denotes the minimum target path reliability $r_{min}$. . . . . 30
2.8 The architecture of the WiPS framework with multiple protocol layers 34
2.9 Testbed nodes built using APU2 and Raspberry Pi SBCs . . . . . 36

3.1 A 2-state discrete time Markov chain model . . . . . 45
3.2 Number of possible Bernoulli sequence combinations of length 10 and success probability p = 0.5. The x-axis represents the number of runs in a Bernoulli sequence. The y-axis represents the number of possible combinations with the respective number of runs. The shaded region is the rejection area, for alpha = 0.05. . . . . 47
3.3 Histogram showing the observed and expected number of pulls for the number of successful events on a sequence. Sequence and subsequence length of $N = 600$ and $k = 30$, respectively and a success probability of 0.8. . . . . 49
3.4 A bursty Bernoulli Sequence with raw PDR = 0.99. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$. . . . . 55
3.5 A statistically independent Bernoulli Sequence with raw PDR = 0.99. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$. . . . . 56
3.6 An oscillatory Bernoulli Sequence with raw PDR = 0.5. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$. 57
3.7 Statistical methods are applied to a different moving configuration $s_{mw} = 60$ and $s_{Bs} = 30$ to the sequences used in Fig. 3.4, and Fig. 3.5 59
3.8 Sample bursty (statistically dependent) sequences with different success probabilities $pdr^e_{raw}$ generated using 2-state Markov model and test results with moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$. . . . 60

3.9 Sample statistically independent sequences with different success probabilities $pdr^{e}_{raw}$ generated using built-in R methods and test results with moving windows of size $s_{mw} = 100$ and $s_{Bs} = 30$. . . . . 61

3.10 Statistical methods are applied to a moving configuration $s_{mw} = 60$ and $s_{Bs} = 30$ to oscillatory sequences with different $\mu$ values . . . . . 63

3.11 Node placement for the experiments in collision-free environment. The dotted circle denotes the communication range of node A. . . . . 65

3.12 First 10000 event outcomes on link A→B during the second experiment. 65

3.13 Results for the experiment where node B is located at the communication range border of node A. Evaluation uses moving windows of size $s_{mw} = 100$ and $s_{Bs} = 30$. . . . . 67

3.14 Floor plan of the AGVS testbed . . . . . 68

3.15 Link 8 → 15 during experiment using setting *S.A.1* . . . . . 69

3.16 Statistical test results for a set of links during an experiment using traffic generation setting *S.A.1*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$). . . . . 70

3.17 A set of links where non-random and statistically dependent (bursty) frame losses were observed during an experiment using setting *S.A.1*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. . . . . 71

3.18 Statistical test results for a set of links during an experiment using traffic generation setting *S.A.3*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$). . . . . 73

3.19 Links during experiment using setting *S.A.3*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. . . . . 74

3.20 Links during experiment using setting *S.A.4*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. . . . . 75

3.21 $\beta$-metric for links during the entire experiment . . . . . 75

3.22 Statistical test results for a set of links during an experiment using traffic generation setting *S.P.1*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$). . . . . 78

3.23 Statistical test results for the link 11→5. Evaluation uses moving windows of size $s_{mw}$ = 100 and $s_{Bs}$=30. . . . . 79

4.1 Characteristics of the synthesized Bernoulli sequences in our evaluation data set. . . . . 90
4.2 Plots showing the $\mu$-value and Target missed $r^e_{\Delta,missed}$ in each moving window from the entire data set. . . . . 92
4.3 Plots showing the $\mu$-value and Target missed $r^e_{\Delta,missed}$ in moving windows where the $r^e_{\Delta,missed} > 0$. I.e., windows were link boosting failed. 93
4.4 Box plot showing the data distribution of Target missed $r^e_{\Delta,missed}$ in moving windows for a specific range of $\mu$-values where $r^e_{\Delta,missed} > 0$. . 94
4.5 3D Scatter plot showing Target missed $r^e_{\Delta,missed}$ (z-axis), the margin of safety $r^e_{mos}$ (y-axis), and the $\mu$-value (x-axis) during each moving window. . . . . 95
4.6 3D Scatter plot showing Target missed $r^e_{\Delta,missed}$ (z-axis), the margin of safety $r^e_{mos}$ (y-axis), and the $\mu$-value (x-axis) during each moving window depending on $n^e_{max}$. . . . . 97
4.7 Plots showing the $z$-value and Target missed $r^e_{\Delta,missed}$ in each moving window. . . . . 99
4.8 Box plot showing the data distribution of Target missed $r^e_{\Delta,missed}$ in moving windows for a specific range of $z$-values where $r^e_{\Delta,missed} > 0$. . 100
4.9 Correlation between the $\mu$-value and $z$-value during an evaluation with moving window configurations, $s_{Bs}$ = 30 and $s_{mw}$ = 100. . . . . 100
4.10 Plot showing the $chi_q$ and Target missed $r^e_{\Delta,missed}$ during each moving window with a configuration, $s_{Bs}$ = 30 and $s_{mw}$ = 100. . . . . 101
4.11 Histogram showing the distribution of Target missed $r^e_{\Delta,missed}$ in moving windows with a configuration, $s_{Bs}$ = 30 and $s_{mw}$ = 100, where $chi_q > 0$ . . . . . 102
4.12 Plots showing the $chi_q$ and Target missed $r^e_{\Delta,missed}$ in moving windows where the $r^e_{\Delta,missed} > 0$ and $chi_q > 0$. I.e., windows were link boosting failed. Moving window uses a configuration, $s_{Bs}$ = 30 and $s_{mw}$ = 100. 102
4.13 Histogram showing the distribution of Target missed $r^e_{\Delta,missed}$ in moving windows with a configuration, $s_{Bs}$ = 30 and $s_{mw}$ = 100, where $chi_q = 0$. . . . . 103
4.14 Evaluating the effectiveness of link boosting and the response of metrics used to measure the statistical properties on the link 4 → 8. Evaluation uses a moving window configuration of $s_{Bs}$ = 30 and $s_{mw}$ = 100.106

4.15 Performing link boosting on the link 7 → 10. Evaluation uses a moving window configuration of $s_{Bs} = 30$ and $s_{mw} = 100$. . . . . 107
4.16 Performing link boosting on multiple links operating in a sensitive region . . . . . 109
4.17 Plots showing the results of curve-fitting after each step. . . . . 111
4.18 tolerable $\mu$-value while using multiple moving window configurations for a target reliability of $r^e_{target} = 0.98$. Among moving windows with failed link boosting, for a "normalized margin of safety" (the x-axis), the smallest $\mu$-value (the y-axis) observed during a window is plotted. The green, blue, black, and red lines denote the tolerable $\mu$-value for sample sizes of 600, 1800, 3000, and 4200 event outcomes. . . . . 114
4.19 tolerable $\mu$-value while using multiple target reliabilities and same window configurations ($s_{Bs} = 30$ and $s_{mw} = 100$). The green, blue, black, red, brown, yellow, and grey lines denotes the tolerable $\mu$-value for target reliability $r^e_{target}$ 0.99, 0.98, 0.95, 0.93, 0.90, 0.85, and 0.80 event outcomes. . . . . 116
4.20 The minimum required margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1 depending on multiple target reliabilities. . . . . 118
4.21 The minimum required margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1 depending on target reliabilities between 0.85 and 0.99. The red dotted line is the regression line. . . . . 118

5.1 Medium Management Layer in the WiPS . . . . . 122
5.2 Medium Management Layer modules . . . . . 123
5.3 Time Token Bucket Module (Original version using *static refilling*) . . 132
5.4 Time Token Bucket Module with dynamic token refilling interval. . . 140
5.5 $mor^v_{wastedUnUsableRatio}(t)$ for nodes from the experiments on channel 1 with distinct medium requirements, where $t = 20\,\text{min}$. . . . . 147
5.6 $mor^v_{wastedUnUsableRatio}(t)$ for nodes from the experiments on channel 6 with distinct medium requirements, where $t = 20\,\text{min}$. . . . . 148
5.7 $mor^v_{wastedUnUsableRatio}(t)$ for nodes from the experiments in the absence of external traffic with distinct medium requirements, where $t = 20\,\text{min}$. . . . . 149
5.8 Topology used in the experiments . . . . . 151
5.9 Simulation results of link 1 → 2 from experiment B.I . . . . . 154
5.10 Simulation results of link 1 → 2 from experiment B.II . . . . . 156
5.11 Simulation results of link 1 → 2 from experiment B.III . . . . . 159
5.12 Simulation results of link 1 → 2 from experiment B.IV . . . . . 161

6.1 Moving window with size $s_{mw} = 20$ PDR slots [MG21]. . . . . 172
6.2 Node placement in the testbed . . . . . 180
6.3 Density plot of observed PDR values ($pdr^{e}_{raw}(n_{ps})$, where $d_{ps} = 20s$) of a subset of links . . . . . 181
6.4 Observed and estimated (OBEWMA$^{e}_{\alpha,p}$) link qualities of link $e_1 = (15, 8)$ (black coloured) and $e_2 = (14, 10)$ (grey coloured) during an experiment where the estimation metric uses a PDR slot duration of $d_{ps} = 10$ and moving window size $s_{mw} = 30$. . . . . 183
6.5 Applying link metric on link $e_1 = (15, 8)$ where $d_{ps} = 10$ and $s_{mw} = 30$. 183
6.6 Three links $e_3$, $e_4$, and $e_5$ having synthetically generated PDR values where synthesized data have the same mean (0.85) and different standard deviation. . . . . 184
6.7 Applying metrics to raw data for link $e = (3, 5)$ . . . . . 185
6.8 The ratio of wrong estimates $r^{e}_{-}$ during an experiment. . . . . 188
6.9 $r_{min}$-routing experiments on path $p = (5, 3, 8, 15)$ where links used a PDR slot duration $d_{ps} = 20$s. Red line denotes the reliability target $r_{min} = 0.95$ where the green line denotes observed path reliability $r^{p}_{obs}$ 190
6.10 $r_{min}$-routing experiments on path $p = (5, 3, 8, 15)$ where links used a PDR slot duration $d_{ps} = 30$s. Red line denotes the reliability target $r_{min} = 0.95$ where the green line denotes observed path reliability $r^{p}_{obs}$ 191

# List of Tables


2.1 Transmission allocation on each link $e \in p$ after each iteration by MTX to meet the specified target reliability $r_{min} = 0.80$ . . . . . 16
2.2 Transmission allocation on each link $e \in p$ after each iteration by MTX and SMTX to meet the specified target reliability $r_{min} = 0.98$ . 17
2.3 Transmission allocation on each link $e \in p$ after each iteration by SMTX to meet the specified target reliability $r_{min} = 0.90$ . . . . . 21
2.4 Packet format . . . . . 25
2.5 Paths selected by each metric from node 6 to 2 . . . . . 26
2.6 Properties for routes selected by each metric from node 7 to all other nodes . . . . . 29
2.7 Path selected by each metric from node 7 to 20 . . . . . 30

3.1 A raw data sample for the link 10→11 logged at node 11 and its corresponding Bernoulli sequence . . . . . 52
3.2 Frame generation settings used in aperiodic traffic experiments . . . . 68

4.1 A raw data sample on link 10→11 with $n^e_{max} = 2$ . . . . . 83
4.2 Bernoulli sequence with outcome of transmissions with *txCtr* between 1 and 20 for some link $e$ for $n^e_{max} = 1$. . . . . 86
4.3 Scenario when retransmissions are applied ($n^e_{max} = 2$). Superscript denotes the number of transmission attempts. . . . . 87
4.4 The table shows the percentage of moving windows (out to complete data set) with successful link boosting for different tolerance levels. . 104
4.5 Required margin of safety to tolerate up to a $\mu$-value of 0.01. . . . . 117

5.1 Parameters for the experiments A.I and A.II . . . . . 143
5.2 Metric values from the experiment A.I where $t = 3\,300\,000\,000\,\mu s$ or 55 min . . . . . 143
5.3 Metric values from the experiment A.II where $t = 3\,300\,000\,000\,\mu s$ or 55 min . . . . . 144
5.4 Parameters for the experiments A.III . . . . . 145
5.5 Metric values from the experiment A.III where $t = 3,300,000,000\,\mu s$ or 55 min . . . . . 145
5.6 Parameters for the experiments A.IV, and A.V . . . . . 146
5.7 Parameters for the experiments A.VI . . . . . 149
5.8 Application settings for experiments B.I and B.II . . . . . 152

5.9 MML settings for experiment B.I . . . . . 153
5.10 Application settings for experiments B.III and B.IV . . . . . 157
5.11 MML settings for experiments B.III and B.IV . . . . . 160

6.1 Effect of different configuration parameters on link $e$ = (15,8) until the last moving window during the experiment for $p = 0.1$ . . . . . 187
6.2 Effect of different metrics on link $e$ = (15,8) until the last moving window during the experiment, $d_{ps} = 30s$. . . . . 187
6.3 Average transmission budget allocation on route $p = (5, 3, 8, 15)$ calculated by $r_{min}$-routing using different metrics during different experiments . . . . . 193

# List of Abbreviations

| | |
|---|---|
| AP | Access Point |
| API | Application Programming Interface |
| AQOR | Ad hoc QoS On-demand Routing |
| BER | Bit-Error Rate |
| CBF | Channel Busy Fraction |
| CBT | Channel Busy Time |
| CPDF | Conditional Probability Distribution Function |
| CSMA/CA | Carrier-Sense Multiple Access with Collision Avoidance |
| DIFS | DCF Interframe Space |
| ETOP | Expected Number of Transmissions on a Path |
| ETX | Expected Transmission Count |
| EWMA | Exponential Weighted Moving Average |
| FIFO | First In First Out |
| HCF | Highest Common Factor |
| ISM | Industrial, Scientific and Medical |
| LABQ | Link Availability-Based QoS-aware |
| LCM | Least Common Multiple |
| MAC | Medium Access Control |
| ML | Minimum Loss |
| MLO | MAC Layer Overhead |
| MML | Medium Management Layer |
| MTX | Minimum Transmission Count |
| MUT | Medium Utilization Time |

| | |
|---|---|
| OB | Outlier Boundary |
| OB-EWMA | Outlier Bounded Exponential Weighted Moving Average |
| OR | Opportunistic Routing |
| PCI | Peripheral Component Interconnect |
| PDR | Packet Delivery Ratio |
| PLO | Physcial Layer Overhead |
| PSU | Power Supply Unit |
| QoS | Quality-of-routing |
| RSSI | Received Signal Strength Indication |
| SBC | Single Board Computer |
| SD | Standard Deviation |
| SINR | Signal-to-Interference-plus-Noise Ratio |
| SMTX | Smallest Maximum Number of Transmissions |
| SOC | System-on-Chip |
| SP | Shortest Path in hops |
| TDMA | Time-division multiple access |
| TTBA | Time Token Bucket Algorithm |
| TWMA | Time Weighted Moving Average |
| WMEWMA | Window Mean with Exponential Moving Average |
| WNCS | Wireless Networked Control Systems |

# 1 Introduction

## Contents


**1.1 Problem Statement** . . . . . **2**

**1.2 Approach** . . . . . **4**

**1.3 Contributions and the outline** . . . . . **6**

In the early 1800s, Michael Faraday conducted the iron-ring experiment to test his hypothesis that electricity and magnetism might be transmitted by waves resembling sound or light [FM14]. Following that, James Maxwell represented the findings of Faraday in mathematical representation (known as the Maxwell Faraday equation) and made significant contributions to the field of electromagnetic wave theory by explaining electromagnetic waves [FM14; Max73]. Later, Guglielmo Marconi demonstrated the use-cases of electromagnetic waves with the invention of the wireless telegraph system [Mar09; Rab16]. Subsequent developments in the paradigm of wireless technologies influenced the communication domain, resulting in wireless networks being used for multiple applications and the recent invention of cellular phones, satellite televisions, mobile Internet, or IoT devices. These revolutionary developments in wireless technologies significantly increased the number of consumer electronics that are equipped with wireless technologies for their operations. They use the wireless network to communicate between the devices of a network (intranet) and the Internet. Compared to wired networks, wireless networks are suitable for mobile applications, and other applications such as industrial automation. Further, the simplicity and quickness of deploying a wireless network without the hassle of a connected wired network infrastructure make it favorable for several applications.

One such domain that benefited from the developments in wireless networks is process automation in factories, also known as industrial automation. For example, the machines may need to communicate with each other to automate the production process, or to monitor the factory environment. In such use cases, using a wireless network as the backbone for communications rather than a wired network has several advantages, such as mobility or simplicity in deploying and maintaining the infrastructure. However, a major challenge when using wireless networks is communication reliability.

## 1.1 Problem Statement

One of the prominent wireless technologies specially designed for industrial automation is WirelessHart [Com+10]. It is a TDMA based communication technology that provides deterministic guarantees based on IEEE 802.15.4. In a WirelessHart network, nodes or devices are scheduled to access the medium at predefined time slots. Additionally, it uses the channel blacklisting and channel hopping mechanisms to avoid occupied channels. ZigBee [Erg04] is another technology based on IEEE 802.15.4 for industrial automation. ZigBee provides both contention-based and reservation-based medium accesses, where the contention-based period uses CS-

MA/CA and the reservation-based period uses TDMA medium access protocols (guaranteed time slots for each node).

As mentioned, the number of consumer electronics and wearables that use wireless technologies such as WiFi or Bluetooth and the number of WiFi access points is increasing. Further, it is possible to open or create WiFi Access Points (AP) using smartphones. Thus, using several smartphones, it is even possible to jam the 2.4 GHz ISM band, which is also shared by IEEE 802.15.4 [Sch+17]. Consequently, it is increasingly challenging to configure those channels used by the WirelessHART network from those used by WiFi access points without overlap so that they do not interfere with each other. Techniques such as channel hopping may mitigate this issue to a certain extent, which could improve statistical reliability. However, the deterministic behavior of the technology is not guaranteed.

Meanwhile, WiFi technology has evolved. In recent years, the communication speed of WiFi networks has increased significantly and become applicable for several applications, specifically for industrial applications. Theoretically, recent versions of WiFi (IEEE 802.11 ax) offer a speed of 9.6 Mbps [Kho+18]. Thus, it is possible to transmit at a substantially higher rate using WiFi than other technologies used to build industrial networks such as WirelessHart or ZigBee. However, reliability is one of the key challenges while developing a communication network using WiFi technology in contention mode. The communication reliability of a contention-based network, for instance, WiFi, depends on the number of devices competing for the shared medium, and the amount of traffic. I.e., unlike in a reservation-based network, a node must compete to access the medium. Further, it is also necessary to control and monitor the medium usage, as the medium is shared.

One common and frequent requirement for an industrial multi-hop wireless network is to ensure a specified minimum end-to-end reliability. Wireless devices or nodes, for instance, are attached to the machines in the production system to communicate via direct link or by forwarding the message through multiple nodes. I.e., nodes that are in direct communication range may communicate directly, whereas nodes that are far apart (without a direct communication link) forward their message as packets through a path or route (predetermined or on-demand) to the destination. A routing protocol determines and manages these routes, i.e., it manages the routing functionalities of a network based on suitable routing metrics.

Quality-of-Service (QoS) routing protocols determine the routes based on QoS requirements, such as maximum delay, minimum bandwidth, minimum energy consumption, or minimum reliability. For an industrial wireless control network, it is necessary to have routes that can guarantee statistical reliability required for the

successful operation of applications. However, wireless networks in contention mode do not satisfy this requirement. The challenge is to make contention-based wireless technologies such as WiFi fit to provide predictable statistical end-to-end reliabilities. Thus, developing an industrial wireless network operating in contention mode calls for a routing protocol that prioritizes reliability as the QoS requirement. Further, the reliability requirements vary for each application, and thus, the routing protocol must determine routes that guarantee the minimum statistical reliability required by each application.

Solutions based on IEEE 802.15.4, such as ProNet 4.0 [Got20], have benefited from specialized hardwares and customized MAC layers. However, this has created a dependency on hardware platforms. Similarly, it is possible to use specialized hardwares to create solutions using standard WiFi technology. For instance, IWLAN [Zhe+09] provides a solution using WiFi technology, where a Deterministic Scheduling Layer (based on time slot scheduling) is implemented on top of standard WiFi or modified WiFi to provide a deterministic communication mechanism. Using dedicated hardwares has several advantages instead of using off-the-shelf commodity WiFi hardwares. For instance, they provide further channel quality data, bit-level error details of a received WiFi frame (if there exists any error), the possibility of transmitting WiFi frames at specified transmission power, installing specialized antennas, or performing reservation-based medium access (for instance, in case of IWLAN). Thus, the developed protocols, metrics, and algorithms can exploit the extra knowledge and freedom from using dedicated or specialized hardware. However, the protocol's design may depend on the underlying hardware, and thus, the network operations may depend on the hardware functionalities. If the vendor that produces the hardware halts the production of the dedicated hardware, then it may be challenging to port the functionalities designed based on it, or even it is not possible. Thus, hardware-dependent solutions reduce flexibility, and a network has to rely on specific hardwares.

## 1.2 Approach

An alternative approach is to construct a solution that can also be deployed using off-the-shelf hardware such as WiFi operating in contention mode for wireless controlled production systems. Considering the challenges and requirements mentioned earlier, and observing that in practice, only TDMA-based communication systems provide predictable reliability, the approach proposed in this research is to achieve comparable statistical reliability based on contention-based communica-

tion systems. Although off-the-shelf WiFi hardwares are low-cost and developing hardware-independent protocols [1] reduces the dependency on certain hardware, it is challenging to develop it. I.e., the protocols and algorithms used for the functioning of the network must rely solely on the information provided by the off-the-shelf hardware. However, the main challenge is determining and maintaining reliable end-to-end communication routes, as route reliability is vital for industrial applications. Therefore, the underlying routing protocol must prioritize reliability as the performance criterion for route selection. At the same time, it is also essential to minimize the bandwidth usage by the routes since WiFi operates in license-free channels. Hence, for wireless networks in industrial applications using off-the-shelf WiFi in contention mode, it is necessary to operate routes with minimum bandwidth usage that guarantees the specified minimum statistical reliability.

Routing protocols perform route selection using different routing metrics, which requires multiple information regarding different aspects of the network, such as network topology, link quality, or available bandwidth, to determine and maintain the routes. In WiFi, the nodes pursue a greedy approach to access the medium when new data is available to transmit, i.e., they immediately attempt to transmit. As the network density increases, the competition for the medium increases and may deteriorate the network quality. Thus, it is necessary to monitor the medium condition and control the medium access (shape the outgoing traffic) by the nodes to maintain the quality of the network. Together with developing a routing protocol emphasizing the reliability requirements, it is vital to develop protocols to collect the additional network status data required by the routing protocols, i.e., protocols and metrics to disseminate and quantify the link quality across the network and monitor the medium conditions.

Another critical factor that affects network performance is the weakness in the statistical link properties required for the functioning of protocols. For a network operating in a license-free channel (ISM band), it is challenging to guarantee the existence of certain statistical properties as external devices (devices not part of the network) can easily influence the characteristics of the channel. However, it is possible to monitor and validate the strength of the required statistical link properties and control the activities of the nodes in the network with the motivation to maintain those statistical properties. Therefore, instead of assuming the existence of the required statistical link properties, it is necessary to reckon the strength of those

[1] Protocols whose functionalities solely rely on the information provided by off-the-shelf WiFi hardware without any modifications to the WiFi protocol and without using any specialized hardwares.

properties in practice and take necessary measures to strengthen the establishment of the required properties.

## 1.3 Contributions and the outline

In the research undertaken for this doctoral thesis design, implement, and test multiple protocols and algorithms for reliability-constrained routing in contention-based wireless networks, with emphasis on WiFi. As the starting point, we investigated the state-of-art routing mechanisms that could fit the requirements, i.e., a reliability-constrained routing methodology that can guarantee a specified minimum end-to-end reliability which can also function on off-the-shelf IEEE 802.11 (WiFi) networks in contention mode. In this study, we show that existing approaches fall short of our objective. Therefore, we develop and present a novel routing methodology. However, the success of the proposed methodology depends on certain statistical properties of the network, and information for route calculation, such as link quality, and this directs to the following questions in general:

1. Does the network maintain the statistical properties for the successful operation of the proposed routing methodology?

2. Are the prerequisites affected during operation? If so, how to monitor any divergences?

3. How do the traffic characteristics of a node affect the network and statistical link properties? Is it possible to shape the outgoing traffic of each node to maintain the requirements?

4. Do the existing algorithms and metrics fit the use case, such as link quality estimation?

We focus on a specific research problem in each chapter. The solutions proposed in the thesis have been implemented on top of off-the-shelf commodity WiFi hardware. The following are the contributions of this thesis:

- Chapter 2 presents a novel routing approach for reliability-constrained QoS routing with specified statistical minimum route reliability in IEEE 802.11 (WiFi) based medium-sized commercial networks operating in contention mode. The proposed routing protocol uses a novel routing metric to determine bandwidth-efficient routes that can provide a specified statistical minimum reliability. For the successful operation of the routes, we assume that the

network and the links satisfy certain statistical properties. Further, to determine routes, the proposed protocol assumes the availability of status data about network topology and the quality of the links. These assumptions are addressed and lifted in the following chapters of the theses.

- The quality of the determined routes depends on the quality of the individual links in the route. If the transmission outcomes on links have certain statistical properties, the link quality is increased or boosted by retransmitting failed transmissions. In Chapters 3 and 4, we explore metrics to evaluate and examine different statistical link properties required for successfully boosting the link quality through retransmissions, which is necessary for the successful operation of the routes. Further, we examine the validity of the statistical properties in practice.

- In a WiFi network operating in contention mode, nodes pursue a greedy approach to access the medium. Unsupervised medium access could lead to traffic bursts, which may affect the link quality and, subsequently, the routes. Chapter 5 presents an algorithm for traffic shaping and sharing the medium occupancy time among nodes depending on the applications' requirements. Further, a hardware-independent metric is derived from the proposed mechanism to detect medium overload locally. To understand the effects of unsupervised medium access, we evaluate the impact of traffic patterns generated by the nodes on the statistical link properties. Furthermore, we evaluate the influence of the proposed traffic shaping mechanism to mitigate the impact.

- Another important aspect that determines the quality of the routes is the link metric used to estimate the link quality. For instance, if the estimated link qualities are higher than the actual ones, or if the link metric did not adapt to the changes in the link quality, it could lead to route failures, i.e., routes may fail to meet the specified route reliability. Further, if the link quality estimates vary frequently or are unstable, it could lead to frequent route recalculations. Links of a WiFi network operating in contention mode and, moreover, in a license-free channel, are susceptible to changes during the operation of the network. Thus, the link metric must account for multiple aspects when calculating or estimating the link quality. Chapter 6 presents a link metric suitable for reliability-constrained routing that determines adaptive, conservative, and stable link quality estimates.

- Finally, Chapter 7 concludes the findings and presents future works that could further expand on the presented solutions.

# 2

# $r_{min}$-Routing :- A Protocol for Statistically Reliable Routing for Contention-based Wireless Networked Control Systems

## Contents


**2.1 Introduction** . . . . . **10**
**2.2 $r_{min}$-routing** . . . . . **10**
2.2.1 State of the art . . . . . 11
2.2.2 Motivation . . . . . 12
2.2.3 $r_{min}$ routing metric . . . . . 15
2.2.4 Route Selection . . . . . 20
2.2.5 Experiments . . . . . 23
2.2.6 Conclusion . . . . . 31
**2.3 Further contributions** . . . . . **32**
**2.4 Implementation framework and the hardwares** . . . . . **33**
2.4.1 WiPS Framework . . . . . 34
2.4.2 Hardwares used in evaluations . . . . . 35
**2.5 Summary** . . . . . **37**

## 2.1 Introduction

Reliability is one of the key challenges while developing a communication network using off-the-shelf commodity hardwares in contention mode, such as IEEE 802.11 (WiFi). IEEE 802.11 or WiFi operates in a license-free band (ISM band), where multiple networks may operate. In contention mode, the devices in the network compete with one another to access the medium. Thus, the communication reliability between devices in such a network depends on multiple factors. Communication between devices or nodes in a multi-hop wireless network is determined by the routing protocol used by the network. Quality-of-Service (QoS) routing protocols are developed by prioritizing a specific domain (or domains), for instance, latency, as the criteria for path selection. This chapter presents a novel QoS routing approach called $r_{min}$-routing, prioritizing path reliability, which operates $r_{min}$-routes with specified minimal reliability in off-the-shelf IEEE 802.11 (WiFi) mid-sized multi-hop networks in contention mode.

The chapter is structured as follows: Sec. 2.2 presents the routing protocol prioritizing reliability-constrained routes in wireless networks. Sec. 2.3 presents the research questions addressed in the remainder of this thesis, which synergizes the probability of successful operation of the proposed routing methodology in practice. Sec. 2.4 introduces the framework and hardwares used to implement and evaluate the protocols and algorithms developed in this thesis. Finally, Sec. 2.5 summarizes the chapter.

## 2.2 $r_{min}$-routing

This section introduces $r_{min}$-routing [KMG19a], which determines bandwidth-efficient routes with statistical minimum reliability $r_{min}$. $r_{min}$-routing selects a feasible route that uses an optimal minimum number of transmissions to achieve the target reliability. For instance, a route selection with an objective of $r_{min} = 0.9$ requests for routes with a statistical success probability of at least 90% for the packets to reach their destinations. In Sec. 2.2.1, we discuss the state of the art of routing protocols for wireless networks. Sec. 2.2.2 presents the network model and the motivation behind the development of $r_{min}$-routing. Then, Sec. 2.2.3 presents the routing metric to determine the minimum number of transmissions and the optimal distribution of transmissions along the path to satisfy the target path reliability $r_{min}$. Following that, in Sec. 2.2.4, we present the route selection steps.

Finally, in Sec. 2.2.5, we evaluate the performance of $r_{min}$-routing and compare it to existing state-of-the-art approaches.

### 2.2.1 State of the art

Many routing protocols exist for wireless networks, and this section discusses various state-of-the-art routing algorithms. AODV [PR99], DSDV [PB94], and Dynamic Source Routing (DSR) [JM96] are the classical routing algorithms for wireless networks. Ad-hoc QoS On-demand Routing (AQOR) [XG03], QOLSR [MF07], and Link Availability-Based QoS-aware (LABQ) [Yu+07] are some of the earliest routing protocols developed for QoS routing. AQOR focuses on selecting and operating routes with low end-to-end latency and minimal bandwidth usage. However, AQOR assumes that all nodes use the same data rate, and the transmission range between nodes is identical. QOLSR is a variant of the original OLSR protocol redesigned for QoS routing. Like AQOR , QOLSR also focuses on end-to-end delay as a constraint for route selection. LABQ is designed considering the mobility of wireless nodes. During route selection, LABQ prioritizes links that are predicted to have high availability. Further, it uses the well known metric Expected Transmission Count (ETX) [De +03] as the link quality metric. However, in the routing protocols mentioned above, there are no measures to guarantee statistical end-to-end target reliability.

ReInForm [DBN03], SPEED [He+03], and MMSPEED [FLE06] are routing protocols that consider the end-to-end reliability together with other QoS requirements. ReInForm is a multi-path route protocol where each packet from the source to the sink is delivered through multiple paths. The protocol is developed for single sink network models and collision-free TDMA based networks. SPEED [He+03] and MMSPEED [FLE06] are location-aware routing protocols. SPEED is designed for end-to-end soft real-time communication. Intermediate or relay nodes in a route are selected depending on the geographic distance between nodes and the destination node. MMSPEED is a variant of the SPEED routing protocol. Compared to SPEED, MMSPEED focuses on improving the reliability and timeliness further by multi-path routing. Location-aware protocols are highly scalable, and have no overhead for updating the link quality and disseminating the topological information among the nodes. However, during route selection, they assume that the link quality in subsequent hops is identical to the present hop [FLE06], which is unrealistic.

Opportunistic Routing (OR) Algorithms [BM04; JS16; Kim+19] are another class of QoS routing algorithms where the relay node is non-deterministically selected on runtime. For instance, Fig. 2.1 shows multiple paths between nodes $v_1$ and $v_4$. Assume node $v_1$ wants to forward a frame to $v_4$. The link quality from node $v_1$ to $v_4$

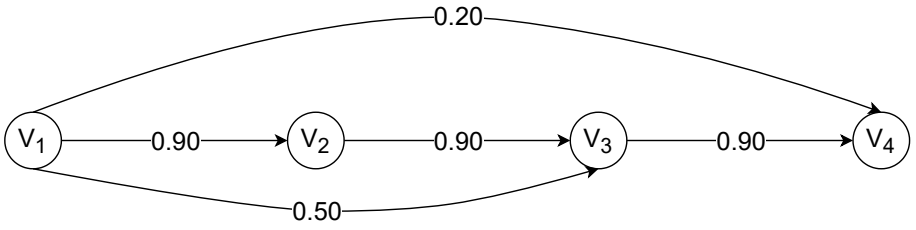



Figure 2.1: Multiple paths between node $v_1$ and $v_4$

is less reliable than the links to nodes $v_2$ and $v_3$. In OR routing, node $v_1$ multicasts the frame destined to node $v_4$ and delays the acknowledgements accordingly. In this sample topology, node $v_4$ has the earliest opportunity (shortest delay) to send an acknowledgement, followed by nodes $v_3$ and $v_2$. If the node $v_4$ successfully receives the frame, it acknowledges with an acknowledgement frame. Nodes $v_3$ and $v_2$ overhear this message, and cancel their acknowledgements (even though they received the frame) as the frame travelled further. Otherwise, if the frame is not successfully delivered to node $v_4$ but to node $v_3$ and $v_2$, then node $v_3$ sends the earliest acknowledgement. In this manner, the OR utilizes overhearing by the neighbouring nodes to move the frame closer to the destination. A major drawback to this approach is that modifications are required to the existing standard 802.11 MAC protocol to implement OR Routing Algorithms.

### 2.2.2 Motivation

Route discovery in $r_{min}$-routing is based on a network model where the wireless network is modelled as a directed graph. We model the network as a graph $G$ consisting of a set of nodes $V$ and a statistical single-hop link quality function $q : V \times V \to \mathbb{R}_{[0,1]}$. For instance, $q(v, v')$ may denote the probability for successful single-hop delivery of a frame from node $v$ to $v'$. This implicitly defines the set of links $E =_{def} \{(v, v') \in V \times V \mid q(v, v') > 0\}$, with $q^e$ denoting the link quality for link $e \in E$. $P \subset E$ denotes the set of all cycle-free paths in the network $G$. Further, the transmission event outcomes on a link and success probabilities of transmission events between links are assumed as statistically independent.

Assuming that the transmission outcomes on a link are statistically independent, retransmissions are applied to failed transmissions to increase or boost the link

quality. The link quality [2] of up to $n^e_{max}$ transmissions of a frame on link $e$ is increased to $q^e_{n^e_{max}}$,

$$q^e_{n^e_{max}} = 1 - (1 - q^e)^{n^e_{max}} \tag{2.1}$$

Further, the path reliability for a given path $p \in P$ is the product of link quality of links in the path. I.e.,

$$q^p = \prod_{e \in P} q^e \tag{2.2}$$

The path reliability can be increased or boosted by increasing or boosting the qualities of links in the path. I.e., for a path $p \in P$ and with a maximum of $n^e_{max}$ transmissions on link $e \in P$, we obtain a path reliability $q^p_{n^p_{max}}$ and a maximum number of retransmissions on the path $n^p_{max}$,

$$n^p_{max} = \sum_{e \in P} n^e_{max} \tag{2.3}$$

$$q^p_{n^p_{max}} = \prod_{e \in P} q^e_{n^e_{max}} \tag{2.4}$$

Consider the 2-hop path depicted in Fig. 2.2. The link qualities of links in path $p = \langle e_1, e_2 \rangle$ is $q^{e_1}_1 = q^{e_2}_1 = 0.80$. Applying Eqn. 2.2, the path reliability $q^p$ for path $p$ from node $v_1$ to $v_3$ without any retransmissions is 0.64. If the specified statistical minimum path reliability $r_{min}$ is 0.90, retransmissions are required. Permitting one retransmission per link, i.e., $n^e_{max} = 2$, boosts link qualities of the intermediate links in the path $p$ to $q^{e_i}_2 = 0.96$ (applying Eqn. 2.1 and then, Eqn. 2.2). Thus, the path reliability of $p = \langle e_1, e_2 \rangle$ is boosted to $q^p_4 = 0.9216$ with $n^p_{max} = 4$ (a maximum of 4 transmissions on the path).

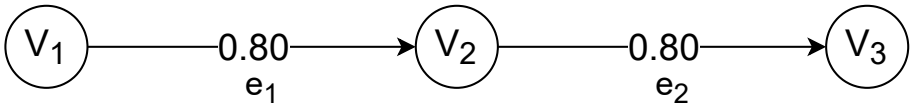



Figure 2.2: A sample route

Consider that the specified statistical minimum path reliability $r_{min}$ is increased from 0.90 to 0.95. It can satisfy the specified target reliability by permitting an extra retransmission on any link, i.e., $n^e_{max}$ to 3. Permitting three transmissions on either link $e_1$ or $e_2$ boosts the path reliability from 0.80 to 0.992. Thus, with two transmissions on one link and three on another, the link qualities are boosted to

[2] In practice, the link quality $q^e$ is calculated from link reliability or the Packet Delivery Ratio (PDR) and considers the statistical characteristics of historical PDR values.

0.96 and 0.992, respectively. This results in a boosted path reliability $q_5^p$ of 0.952 with $n_{max}^p$ of 5.

In IEEE 802.11, the number of transmission attempts is set to a default value $n_{max}$ between 4 and 7 [3], independent of the link quality [IEE]. For the sample 2-hop path depicted in Fig. 2.2, permitting four transmissions (the default value) on a link is sufficient to satisfy the statistical minimum target reliability $r_{min} \leq 0.996$ on path $p = \langle e_1, e_2 \rangle$. However, for a link quality of $q_1^{e_1} = q_1^{e_2} = 0.80$, $n_{max}^e = 2$ is sufficient to satisfy the statistical minimum target reliability $r_{min}$ of 0.90. I.e., the additional attempts are not required. By restricting the number of transmissions on a route depending on the target reliability, it is possible to save the bandwidth rather than using a default number of transmission attempts.

Now, consider the link qualities of links $e_1$ and $e_2$ are $q_1^{e_1} = q_1^{e_2} = 0.50$. Then, by assigning a default value of four transmissions per link, the path reliability $q^p$ is raised only up to 0.87. If the specified statistical minimum target path reliability $r_{min}$ is 0.90, increasing the default number of transmissions on either one link $e_1$ or $e_2$ (from 4 to 5) increases the path reliability $q^p$ to 0.908. I.e., it is sufficient to increase the transmission attempts at one link rather than increasing the default number of transmissions on both links. The latter approach, increasing the default number of transmissions on both links to increase the path reliability, reduces the network's usable throughput. Hence, it is necessary to carefully select the upper bound for the number of transmissions per link. Depending upon these aspects, it is not efficient to use the maximum default transmission attempts to achieve a reliability target between nodes. Limiting the transmissions based on route requirements creates space for more routes and creates a (soft) upper limit for the runtime latency (not considering the back-off due to medium congestion).

Further, instead of using a default number of transmissions, the resource (transmission attempts) can be redistributed to weaker links in a route and increase the route's reliability. It could transform a weaker link into a feasible link for routing. Thus, instead of using the same default number of transmissions on all links, dynamically distributing the resource depending on the route requirements saves the bandwidth, increase the route's reliability, and creates space for more routes.

$r_{min}$-routing calculates a *feasible* and *optimal* path $p$, if there exists. A path $p$ is *feasible* if it can match the specified target reliability $r_{min}$; otherwise, it is unfeasible. $r_{min}$-routing calculates the sufficient minimum number of transmissions $n_{max}^e$ required on each link $e \in p$ such that $q_{n_{max}^p}^p \geq r_{min}$. A feasible path $p$ is optimal, if there is no other feasible path $p'$ with $n_{max}^{p'} < n_{max}^p$. Using optimal paths

[3] The default value is 4, and it is possible to increase up to 7 transmissions.

ensures that the bandwidth usage by $r_{min}$ routes is minimized. Hence, compared to using a fixed default number of transmissions $n_{max}$ on all links, this flexible approach consumes less bandwidth and yields the potential for more routes.

These criteria are exemplified in the examples mentioned earlier using the path Fig. 2.2, i.e.,

- Feasibility: For link qualities $q_1^{e_1} = q_1^{e_2} = 0.50$ and $r_{min} = 0.9$, a *feasible* path $p$ is calculated by choosing $n_{max}^{e_1} = 4$, and $n_{max}^{e_2} = 5$, not by $n_{max} = 4$.
- Optimality: For link qualities $q_1^{e_1} = q_1^{e_2} = 0.80$ and $r_{min} = 0.9$, it is sufficient to choose a maximum number of transmissions $n_{max}^{e} = 2$ on both links instead of $n_{max} = 4$, thus saving the bandwidth.

### 2.2.3 $r_{min}$ routing metric

$r_{min}$-routing relies on the SMTX (Smallest Maximum Number of Transmissions) metric evolved from MTX (Minimum Transmission Count) metric. For a feasible path $p$, SMTX calculates the smallest number of transmissions that is needed to guarantee the specified statistical reliability target $r_{min}$. The algorithm to compute SMTX and proof of optimality is presented in [KMG19a]. It is not further discussed as it is not the scope of this thesis. For completeness, we briefly explain the calculation of SMTX using an example and the improvements compared to MTX.

The earliest version of $r_{min}$-routing used MTX to allocate the transmission budget or calculate the number of transmissions required on each link $e \in p$ to satisfy the specified target reliability $r_{min}$. MTX works as follows:

1. Calculate the path reliability $q^p_{n^p_{max}}$ for $n^e_{max} = 1$ or without any retransmissions.
2. If the path reliability is less than the specified target reliability, i.e., $q^p_{n^p_{max}} < r_{min}$, select the weakest link or link with the smallest reliability $e_i \in p$ with $n^{e_i}_{max}$ transmissions, and permit an additional transmission on link $e_i$.
3. Repeat step 2 until the boosted path reliability $q^p_{n^p_{max}}$ is higher than the specified target reliability, i.e., $q^p_{n^p_{max}} \geq r_{min}$, or terminate if $n^p_{max}$ exceeds a predefined average transmission budget on paths $n_{maxAvg}$.

Consider the path $p$ depicted in Fig. 2.3. Without any retransmissions, i.e., $n^e_{max} = 1$ or $n^p_{max} = 3$, the path reliability is $q^p_3 = 0.336$. Starting from $n^e_{max} = 1$ on each link $e \in p$, on each iteration, the algorithm allocates an extra transmission to the weakest link. The algorithm continues until the path reliability exceeds the

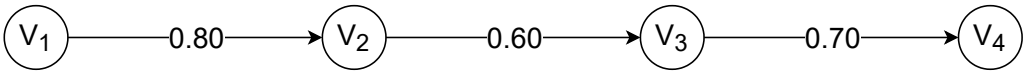



Figure 2.3: A 3-hop path $p = \langle e_1, e_2, e_3 \rangle$

specified target reliability, i.e., $q^p_{n^p_{max}} \geq r_{min}$. Tab. 2.1 shows the transmission allocation after each iteration for path $p$. Consider the specified target reliability $r_{min} = 0.80$. The bold cell denotes the weakest link $e_i \in p$ selected to allocate an additional transmission in the forthcoming iteration. On iteration 1, the algorithm checks whether the specified target reliability is satisfied without any additional transmissions. Since specified target reliability $r_{min}$ is not satisfied, an additional transmission on link $e_2$ (i.e., $n^{e_2}_{max} = 2$) is allocated in iteration 2, raising the path reliability to $q^p_4 = 0.470$. The grey coloured cell shows the boosted link quality with an additional transmission compared to the previous iteration. As mentioned earlier, the bold cell denotes the weakest link selected after each iteration. The iteration continues until the path reliability exceeds the specified target reliability $r_{min}$.

| Iteration | $q^{e_1}_{n^{e_1}_{max}}$ | $q^{e_2}_{n^{e_2}_{max}}$ | $q^{e_3}_{n^{e_3}_{max}}$ | $q^p_{n^p_{max}}$ |
|---|---|---|---|---|
| 1 | $q^{e_1}_1 = 0.80$ | $\boldsymbol{q^{e_2}_1 = 0.60}$ | $q^{e_3}_1 = 0.70$ | $q^p_3 = 0.336$ |
| 2 | $q^{e_1}_1 = 0.80$ | $q^{e_2}_2 = 0.84$ | $\boldsymbol{q^{e_3}_1 = 0.70}$ | $q^p_4 = 0.470$ |
| 3 | $\boldsymbol{q^{e_1}_1 = 0.80}$ | $q^{e_2}_2 = 0.84$ | $q^{e_3}_2 = 0.91$ | $q^p_5 = 0.611$ |
| 4 | $q^{e_1}_2 = 0.96$ | $\boldsymbol{q^{e_2}_2 = 0.84}$ | $q^{e_3}_2 = 0.91$ | $q^p_6 = 0.733$ |
| 5 | $q^{e_1}_2 = 0.96$ | $q^{e_2}_3 = 0.939$ | $\boldsymbol{q^{e_3}_2 = 0.91}$ | $q^p_7 = 0.817$ |

Table 2.1: Transmission allocation on each link $e \in p$ after each iteration by MTX to meet the specified target reliability $r_{min} = 0.80$

However, the transmission allocation returned by MTX is not optimal. As mentioned earlier, in [KMG19a], SMTX is introduced to calculate the smallest number of $n^p_{max}$ to achieve the target reliability and, thus, optimality.

$$SMTX^p_{r_{min}} = min\left\{n^p_{max} \in \mathbb{N} \mid q^p_{n^p_{max}} \geq r_{min}\right\} \tag{2.5}$$

Compared to MTX, SMTX allocates the additional transmission to a link $e \in p$, where this allocation results in the highest proportional increase of quality in each iteration, i.e., the permitted additional transmission yields the highest gain. The proportional increase $Pinc(k, q^e)$ in link quality $q^e$, when an additional transmission to link $e$ that has $k$ transmissions allocated is denoted as

$$Pinc(k, q^e) = \frac{q^e_{k+1}}{q^e_k} \tag{2.6}$$

| Iteration | Metric | $q^{e_1}_{n^{e_1}_{max}}$ | $q^{e_2}_{n^{e_2}_{max}}$ | $q^{e_3}_{n^{e_3}_{max}}$ | $q^{p}_{n^{p}_{max}}$ |
|---|---|---|---|---|---|
| 1 | MTX | $q^{e_1}_1 = 0.80$ | $\boldsymbol{q^{e_2}_1 = 0.60}$ | $q^{e_3}_1 = 0.70$ | $q^{p}_3 = 0.336$ |
| | SMTX | $q^{e_1}_1 = 0.80$ | $q^{e_2}_1 = 0.60$ | $q^{e_3}_1 = 0.70$ | $q^{p}_3 = 0.336$ |
| | $Pinc(k, q^e)$ | 1.2 | **1.4** | 1.3 | |
| 2 | MTX | $q^{e_1}_1 = 0.80$ | $q^{e_2}_1 = 0.84$ | $\boldsymbol{q^{e_3}_1 = 0.70}$ | $q^{p}_4 = 0.470$ |
| | SMTX | $q^{e_1}_1 = 0.80$ | $q^{e_2}_1 = 0.84$ | $q^{e_3}_1 = 0.70$ | $q^{p}_4 = 0.470$ |
| | $Pinc(k, q^e)$ | 1.2 | 1.114 | **1.3** | |
| 3 | MTX | $\boldsymbol{q^{e_1}_1 = 0.80}$ | $q^{e_2}_2 = 0.84$ | $q^{e_3}_2 = 0.91$ | $q^{p}_5 = 0.611$ |
| | SMTX | $q^{e_1}_1 = 0.80$ | $q^{e_2}_2 = 0.84$ | $q^{e_3}_2 = 0.91$ | $q^{p}_5 = 0.611$ |
| | $Pinc(k, q^e)$ | **1.2** | 1.114 | 1.069 | |
| 4 | MTX | $q^{e_1}_2 = 0.96$ | $\boldsymbol{q^{e_2}_2 = 0.84}$ | $q^{e_3}_2 = 0.91$ | $q^{p}_6 = 0.733$ |
| | SMTX | $q^{e_1}_2 = 0.96$ | $q^{e_2}_2 = 0.84$ | $q^{e_3}_2 = 0.91$ | $q^{p}_6 = 0.733$ |
| | $Pinc(k, q^e)$ | 1.033 | **1.114** | 1.069 | |
| 5 | MTX | $q^{e_1}_2 = 0.96$ | $q^{e_2}_3 = 0.939$ | $\boldsymbol{q^{e_3}_2 = 0.91}$ | $q^{p}_7 = 0.817$ |
| | SMTX | $q^{e_1}_2 = 0.96$ | $q^{e_2}_3 = 0.939$ | $q^{e_3}_2 = 0.91$ | $q^{p}_7 = 0.817$ |
| | $Pinc(k, q^e)$ | 1.033 | 1.041 | **1.069** | |
| 6 | MTX | $q^{e_1}_2 = 0.96$ | $\boldsymbol{q^{e_2}_3 = 0.939}$ | $q^{e_3}_3 = 0.973$ | $q^{p}_8 = 0.874$ |
| | SMTX | $q^{e_1}_2 = 0.96$ | $q^{e_2}_3 = 0.936$ | $q^{e_3}_3 = 0.973$ | $q^{p}_8 = 0.874$ |
| | $Pinc(k, q^e)$ | 1.033 | **1.041** | 1.019 | |
| 7 | MTX | $\boldsymbol{q^{e_1}_2 = 0.96}$ | $q^{e_2}_4 = 0.974$ | $q^{e_3}_3 = 0.973$ | $q^{p}_9 = 0.91$ |
| | SMTX | $q^{e_1}_2 = 0.96$ | $q^{e_2}_4 = 0.974$ | $q^{e_3}_3 = 0.973$ | $q^{p}_9 = 0.903$ |
| | $Pinc(k, q^e)$ | **1.033** | 1.0157 | 1.019 | |
| 8 | MTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_4 = 0.974$ | $\boldsymbol{q^{e_3}_3 = 0.973}$ | $q^{p}_{10} = 0.94$ |
| | SMTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_4 = 0.974$ | $q^{e_3}_3 = 0.973$ | $q^{p}_{10} = 0.940$ |
| | $Pinc(k, q^e)$ | 1.0064 | 1.0157 | **1.019** | |
| 9 | MTX | $q^{e_1}_3 = 0.992$ | $\boldsymbol{q^{e_2}_4 = 0.974}$ | $q^{e_3}_4 = 0.991$ | $q^{p}_{11} = 0.958$ |
| | SMTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_4 = 0.974$ | $q^{e_3}_4 = 0.991$ | $q^{p}_{11} = 0.958$ |
| | $Pinc(k, q^e)$ | 1.0064 | **1.0157** | 1.005 | |
| 10 | MTX | $q^{e_1}_3 = 0.992$ | $\boldsymbol{q^{e_2}_5 = 0.989}$ | $q^{e_3}_4 = 0.991$ | $q^{p}_{12} = 0.973$ |
| | SMTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_5 = 0.989$ | $q^{e_3}_4 = 0.991$ | $q^{p}_{12} = 0.973$ |
| | $Pinc(k, q^e)$ | **1.0064** | 1.0062 | 1.0057 | |
| 11 | MTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_6 = 0.995$ | $\boldsymbol{q^{e_3}_4 = 0.991}$ | $q^{p}_{13} = 0.979$ |
| | SMTX | $q^{e_1}_4 = 0.998$ | $q^{e_2}_5 = 0.989$ | $q^{e_3}_4 = 0.991$ | $q^{p}_{13} = 0.981$ |
| 12 | MTX | $q^{e_1}_3 = 0.992$ | $q^{e_2}_6 = 0.995$ | $q^{e_3}_4 = 0.997$ | $q^{p}_{14} = 0.985$ |

Table 2.2: Transmission allocation on each link $e \in p$ after each iteration by MTX and SMTX to meet the specified target reliability $r_{min} = 0.98$

In each iteration, SMTX allocates an additional transmission to the link $e \in p$ with the highest value for $Pinc(k, q^e)$.

For instance, Tab. 2.2 shows the iteration results to meet a target reliability $r_{min} = 0.98$ on the same path $p$ depicted in Fig. 2.3. In iteration 1, MTX selects link $e_2$ since it is the weakest link to allocate an extra transmission in iteration 2. In comparison, SMTX also selects link $e_2$ since it has the highest proportional increase with an additional allocation. I.e., applying Eqn. 2.2.3 to each link $e \in p$,

$$e_1, \quad Pinc(2, 0.8) = \frac{0.96}{0.80} = 1.2$$

$$e_2, \quad Pinc(2, 0.6) = \frac{0.84}{0.60} = 1.4$$

$$e_3, \quad Pinc(2, 0.7) = \frac{0.91}{0.70} = 1.3$$

In this example, until iteration 10, MTX and SMTX select the same link to allocate an additional transmission to satisfy the target reliability. However, in iteration 11, with $n^p_{max} = 13$, SMTX found the optimal allocation of the transmission budget on each link $e \in p$ to meet the specified target reliability $r_{min}$, whereas MTX did not. It took MTX an additional iteration or transmission to find the budget allocation to meet the target path reliability.

Next, we compare SMTX with other potential routing metric candidates for reliability-constrained routing. The selection pool is limited since we only consider the implementable metrics on top of commodity (off-the-shelf) WiFi hardware. The routing metric SP [Dua+09] calculates the shortest path between source $v_s$ and destination $v_d$.

$$SP_{v_s,v_d} = min\left\{|p| \mid p \in P \wedge \text{first}(p) = v_s \wedge \text{last}(p) = v_p\right\} \tag{2.7}$$

where, the functions **first(p)** and **last(p)** return the first and last nodes of path $p$, respectively. The metric SP is more applicable to wired networks. Since SP selects the shortest path, a minimal number of nodes are involved, which may reduce the end-to-end latency and bandwidth usage. However, this depends on the link qualities (influenced by the topology), which are not considered in the route selection (whereas SMTX does). Therefore, SP usually selects paths containing weak and geographically far-away links in wireless networks, thus determining paths with poor reliability.

The routing metric ETX [De +03] is a popular choice, and it was the first metric devised specifically for reliability-constrained routing in wireless sensor networks [Cam+08; Bau+07]. ETX calculates the average number of transmissions required to deliver a packet over a path. ETX of a link, $ETX^e$ is the average number of

transmissions required to send the packet on link $e$, and ETX of a path, $ETX^p$ is the sum of ETX values for the links in path $p$.

$$ETX^p = \sum_{e \in p} ETX^e \tag{2.8}$$

where (2.9)

$$ETX^e = \frac{1}{q^e} \tag{2.10}$$

Like SMTX, ETX also considers the link qualities for selecting the path. ETX selects the path with the minimum value ($ETX^p$), thus minimizing the average bandwidth consumption, i.e., path $p_{minETX}$ is the path selected by applying ETX from a set of paths $P$,

$$\begin{aligned} p_{minETX} &= \arg\min_{p \in P} \ ETX(p) \\ &\text{where,} \\ ETX(p) &= ETX^p \end{aligned} \tag{2.11}$$

However, compared to SMTX, ETX does not consider specified target reliability. Furthermore, an upper bound ($n^e_{max}$) to the number of transmissions is not permitted on links. For instance, if the maximum number of transmissions on a link is $n^e_{max} = 3$, then a link with reliability $q^e = 0.25$ would require 4 ($1/q^e$) average transmissions. Considering this drawback, in [Jak+11], the authors introduced a new routing metric, Expected Number of Transmissions on a Path (ETOP), that considers an upper bound to the maximum number of transmissions per link and, notably, the position of the links. However, ETOP also does not consider a target path reliability. Nevertheless, since ETOP considers the links' positions in a route, it is a potential candidate to filter paths (during the selection of $r_{min}$-routes) with similar SMTX values. ETOP for a path $p$ with length $n$ is calculated [Jak+11],

$$\begin{aligned} ETOP^p_n = &\sum_{i=1}^{n-2} ETX^{e_i} \cdot (\rho_{i+1}/\rho_n) + \\ &n^{e_i}_{max} \cdot (1 - \rho_n)/(\rho_n) + \\ &ETX^{e_{n-1}} \end{aligned} \tag{2.12}$$

where,

$$\rho_h = \prod_{i=0}^{h} q^{e_i}_{n^{e_i}_{max}}$$

where, $\rho_n$ calculates the success probability that the packet is successfully delivered until node $v_n$ from the source node by permitting $n^e_{max}$ transmissions per link.

Finally, we also consider the routing metric ML [Pas+06] that selects a path that can achieve the highest end-to-end reliability between the source and the destination with a single transmission on the links.

$$ML^p = q^p = \prod_{e \in p} q^e. \tag{2.13}$$

Path $p_{maxML}$ is selected by applying the metric ML is the path with highest end-to-end reliability, i.e.,

$$\begin{aligned} p_{maxML} &= \arg\max_{p \in P} \ ML(p) \\ &\text{where,} \\ ML(p) &= ML^p \end{aligned} \tag{2.14}$$

ML is a simple and straightforward metric, and we considered it to compare it with SMTX that considers a specific reliability target.

### 2.2.4 Route Selection

Bandwidth-efficient $r_{min}$-routes are calculated by successively applying multiple routing metrics. The process involves multiple filtering stages applying routing metrics such as SMTX, ETX, and ETOP. We present the recent version of $r_{min}$ route selection that consists of five levels of filtering. Compared to the initial version [Koh18], improvements were introduced in [KMG19a], especially further filtering steps 4 and 5.

The route selection process starts by creating a candidate set $P_{C_1}$, a subset of the set $P$ containing all possible paths between the source and the destination node. The number of possible paths between two nodes increases as the network size increases, and listing all possible paths in a growing network is an NP-complete problem. Therefore, it is not feasible and computationally inefficient to consider all possible paths between two nodes. The route selection process begins by applying Yen's algorithm[Yen70], where the candidate set $P_{C_1}$ that consists of the $k$ shortest paths with respect to ETX is selected.

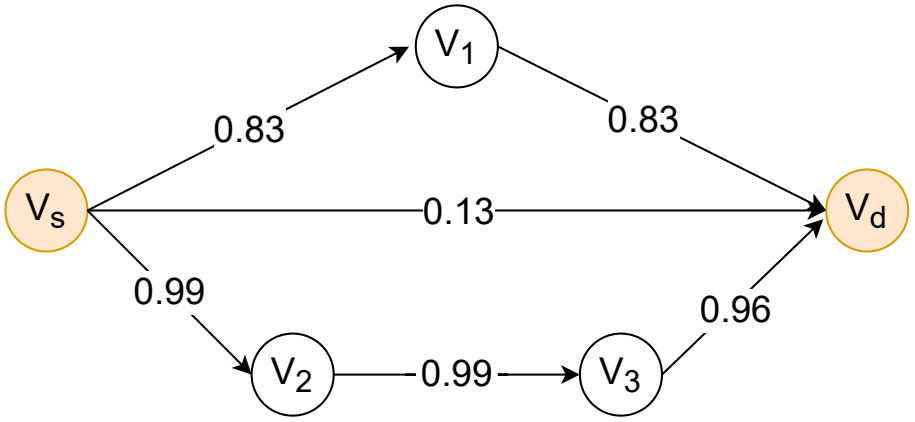



Figure 2.4: Multiple paths between $v_s$ and destination node $v_d$.

| Path | $SP$ | $ML$ | $ETX$ | $SMTX_{0.9}$ |
|---|---|---|---|---|
| $p_1 = \langle v_s, v_d \rangle$ | 1 | 0.13 | 7.69 | 17 |
| $p_2 = \langle v_s, v_1, v_d \rangle$ | 2 | 0.69 | 2.40 | 4 |
| $p_3 = \langle v_s, v_2, v_3, v_d \rangle$ | 3 | 0.94 | 3.06 | 3 |

Table 2.3: Transmission allocation on each link $e \in p$ after each iteration by SMTX to meet the specified target reliability $r_{min} = 0.90$

***Step* I:** For a given source node $v_s$ and destination node $v_d$, and $k$, calculate the set of candidate paths $P_{C_1}(v_s, v_d) \in P$, i.e.,

$$P_{C_1}(v_s, v_d) = P_{min} \subset P, \text{ where } |P_{min}| = k$$
$$\wedge \forall p \in P_{min} \,.\, \forall p' \in P \setminus P_{min} \,.\, \mathrm{ETX}_p \leq \mathrm{ETX}_{p'}$$

Once the candidate set $P_{C_1}(v_s, v_d)$ is created, additional filters are applied. In Step 2, the filtering is based on the length. We filter out paths that are longer than a specific length. Filtering depends on the length of the shortest path $n_{sp}$ (between the source and destination node) and the factor $f_{sp}$, i.e., remove paths that are longer than $\lceil n_{sp} \cdot f_{sp} \rceil$.

Depending on our simulation model used in our evaluation, we selected $f_{sp} = 3$. Fig. 2.4 shows a topology that contains three paths between source node $v_s$ and destination node $v_d$. In the simulation model, the link quality for the link ($v_s$, $v_d$) is determined as 0.13. Further nodes, $v_1$, $v_2$, and $v_3$, are placed as shown in Fig. 2.4. Then, the link qualities of the additional links are determined using the simulation model.

In this topology, there are three paths $p_1$, $p_2$, and $p_3$ between nodes $v_s$ and $v_d$. Tab. 2.3 shows the properties of the paths between $v_s$ and $v_d$. As shown in the

table, the shortest path is the least reliable path. The shortest path $SP_{v_s,v_d}$, $p_1$, has a reliability of 0.13 without any retransmissions on the link. The 2-hop path $p_2 = \langle v_s, v_1, v_d \rangle$ and the 3-hop path $p_3 = \langle v_s, v_2, v_3, v_d \rangle$ have a reliability of 0.69 and 0.94, respectively. Considering this scenario, we selected $f_{sp} = 3$. Thus, if the source and the destination nodes are directly connected, $f_{sp}$ of 3 considers candidate paths of length up to 3.

***Step* II:** For a set of candidate paths $P_{C_1}(v_s, v_d)$, remove the paths that exceed a length $\lceil n_{sp} \cdot f_{sp} \rceil$.

***Step2.* a:** Calculate the length of the shortest path $n_{sp}$ (w.r.t number of hops) between nodes $v_s$ and $v_d$, i.e., $n_{sp} = SP_{v_s,v_d}$ (see Eqn. 2.7)

***Step2.* b:** Then, compute

$$P_{C_2}(v_s, v_d) = \{p \in P_{C_1}(v_s, v_d) \mid |p| \leq \lceil n_{sp} \cdot f_{sp} \rceil\}$$

Next, we remove the paths that contain weaker links. Instead of filtering based on the absolute values for link quality, we filter based on the maximal average number of transmissions $n_{maxAvg}$ required for successful reception. For instance, $n_{maxAvg} = 2$ implies that the paths that contain at least one link $e$ with link quality $q^e$ below 0.5 are filtered out.

***Step* III:** For a set of candidate paths $P_{C_2}(v_s, v_d)$, remove the paths that contain at least one weak link.

$$P_{C_3}(v_s, v_d) = \{p \in P_{C_2} \mid \forall e \in p \; . \; 1/q^e \leq n_{maxAvg}\}$$

Following Step III, we apply our metric SMTX to the candidate set $P_{C_3}$ to find sufficiently bandwidth-efficient paths. The paths with a lower value for SMTX that could satisfy the specified target path reliability $r_{min}$ after applying certain filters are kept. $n_{smtx,min}$ denotes the smallest SMTX value, i.e., lowest bandwidth consumption among all paths in the candidate set $P_{C_3}$. In this step, we filter out paths with slightly higher bandwidth consumption ($f_{smtx} \cdot n_{smtx,min}$) than the lowest possible, and with an average transmission budget per hop greater than $f_b \cdot |p|$. Thus, the factors $f_{smtx}$ and $f_b$ determine the number of possible candidates. For instance, $f_{smtx} = 1.2$ and $n_{smtx,min} = 10$ filters out paths with SMTX higher than 12, and $f_b = 4$ permits an average of four transmissions per hop.

***Step* IV:** For a set of candidate paths $P_{C_3}(v_s, v_d)$, apply SMTX to find bandwidth-efficient paths.

$$P_{C_4}(v_s, v_d) = \{\ p \in P_{C_3}(v_s, v_d) \mid \mathrm{SMTX}^p_{r_{min}} \leq f_{smtx} \cdot n_{smtx,min} \\ \wedge\, \mathrm{SMTX}^p_{r_{min}} \leq f_b \cdot |p|\}$$

where,

$$n_{smtx,min} = min\{\ \mathrm{SMTX}^p_{r_{min}} \mid p \in P_{C_3}(v_s, v_d)\}$$

The route selection process stops if the candidate set is empty or contains only one element. In the latter case, the remaining element is selected as the route. If the set $P_{C_4}(v_s, v_d)$ contains multiple elements, additional filters are applied. The final step applies additional metrics to the possible candidates depending on the routing objectives. The routes are ranked in the following filtering, and the top route is selected.

***Step* V:** Depending on the objective

- For *average performance,* apply ETOP and select the route with the lowest ETOP value.
- For *worst-case performance*, apply SMTX and select route with the lowest SMTX. If there exist multiple possible candidates after the filtering, apply ETOP and select the one with lowest value.

### 2.2.5 Experiments

Next, we present the results of experiments performed to evaluate $r_{min}$-routing and the performance of SMTX compared to the existing routing metrics. Experiments are conducted on top of the WiPS framework running on top of the NS3 simulator [RH10]. $r_{min}$-routes specify the number of transmissions required on each link $e \in p$ to meet the minimum target path reliability. This requirement varies for each route. For instance, route A requires two transmissions on link $e$, whereas route B may require three. Therefore, operating in the standard IEEE 802.11 environment where the number of transmissions on a link is fixed is not sufficient. Rather, it is necessary to enforce the number of transmissions specified by the metric SMTX on a link for each packet. This is managed by the WiPS framework introduced in Sec. 2.4.

The experiment results presented in this section are first reported in [KMG19a]. The motivation of these experiments is to evaluate the performance of $r_{min}$-routing using SMTX. First, we briefly discuss the experimental setup and the operation of

routes in the experiment. Then, we compare the properties of the paths selected by SMTX and other commonly used routing metrics. Finally, we examine the reliability of routes calculated by each metric.

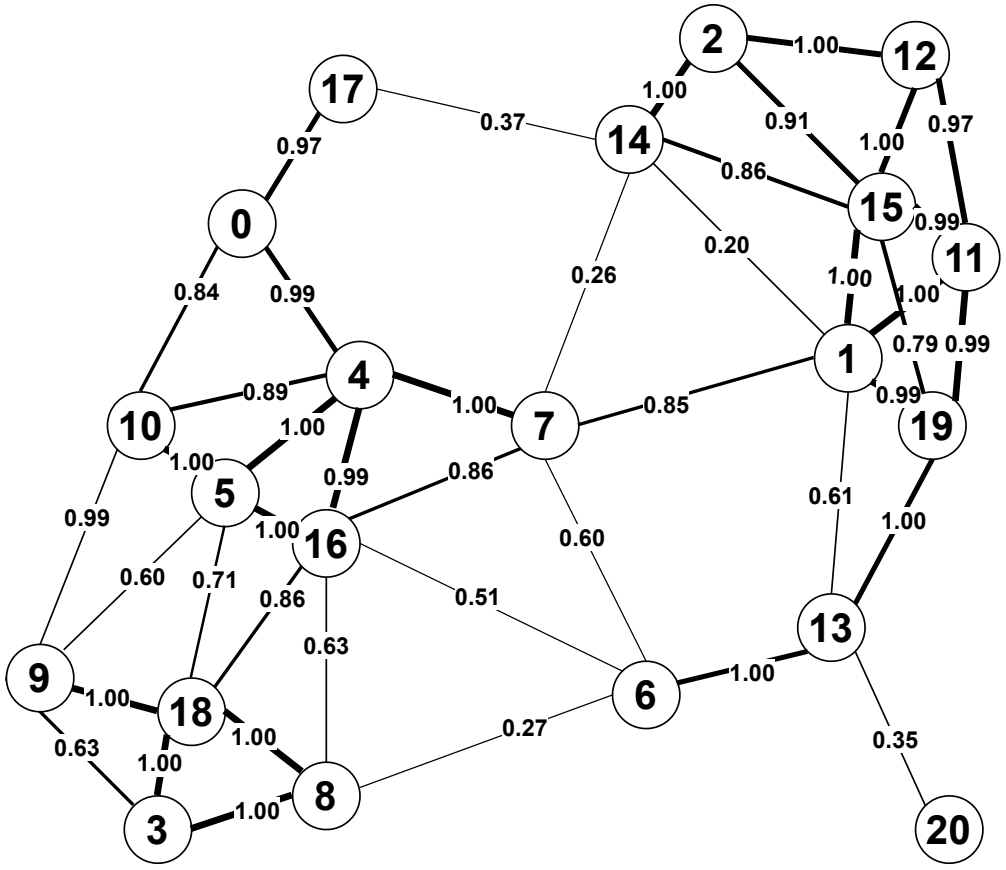



Figure 2.5: Topology used in the experiments

#### 2.2.5.1 Route operation in the experiments

NS3 simulations in the evaluation use the log-distance propagation model to replicate real-world WiFi signal characteristics, i.e., to imitate the loss in signal quality. It is a commonly used model to replicate suburban scenarios [Zon+14]. The random propagation model [SR12] is added on top of the propagation signal loss model to control link reliabilities further.

In these simulations, the PDR values (or link reliabilities) calculated by measuring the success ratio of the isolated probing packet transmissions, are used as link qualities. I.e., isolated probe packets are transmitted by each node in the beginning, and the packets are lost depending on the distance between nodes and the propagation models used. Further, in the evaluation, we assume a central coordinator. The topology information and link qualities determined is available to the central

coordinator. The central coordinator handles the route selection, i.e., calculating the $r_{min}$-routes and the number of transmissions required on each link $e$ to meet the target reliability. The central coordinator is accessible to all nodes without any cost, and thus, nodes issue requests to the central coordinator for a route. During the evaluation, a single route operates at a time, thus, no packet loss due to collisions. I.e., the link reliability or packet loss on a link depends on the propagation models and node positions. Thus, it is possible to control the link reliability and compare the performance using different metrics. Further, 40000 packets are sent through each route selected during the simulation. Fig. 2.5 shows the randomly generated network topology used in the evaluation and the link reliabilities.

| **Field** | **Content** |
|---|---|
| *source* | Source Address |
| *destination* | Destination Address |
| *payload* | Payload in the packet |
| *seq_id* | Unique identifier for the packet |
| *next_hop_list* | List of node address of next hops |
| *smtx_list* | The route to the destination including the required number of transmissions on each link |

Table 2.4: Packet format

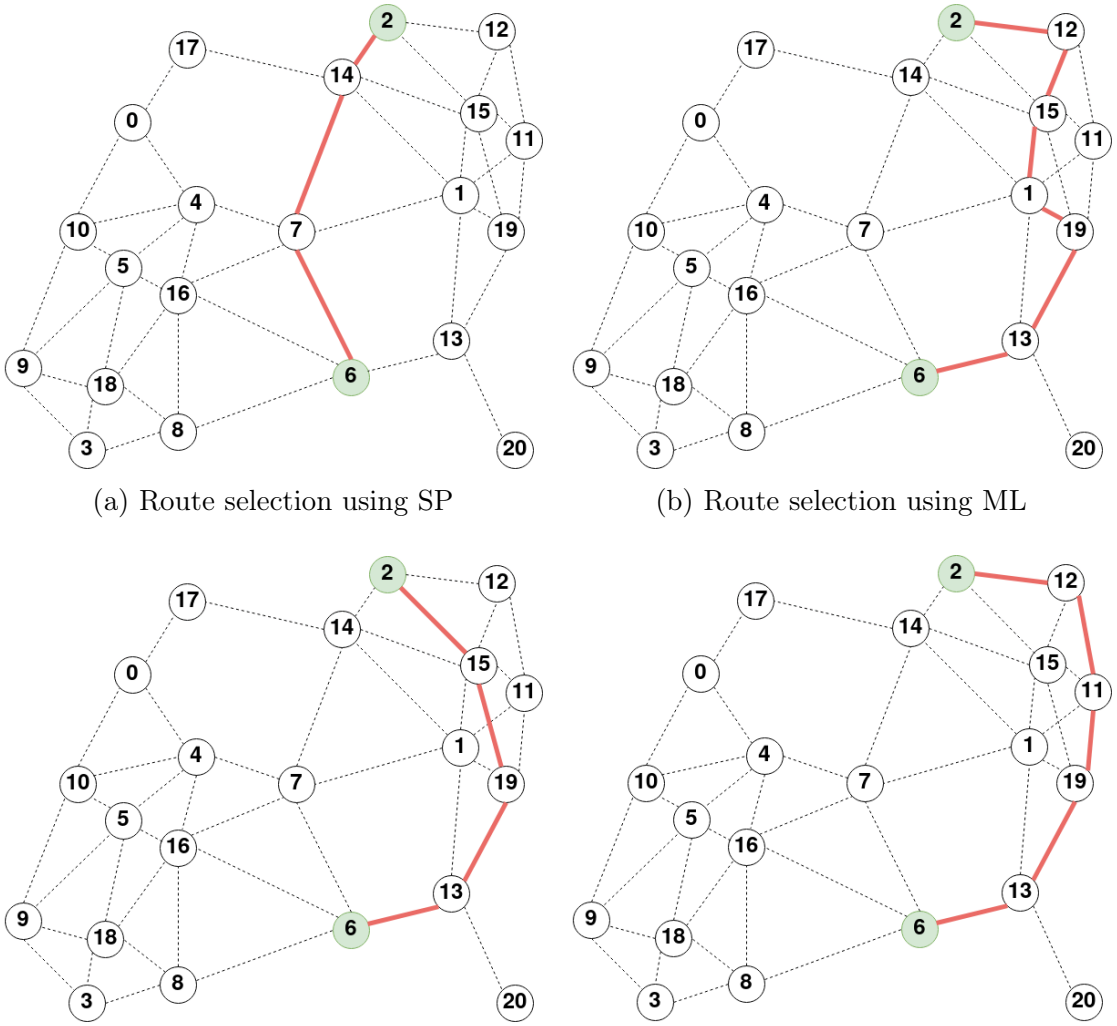
As mentioned earlier, the experiments are conducted on top of the WiPS framework (see Sec. 2.4). WiPS handles the WiFi frame creation, transmission, forwarding, and processing of the packets received. In this simulation, the packets created in the application layer move into the routing layer where necessary routing information is appended and are finally pushed into the MAC layer.

Tab. 2.4 shows the format of a packet used in the simulation. Every packet contains a fixed size payload inserted by the application layer. The *source* and *destination* field contains the source address and the destination addresses of the nodes, respectively. When a node receives a packet, it either processes or forwards the packet, depending on the destination address. The unique *seq_id* identifies a packet and is used to calculate the path reliability. The *next_hop_list* contains the next hops. Further, the *smtx_list* contains the links in the path and the number of transmissions required to meet the target path reliability.

#### 2.2.5.2 Comparing the properties of paths selected by each metric

Next, we compare the properties of the paths selected by each metric between two nodes. Fig. 2.6 shows the different paths selected by each metric to route from

node 6 to node 2. Further, Tab. 2.5 shows the different metric values for each path selected by the metrics. The specified target reliability $r_{min}$ for the experiments is 0.9.




(a) Route selection using SP

(b) Route selection using ML

(c) Route selection using ETX

(d) Route selection using SMTX

Figure 2.6: Routes calculated by multiple metrics between from 2 to 6

| Path | Selected by | $SP$ | $ML$ | $ETX$ | $\text{SMTX}_{0.9}$ |
|---|---|---|---|---|---|
| $p_1 = \langle 6, 7, 14, 2 \rangle$ | SP | 3 | 0.16 | 6.60 | 14 |
| $p_2 = \langle 6, 13, 19, 1, 15, 12, 2 \rangle$ | ML | 6 | 0.99 | 6.01 | 6 |
| $p_3 = \langle 6, 13, 19, 15, 2 \rangle$ | ETX | 4 | 0.72 | 4.36 | 6 |
| $p_4 = \langle 6, 13, 19, 11, 15, 2 \rangle$ | $\text{SMTX}_{0.9}$ | 5 | 0.95 | 5.05 | 5 |

Table 2.5: Paths selected by each metric from node 6 to 2

Fig. 2.6a shows the shortest path $p_1 = \langle 6, 7, 14, 2 \rangle$ from node 6 to 2 selected by the metric $SP$. As discussed earlier, multiple studies [Cam+08; De +03] show that

using the shortest path as a routing metric results in poor network performance. Path $p_1$ has an aggregated path reliability of 0.16 without any retransmission, and it is the lowest compared to the paths selected by any other metric. Moreover, to meet the specified target reliability $r_{min}$ of 0.90, 14 transmissions are required across the links (column $SMTX_{0.9}$).

ML metric calculates the most reliable path without any retransmissions. Using such a criterion for route selection often results in longer paths. Here, the path $p_2 = \langle 6, 13, 19, 1, 15, 12, 2 \rangle$ selected by ML from node 6 to 2 is the longest (6 hops, see Fig. 2.6b) path compared to the paths selected by any other metrics. Path $p_2$ has an end-to-end reliability of 0.99 (without any retransmissions) and meets our specified target reliability. However, it is not the most bandwidth-efficient route between the nodes, as it requires at least six transmissions to reach the destination.

Path $p_3 = \langle 6, 13, 19, 15, 2 \rangle$ (see Fig. 2.6c) is the path with the smallest ETX value. Path $p_3$ selected by ETX is a compromise between reliability and length. Compared to path $p_2$ selected by ML, path $p_3$ is short and produces a low average medium load, i.e., a decrease of 28% in medium load (($\text{ETX}^{p_2}$ -$\text{ETX}^{p_3}$)/$\text{ETX}^{p_3}$). However, similar to path $p_2$, it is not the most bandwidth-efficient path. Fig. 2.6d shows path $p_4 = \langle 6, 13, 19, 11, 15, 2 \rangle$ selected by SMTX, which requires the smallest number of transmissions to meet the target reliability. Path $p_4$ requires only five transmissions to meet target reliability $r_{min} = 0.90$.

#### 2.2.5.3 Comparing the properties of paths selected by each metric

SMTX selects the path that meets the target reliability with the minimum number of transmissions, and returns the required number of transmissions on each link. This information, together with $r_{min}$-routing, has advantages over conventional routing. SMTX focuses on boosting the links on a path with the highest proportional gain. Often, stable or highly reliable links do not require the default number of retransmissions (as per the standard). I.e., some losses are tolerable, and a single transmission is often required on highly reliable links to meet the path reliability requirement. Depending on the reliability requirement, the node decides whether to retransmit or drop a failed transmission attempt, i.e., avoiding over-provisioning. Another advantage is that by saving the transmission attempts on stronger links and redistributing the budget among weaker links, it is possible to transform the weaker links as feasible. For instance, consider a link $e$ with quality $q^e = 0.3$. With the default number of transmissions $n_{max} = 4$ (as per the standard), the link quality is boosted to $q_4^e = 0.76$. Consider a path $p$ consisting the weaker link, for instance, $p = \langle e_1, e_2, e_3 \rangle$, where the link qualities are $q_1^{e_1} = 0.9$, $q_1^{e_2} = 0.8$, and $q_1^{e_3} = 0.3$. When

applying the default number of transmissions of 4 to all links in the path, the path reliability $q^p_{n^p_{max}}$ is increased to 0.758 (applying Eqn. 2.1 and Eqn. 2.4).

$$q_4^{e_1} = 0.9999$$
$$q_4^{e_2} = 0.9986$$
$$q_4^{e_3} = 0.7599$$

thus, the path quality

$$q_{12}^p = 0.9999 \cdot 0.9986 \cdot 0.7599 = 0.7586$$

Whereas by distributing the transmission budget differently, with two transmissions on the first two links and up to eight transmissions on the last link or the weak link, the path reliability is boosted to 0.91.

$$q_2^{e_1} = 0.99$$
$$q_2^{e_2} = 0.96$$
$$q_8^{e_3} = 0.942$$

thus, the path quality

$$q_{12}^p = 0.99 \cdot 0.96 \cdot 0.943 = 0.90$$

Notable, with the same transmission budget, i.e., a total of 10 transmissions, it is possible to improve the link quality up to 0.90 using SMTX compared to 0.758 with a static transmission limit.

To demonstrate the benefits of $r_{min}$-routing and compare it with routes calculated by other metrics, we operate multiple routes from node 7 to all other nodes in the topology depicted in Fig. 2.5. As mentioned earlier, the topology information and the link qualities are available to the central coordinator by sending isolated probe frames. This information is used for route selection. Further, at a time, only one route is operated, thus no collisions. Several packets are sent from node 7 to all other nodes in the network. Packets are sent along each route determined by each metric from node 7 to another node. For instance, multiple packets from node 7 to 4 are sent along paths selected by the metrics. In total, there are 20 different destinations from node 7 (topology consists of 21 nodes). The $r_{min}$-routes are calculated using the route selection process using SMTX (see Sec. 2.2.4). During the operation of routes selected by the metrics ML and ETX, we set the default number of transmissions $n_{max}$ to 4, and for $r_{min}$-routing using SMTX, the maximal average number of transmissions $n_{maxAvg}$ is also set to 4.

| Metric | Hops | Success Ratio | TX Per success | $< r_{min} = 0.9$ |
|---|---|---|---|---|
| SP | 2.95 | 94.8% | 2.85 | 20% |
| ML | 4.10 | 98.8% | 3.41 | 5% |
| ETX | 3.15 | 98.8% | 2.55 | 5% |
| $SMTX_{0.9}$ | 3.20 | 97.0% | 2.58 | 0% |

Table 2.6: Properties for routes selected by each metric from node 7 to all other nodes

Tab. 2.6 shows the results or the properties of routes using each metric. The table shows the average value of all routes selected by each metric. For instance, the routes selected from node 7 to other nodes using SP as the routing metric have an average length of 2.95 hops (the first column). Further, on average, the success ratio of the routes selected by SP is 94.8% (second column). The third column shows the ratio of the number of transmissions (including failed and successful) to the number of packets successfully delivered to the destination node. The final column shows the ratio of routes selected by a metric that failed to meet the target reliability $r_{min}$ of 0.90. For instance, routes to four other nodes from node 7 (20% of 20 possible destinations) selected using SP metric failed to meet the target reliability $r_{min}$ of 0.90.

The routes selected using ML metric are the most reliable routes without any retransmissions, though the longest, i.e., 4.1 hops on average. Contrary, the routes selected by SP are the shortest and the least reliable. These results are consistent with operating a single route (see Sec. 2.2.5.2). Even though these are the most reliable (selected using ML) and shortest (selected using SP), routes require more transmission attempts per successfully delivered packets than routes selected using ETX or SMTX. As expected, the routes selected by SP perform the worst considering the path reliability, which is not further discussed in our evaluation. Among those metrics, only SMTX (with $r_{min}$-routing) can calculate and allocate the transmission budget to meet the target reliability. As shown in the table, only SMTX meets the reliability requirement ($r_{min} \geq 0.9$) for all cases or routes from node 7. The experiment settings probably influenced these results, i.e., the link reliabilities are constant, and the outcome of transmission events is statistically independent. In practice, the link qualities and statistical link properties may vary during the operation, and it is critical to address these aspects for the successful operation of $r_{min}$-routes.

The topology used in this evaluation consists of highly reliable links. Therefore, the routes selected using the metrics ML and ETX meet the reliability requirement

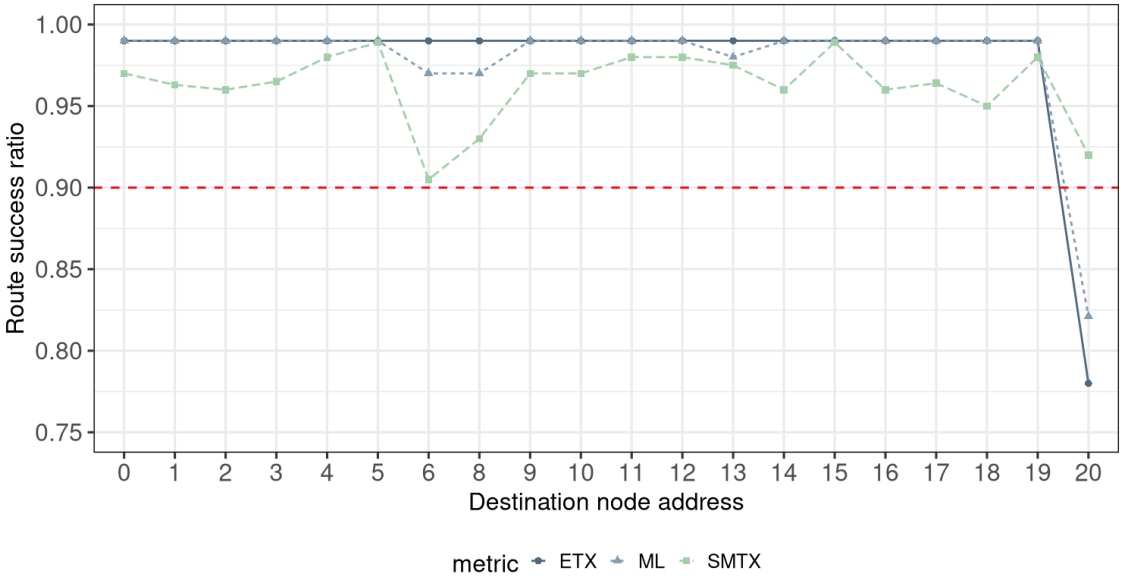



Figure 2.7: Success ratio of routes selected by each metric from node 7 to other nodes in the network. The red dotted lined denotes the minimum target path reliability $r_{min}$.

| Metric | Path |
|---|---|
| ML | $p_1 = \langle 7, 1, 19, 13, 20 \rangle$ |
| ETX | $p_2 = \langle 7, 6, 13, 20 \rangle$ |
| SMTX | $p_3 = \langle 7, 1, 19, 13, 20 \rangle$ |

Table 2.7: Path selected by each metric from node 7 to 20

for almost all routes except one case, i.e., the route to node 20. Fig. 2.7 shows the success ratio of the routes from node 7 to all others selected by each metric. Only SMTX using $r_{min}$-routing met the target reliability in all cases, especially the route to node 20, where the link between nodes 13 and 20 is long and unreliable.

Tab. 2.7 shows the path selected by metrics ML, ETX, and SMTX from node 7 to 20. As mentioned, the default number of transmissions assigned while operating routes selected using ML and ETX, and the maximal average number of transmissions assigned to SMTX is the same, i.e., $n_{max} = n_{maxAvg} = 4$. I.e., using the default number of transmissions on paths selected by the metric ETX and ML, the path quality can be increased (applying Eqn. 2.1 and Eqn. 2.2),

using default number of transmissions on links $p_1$ and $p_2$,

$$\begin{aligned} q_{16}^{p_1} =& (1-(1-q^{e^1})^4)\cdot(1-(1-q^{e^2})^4)\cdot(1-(1-q^{e^3})^4)\cdot(1-(1-e^4)^4) \\ =& (1-(1-0.85)^4)\cdot(1-(1-1)^4)\cdot(1-(1-1)^4)\cdot(1-(1-0.35)^4) \\ =& 0.9995\cdot 1\cdot 1\cdot 0.8215 \\ =& 0.821 \\ q_{12}^{p_2} =& (1-(1-0.60)^4)\cdot(1-(1-1)^4)\cdot(1-(1-0.35)^4) \\ =& 0.80 \end{aligned}$$

using the budget allocation determined by SMTX on link $p_3$,

$$\begin{aligned} q_{10}^{p_3} =& (1-(1-0.85)^2)\cdot(1-(1-1)^1)\cdot(1-(1-1)^1)\cdot(1-(1-0.35)^6) \\ =& 0.902 \end{aligned}$$

Path $p_1$ selected using the ML, with 16 transmissions in total (4 transmission on each link), was unable to meet the target path reliability $r_{min} = 0.90$. Similarly, path $p_2$ selected using ETX with 12 transmissions in total. However, a variable transmission allocation (of 10 transmissions) using SMTX across the path met the target path reliability. In this scenario, it is necessary to allocate more transmissions to the weaker link. SMTX reallocates the transmission budget across the path by reducing the number of transmissions permitted to stronger links and assigning them to the weaker ones. Thus, in a less reliable network, the redistribution mechanism using SMTX leads to a significant increase in path reliability.

Further, the SMTX avoids over-provisioning, which is reflected in the results. Using other metrics, the success ratio of the routes is significantly higher than the target (see Fig. 2.7). However, avoiding over-provisioning has only a minor effect on a few routes, i.e., the observed path reliability is not significantly higher than the target path reliability. This is due to the shorter routes evaluated in this experiment (all nodes are at least 4-hop distance from node 7), and as route length increases, SMTX has a higher degree of freedom to redistribute the transmission budget across the links.

### 2.2.6 Conclusion

This section introduced $r_{min}$-routing, a novel approach for reliability-constrained routing. $r_{min}$-routing establishes bandwidth-efficient routes that satisfy specified minimum target reliabilities $r_{min}$. The routing metric SMTX is developed to identify

the optimal number of transmissions across a feasible path to meet the minimum target reliability. $r_{min}$-routing, together with SMTX and other routing metrics such as ETX and ETOP, determine bandwidth-efficient $r_{min}$-routes and, notably, meet the specified target reliability. Simulation experiments were conducted to compare the performance and advantages of $r_{min}$-routing to the existing metrics. However, the performance of $r_{min}$-routing in practice relies on assumptions and underlying methodologies that provide information to calculate $r_{min}$-routes.

## 2.3 Further contributions

$r_{min}$-routes calculated by the selection process return the required number of transmissions at each link to satisfy the target reliability. The route selection depends on the given network topology information. In the experimental evaluation using NS3 (see Sec. 2.2.5), the link qualities are stable and not affected by frame collisions or the influence of external nodes. Moreover, the bandwidth availability or requirements are not considered in the simulation. However, in practice, the successful operation of $r_{min}$-routing requires several pre-requisites. For instance, the link reliabilities may change during the operation and require recalculating or repairing the routes. As mentioned earlier, in this thesis, we focus on developing the necessary algorithms and metrics for the successful operation of $r_{min}$-routing. The methodologies presented in the thesis are hardware-independent and pertinent for other reliability-constrained routing protocols.

The crux to the operation of $r_{min}$-routing and several other routing protocols is the assumption of statistical independence. Transmission event outcomes on a link depend on multiple factors. For instance, it is influenced by the network topology, external networks or devices operating on the same channel, the characteristics of applications running on each node, or some unfamiliar events! First, it is vital to evaluate the validity of the assumption "statistical independence" in practice. In the thesis, we investigate the statistical properties of the links in multiple channel conditions and develop metrics to measure the strength of certain statistical properties of links. We discuss further on this in the Chapter 3 and Chapter 4.

Among the factors that could potentially influence the statistical characteristics of a link, there are factors that a network under the user's control could manage, for instance, medium usage by a node or the traffic pattern generated by the nodes. Chapter 4 presents the metric and algorithm to monitor, manage the medium, and shape the traffic generated by a node. It is possible to incorporate the feedback from these protocols during the calculation of $r_{min}$-routes. Depending on the traffic

profile (i.e., characteristics of the traffic generated by an application), link quality, and SMTX values of a path, it is possible to calculate the bandwidth requirements, and add additional filters to the route selection process.

Another factor that influences the route selection and the operation of $r_{min}$-routes is the link quality estimation. $r_{min}$-routes select a route and calculate the transmission requirements based on the link quality $q^e$ of links. In the simulation experiments so far, the link qualities are not influenced by external networks. However, in practice, the link reliabilities may change during the operation and require recalculating or repairing the routes. Therefore, the metric used to estimate the link quality is a vital parameter for the successful operation of $r_{min}$-routes. Along with the link reliability (or PDR), it is necessary to contemplate other factors, such as volatility (variation in the link reliability) and stability, when estimating an adaptive and conservative link quality. Chapter 6 presents a link metric developed by reckoning reliability-constrained routing in wireless networks.

Summarizing the problems mentioned above, the remaining parts of the thesis focus on answering the following questions:

1. What are the statistical properties required for improving link reliability by retransmissions?

2. What statistical tests can measure the statistical properties necessary to improve the link reliability by retransmissions?

3. What are the characteristics of wireless links in different channel conditions?

4. Do application characteristics influence the statistical link properties? If yes, how to mitigate their influence?

5. What are the possible methods to monitor the medium load using off-the-shelf WiFi hardware?

6. Are existing link quality estimation metrics suitable for reliability-constrained routing?

## 2.4 Implementation framework and the hardwares

This section presents the framework and hardwares used to evaluate the methodologies introduced in the thesis. The methodologies developed in this thesis are hardware-independent and implementable on top of off-the-shelf (commodity) WiFi hardwares. To control the operations of WiFi hardware, the framework *WiFi Inte-*

*grated Protocol stacks* (WiPS) is developed. We discuss this further in Sec. 2.4.1. Then, in Sec. 2.4.2, we discuss the hardware platforms used in our experiments.

### 2.4.1 WiPS Framework

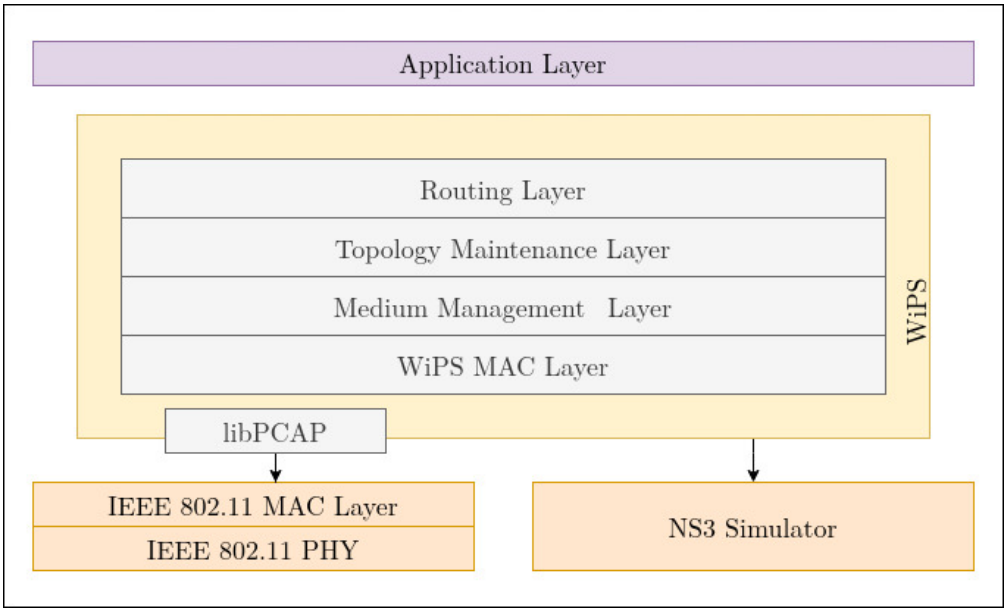



Figure 2.8: The architecture of the WiPS framework with multiple protocol layers

*WiFi Integrated Protocol stacks* (WiPS) is an extensible framework designed and developed for implementing and integrating higher-level protocols on top of IEEE 802.11. WiPS runs on top of Linux, and supports different WiFi hardwares by encapsulating the hardware using *libPCAP* [JM09] and the simulation platform NS3 [RH10]. Fig. 2.8 shows the architecture of the WiPS framework. WiPS implements a minimized MAC layer based on libPCAP API on top of IEEE 802.11 MAC to function on WiFi hardwares, and for the simulations on the NS3 simulator, based on the interface provided by the simulator. Higher-level protocols in WiPS and applications implemented using it use this MAC layer. Further, this WiPS MAC layer supports local addressing (without using the WiFi adapter's MAC address), broadcasts, acknowledgements, automatic retransmissions, sequence numbers, duplicate filtering, and enforces the specified number of transmissions for a frame (for instance, determined by the Routing Layer).

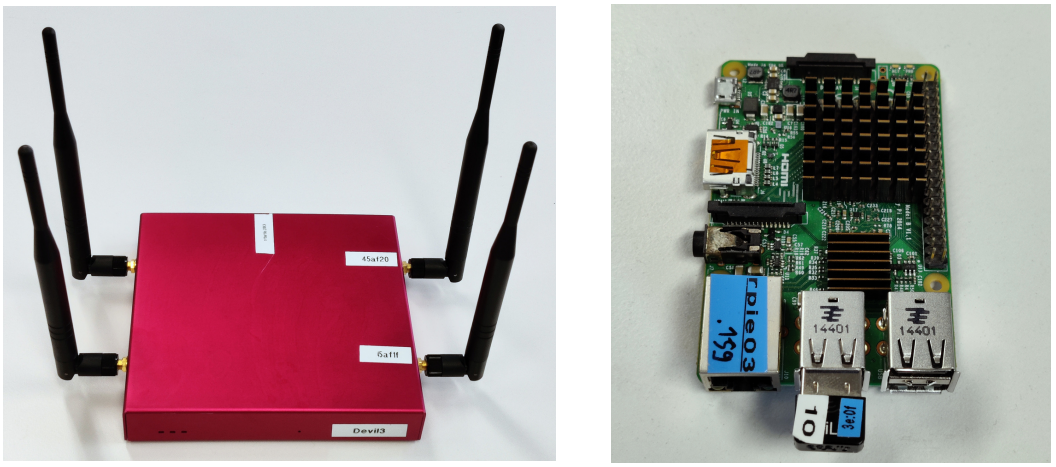
WiPS itself is flexible such that each application using WiPS can select the appropriate subset of protocols integrated into WiPS, depending on the use case. Based on the selection, it builds a tailored protocol stack solely consisting of the selected protocols used by the application. For instance, if the application requires sending

broadcast frames in a 1-hop distance, it can select the subset of protocols excluding the Routing Layer to build the frames solely for this use case. As depicted in Fig. 2.8, WiPS consists of multiple protocols layers, for instance, a Routing Layer, a Topology Maintenance Layer, a Medium Management Layer, and finally, the WiPS MAC layer. Further, WiPS is extensible such that it is possible to integrate additional protocols into the existing design. Frame payload (of a WiFi frame) consists of the data from each protocol used by the application, and WiPS handles the serialization and deserialization of the data from each layer.

As stated, WiPS supports multiple platforms. WiPS MAC layer uses libPCAP API to communicate between the WiFi hardwares, whereas it uses the interface provided by the NS3 simulator to communicate between simulator modules replicating IEEE 802.11 MAC layer. I.e., the code structure of the protocols implemented in the framework and the applications using the framework are independent of the experimenting platform. There are certain requirements when an application built on WiPS uses WiFi hardware. It is assumed that no other application uses the WiFi hardware when WiPS uses it. Further, to have complete freedom on the operations of the WiFi hardware, it must operate in *monitor mode*, which is supported by almost all available off-the-shelf commodity WiFi hardwares. The WiFi hardware or adapter is not associated with any Access Point (AP) when operating in *monitor mode* and it passes all the frames received to the higher layers. In contrast, a WiFi adapter operating in *managed mode* and connected to an AP only passes the frames destined to the adapter. Importantly, an adapter operating in *monitor mode* exchanges further meta-data with higher layers via radiotap headers [Rad], for instance, the signal strength at which the frames are received. This information (meta-data of a received frame) is useful for developing higher layer protocols. Moreover, while operating in *monitor mode*, WiPS can control or specify the number of transmissions assigned to a frame, and it is crucial for the functioning of $r_{min}$-routing.

### 2.4.2 Hardwares used in evaluations

Next, we discuss the hardwares used in our experiments. We use two different hardware platforms for real-world experiments. The testbed deployed inside an office building, used to conduct experiments in the presence of external traffic, uses nodes built using the PC Engines APU2 Single Board Computers (SBCs). For the controlled experiments in the absence of external traffic, we use a mobile testbed that consists of nodes built using the Raspberry PI SBCs.



(a) APU2 testbed node (b) Raspberry Pi testbed node

Figure 2.9: Testbed nodes built using APU2 and Raspberry Pi SBCs

### PC Engines APU2

APU2 is a Single Board Computer (SBC) developed by PC Engines [PCE], specifically for developing networking applications such as building routers, firewalls, dedicated servers, and for educational purposes. Discussing the hardware specifications, it has a 64-bit processor from AMD Embedded G series System-on-Chip (SOC) 412TC with CPU frequency 1 GHz. Multiple variants of the APU2 SBCs are available, and the hardware used in the testbed supports 4 GB RAM and uses an m-SATA SSD hard drive. Notably, APU2 has two miniPCI express slots, allowing plugging in WiFI adapters. Compared to USB WiFi adapters, WiFi adapters connected via PCI interface are faster and use external antennas with higher transmission power. Two WiFi adapters from Compex WLE200NX are plugged into a device or node in the testbed. Compex WLE200NX consists of an Atheros chipset using ath9k drivers that operates in 2.4 and 5 GHz. A single testbed node built using APU2 SBCs (see Fig. 2.9a) costs around 150 Euro, including the cost for an APU2 board, the hardware case, the power supply unit (PSU), WiFi Adapters and antennas, and SSD.

### Raspberry Pi 2

Raspberry Pi is an SBC based on Broadcom's SOC chips developed by the Raspberry Pi foundations [Gay14]. Different variants of Raspberry pi boards are available in the market, where all variants consist of CPU with ARM architecture. In our experiments, we use Raspberry Pi 2 B rev 1.1 and Raspberry Pi 3 B rev 1.2. The former is powered by a 32-bit quad-core ARM Cortex-A7 processor with a clock speed of

900 MHz, whereas the latter by a 64-bit quad-core ARM Cortex-A7 processor with a higher clock speed of 1200 MHz. Both models have 1 GB RAM and a similar number of interfaces (USB, HDMI, and Ethernet ports). A micro SD card is used as persistent memory storage. Unlike the APU2 SBCs, Raspberry pi's do not have PCI interfaces, and thus, USB WiFi adapters are used in the experiments. Logilink WL0084B and TP-Link TL-WN772N USB adapters are used in the experiments. The former is built with a Ralink chipset using rt2800 Linux drivers, whereas the latter with Atheros AR900U chipset that uses ath9k drivers. A testbed node built using Raspberry pi (see Fig. 2.9b) costs below 50 Euros, including the cost for the PSU, SD-card, and USB WiFi adapter. Compared with APU2, Raspberry pi's are cheaper and smaller, whereas APU2 SBCs are faster and have a higher processing power.

## 2.5 Summary

This chapter introduced $r_{min}$-routing, focusing on reliability-constrained routing that discovers and operates $r_{min}$-routes with specified statistical minimum reliabilities. $r_{min}$-routes are bandwidth-efficient routes that use a minimum number of transmissions to meet the specified reliability. Routes are determined using multiple filtering criteria using multiple metrics, where the metric SMTX determines the optimal minimum number of transmissions on a feasible path. Simulation experiments are conducted to elucidate the performance of $r_{min}$-routing. However, the successful operation of $r_{min}$-routes in practice requires multiple pre-requisites. One of the requirements is to enforce the number of transmissions specified by SMTX on a frame. The framework, WiPS discussed earlier, ensures that a frame attempts the number of transmissions specified by SMTX. Further, in Sec. 2.3, we discussed the essentials investigated in the remaining of this thesis for the successful operation of $r_{min}$-routes in practice.

# 3

# Statistical Foundations for Improving the Reliability of Wireless Links through Retransmissions

## Contents


**3.1 Introduction** . . . . . **40**
**3.2 State of the art** . . . . . **42**
**3.3 Statistical Methods** . . . . . **44**
3.3.1 Gilbert-Elliot Model . . . . . 44
3.3.2 Runs-Test . . . . . 46
3.3.3 Chi-square test . . . . . 48
**3.4 Experimental Assessment** . . . . . **50**
3.4.1 Experiment Setup . . . . . 50
3.4.2 Synthetic data . . . . . 54
3.4.3 Real link data from a collision-free environment . . . . . 64
3.4.4 Environment with foreign traffic . . . . . 68
**3.5 Conclusion** . . . . . **79**

## 3.1 Introduction

"We assume statistical independence" is a common postulation in many papers on communication protocols for nondeterministic wireless networks. Statistical independence is a concept in probability theory where the outcome of an event is independent to the outcome of a previous event or events. Communication protocols assume that the transmission outcomes of a link are statistically independent and model it as a Bernoulli distribution [Dod08]. Together with the model, it is important to understand and measure the strength of assumptions used as the foundation.

In [Bia00], the author shows that the frame collision probability in 802.11 networks, referred as *conditional collision probability* is independent when the number of nodes competing for the medium is fairly large[4]. As the number of contenders increases, depending on their traffic pattern, collisions become random or unpredictable. Considering this result, several protocols assume the probability of a successful transmission over a link is independent of the outcome (outcomes) of previous event (events), i.e. "statistically independent"

Several reliability-constrained routing protocols [Jak+11][YYL11][KMG19a] for their functioning assume collision in a link as "statistically independent". With this assumption, link reliability can be improved to target reliability by retransmissions, and eventually, multiply link reliabilities to measure the reliability of a path. This is a two-step process. We use the terminology *link boosting* to describe the first step. This chapter focuses on *link boosting* and laying foundations to investigate possible statistical properties required to improve link reliability by retransmissions.

To achieve a specific target reliability $r^e_{target}$ for a link $e$, a common practice is to estimate the link reliability $q^e$, for instance in terms of packet delivery ratio (PDR), and calculate the required number of (re-)transmissions $n^e_{max}$ such that the boosted link reliability $q^e_{n_{e,max}}$ is larger or equal to $r^e_{target}$:

$$q^e_{n^e_{max}} = 1 - (1 - q^e)^{n^e_{max}} \tag{3.1}$$

where,

$q^e$ = a function that returns the aggregated link quality of link $e$
$n^e_{max}$ = maximum number of transmissions on link e
$q^e_{n_{e,max}}$ = boosted link reliability

This equation is only valid under the assumption that the success probability for each transmission event is independent to the outcome of previous event/ events.

[4]A minimum of 10 nodes in an interference domain

To apply Eqn. 3.1, results of transmissions on a link are mapped into Bernoulli trials. A transmission event with outcome 1 denotes a *success* event and 0 a *failed* event. Next, the PDR or probability for an event to be a *success* event, P(X==1) of a given Bernoulli trial is calculated. If the events are statistically independent, the probability for an event to be a *success* event is independent to the outcome of a previous event or events. From calculated historical PDR, a link reliability metric estimates PDR for upcoming transmission events. This estimated PDR is used for *link boosting* by assuming events are independent.

Further, is it necessary to consider "very long-term" dependencies for *link boosting*, or is it sufficient to consider short-term dependencies? To perform *link boosting*, the link reliability (PDR) of a link is often recalculated (recalculation frequency depends on the reliability metric). *Link boosting* becomes successful when the success probability of each transmission event(s) is greater than or equal to the estimated link reliability. For this use case, it is sufficient to measure short-term dependencies of events, as link reliability is often recalculated.

In this chapter, we investigate prerequisites for *link boosting* and explore suitable statistical methods that can be used at run-time to assess the strength of respective prerequisites. Following are the questions to be considered:

***Question* I:** What are the statistical properties required for improving link reliability by retransmissions?
This question is decomposed into the following sub-questions.

- ***Q* I.a:** Is it necessary that the outcome of transmission events on a link is statistically independent of the previous event/events? I.e., to understand and measure the strength of *statistical independence*.
- ***Q* I.b:** Is it required that the outcome of transmission events are random? I.e., to understand and measure *randomness*
- ***Q* I.c:** Is it needed that the outcome of transmission events fits a Bernoulli distribution? I.e., to check the degree of fitness of the *Bernoulli distribution* for transmission events of a link

***Question* II:** What statistical tests can be used to measure the statistical properties addressed in Question I.?

***Question* III:** What are the characteristics of wireless links in different medium conditions?

To find an answer to Question I, first, we need to find the solution for Question II, i.e., possible methods to measure various statistical properties of a link. Statistical methods such as Gilbert-Elliot Model [Gil60][Ell63], Wald–Wolfowitz Runs-Test [Bra68], and Chi-square goodness-of-fit test are possible candidates to find solutions for the Question II. We apply those methods over synthesized data of known types to verify whether tests provide expected results. To understand the link properties in different medium conditions (Question III), we apply various statistical methods to data collected from experiments in a collision-free environment and real-world medium conditions. Depending on our findings, we select methods that suit our requirements, thus answer Question I.

The remaining portion of this chapter is structured as follows: We first discuss existing studies in Sec. 3.2. In Sec. 3.3, we discuss statistical methods used for link evaluation. Later, we present experimental setups and results in Sec. 3.4. Finally, in Sec. 3.5, we conclude our findings and revisit the questions mentioned earlier.

## 3.2 State of the art

Frame transmissions on a wireless link are failed either due to collision or network's physical layer properties[5]. Temporal properties (transmission timings) of the network may cause frame collision, while transmission failures due to spatial properties may occur due to weak reception. Signal reflection or refraction, spatial and other signal characteristics affects the reception.

The temporal properties of a network depend on the medium access method used in nodes and the applications running on it. As mentioned earlier, the author [Bia00] shows that collisions become random in the case of 802.11 networks with fairly numbered nodes. In [Hum77], the author articulates the role and relationship between independence and randomness in probability theory.

Physical layer properties are another reason for a *failed* transmission event. In [Sri10], the author studied the independence characteristic of wireless links in a collision-free medium. For a wireless link, the author evaluated short-term and long-term dependencies. Experiment results showed that the most links are "bursty" in nature, i.e., the outcome of a transmission event depends on the previous transmission or transmissions(s). The study concludes that random back-off is ineffective if the frame loss is not due to collision and vice-versa.

In [Sri+08], the authors introduce an offline metric called $\beta$- metric to measure the link burstiness. A link is *bursty* if it has a statistically dependent delivery ratio,

[5]Assuming no decoding errors due to damaged hardware

i.e., in the opposite spectrum of a statistically independent delivery ratio. In other words, for a *bursty* link, the probability of the next successful transmission depends on previous events. $\beta$-metric uses the Conditional Probability Distribution Function (CPDFs) and Kantorovich-Wasserstein (KW) distance to measure the burstiness. First, it calculates the conditional probability function, *c(n)* for all transmission streaks[6] of length $n$ that the next transmission is a success. A positive value for $n$ denotes the streak of successful transmissions with length $n$ and a negative value for $n$ denotes the streak of failed transmissions with length $n$. For example, $c(4)$ calculates the probability of a successful transmission after 4 successful transmissions, and similarly, $c$(-4) calculates the probability of a successful transmission after 4 failed transmissions. A $\beta$-value of 1 and 0 represents a bursty and statistically independent link, respectively. A negative $\beta$ shows a negative correlation between the events, and there is no lower bound or limit, i.e., $\beta$-value of -1, -5 or -10 is possible. Hence, a negative value for $\beta$ does not provide much information about the strength of the negative correlation.

In their experimental analysis [Sri+08], the authors found that most of the links behaved as bursty links in the absence of collisions. $\beta$-metric is designed to measure the burstiness and exploit the burstiness for Opportunistic Routing (OR) [BM04]. The computational complexity of calculating the $\beta$-metric is high. It requires a large dataset, and therefore it is only suitable for offline analysis. In [Sri10], the author emphasizes the need for an online evaluation to measure the burstiness and evaluate the characteristic of links in the presence of collision. For Opportunistic Routing, [Ali+11] introduced a combination of online metrics, $MAC_3$ and EFT, to calculate the burst length and estimate the burstiness of wireless links.

A simple method to measure short-term dependencies is using a 2-state Markov chain known as Gilbert Elliot (GE) model. In [Yaj+99], the authors use 2-state Markov chain (GE model) and $k$-th order Markov chain model to measure the temporal dependencies of Internet packet losses. Similarly, in [HH08], the authors use the GE model with h=0 to capture the temporal correlation of packet loss in real time Internet traffic. Further, in [Wen+07], the authors use a similar model to measure bursty losses in wireless links by extending the loss state to model consecutive frame losses. In [Sri10] and [Bil+15] discusses the use of *channel memory* $\mu$ parameter calculated by GE model in measuring the dependency between events. We discuss it further in the next section.

[6] A streak is a continuous sequence of events of successful or failed transmissions.

## 3.3 Statistical Methods

Statistical independence is a "relationship property". It shows the relationship between events or the influence of an event on a future event. From the perspective of transmission events on a link, for a given set of Bernoulli trials with success probability $p$, to assume the sample as a statistically independent, the following conditions need to meet:

- the success probability of each event must be equal to $p$, and
- implicitly the failure probability of each event must be equal to *1 - p*

Each continuous sub-sample (of any length) of the sample set should also meet the aforementioned conditions. I.e., for a sample (Bernoulli trial) of length $n$, any continuous sub-sample (of the length between 1 and $n-1$) must have a success ratio $p$.

In this section, we discuss different statistical methods used in our evaluation. Following tests are used to evaluate the transmission outcome on a link,

- *Channel memory* ($\mu$) from Gilbert-Elliot model,
- Wald-Wolfowitz Runs-Test, and
- Chi-square test for goodness of fitness.

The $\mu$ parameter from Gilbert-Elliot model measures independency between successive events, the Runs-Test measures the randomness of a Bernoulli sequence, and the Chi-square test for goodness of fitness measures the fitness of a Bernoulli sequence to a certain (we check Bernoulli distribution) distribution. Each of these methods evaluates a specific characteristic of the link's historical transmission events. From the tests mentioned above, we derive or select a suitable metric to measure the independence property that supports *link boosting* capability in run time. For offline analysis and comparing the measurement quality of the metric, we use $\beta$-metric [Sri+08] as a reference. Next, we explain each test in detail.

### 3.3.1 Gilbert-Elliot Model

Gilbert-Elliot model is a discrete-time 2-state Markov chain model introduced by Gilbert in [Gil60] to model error-free burst-noise binary channels. Later, Elliot [Ell63] extends this model by considering errors in both states. Figure 3.1 shows a 2-state Gilbert-Elliot model.

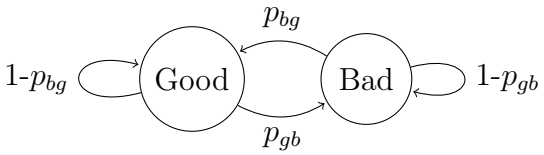



Figure 3.1: A 2-state discrete time Markov chain model

It is extensively used for modeling different problems in communication systems. In [Sri10], the authors mention the use-case of *channel memory* $\mu$ parameter calculated from transition probabilities of Gilbert-Elliot (GE) model. To explain $\mu$, and for further explanation, we borrow the terminologies used to denote state transition probabilities in [Sri10]. Transition probability from good state to bad state and from bad state to good state is defined as $p_{gb}$ and $p_{bg}$, respectively. I.e.,

$$p_{gb} = P(s_t = Bad \mid s_{t-1} = Good)$$
$$p_{bg} = P(s_t = Good \mid s_{t-1} = Bad)$$

where $s_t$ is the state at time $t$.

From these transition probabilities, *channel memory* $\mu$ is a calculated as

$$\mu = 1 - p_{gb} - p_{bg} \tag{3.2}$$

A value 0 and 1 for $\mu$ implies that the link (outcome of transmission events) is independent and dependent (bursty), respectively. A value close to -1 implies the link oscillates between good and bad state. I.e., the degree of statistical independence is expressed by $\mu$ on a scale [-1,1]. In [Sri10], the authors compared $\mu$ with the $\beta$-metric and found them similar. They prefer $\beta$-metric over $\mu$ due to the former's ability to capture very long-term dependencies.

Consider the following three sequences, (i) 11011110010010001101, (ii) 1010101010101010, and (iii) 11111000000000011111 with a length of 20 and a success ratio of 0.5. The starting state of the model is the outcome of the first event, i.e., if the first event is a *success* event, the *Good* state is the starting state, and vice versa. For instance, for sequence (i), the starting state is the *Good* state since the first transmission outcome is successful (a *success* event). Further, the second transmission outcome is also successful. Thus, a transition from *Good to Good*. The given sequences with a length of 20 have 19 transitions. Sequence (i) has 5 *Good to Good* and *Good to Bad*, 4 *Bad to Bad*, and 5 *Bad to Good* transitions. The $p_{gb}$ and $p_{bg}$ of the sequence (i) is 0.5 ($\frac{5}{10}$) and 0.55 ($\frac{5}{9}$), respectively. The $\mu$ value of sequence (i) is calculated as -0.055 (close to 0). Similarly, the $\mu$ values of

the sequences (ii) and (iii) equal -1 and 0.8, respectively. I.e., the sequences (i), (ii), and (iii) are independent, oscillatory, and dependent, respectively and it is reflected in the $\mu$ value.

In [Bil+15], the authors reuse the 2-state GE model and the *channel memory* parameter $\mu$ to estimate link quality in wireless sensor networks. They use stationary probabilities for their estimations. For this, they represented the transmission events on a link as Bernoulli trials. A trial that has a $\mu$ value close to 0 is considered as a *memory-less* process (i.e., independent frame loss), which is similar to the representation in [Sri10].

Considering the results from the works mentioned above, calculating the *channel memory* $\mu$ value fits our requirement to measure short-term dependencies. This means it can be used to measure the degree of independence between the outcome of transmission events. In our experiments, we continuously calculate $\mu$ value for a specific history and measure the correlation between *link boosting* and $\mu$ value. This helps to answer Question I.a. Depending on the results, we use the *channel memory* $\mu$ parameter for link selection.

### 3.3.2 Runs-Test

Wald–Wolfowitz Runs-test [Bra68] is a non-parametric statistical test to check whether a given sequence (binary value sequences) is generated by a random process. It is a *statistical hypothesis test* to check whether a given sequence is generated by a random process depending on the number of runs. Following are the test hypotheses:

$$H_O : \text{Events are drawn from random process}$$
$$H_1 : \text{Events are not drawn from random process}$$

A *run* is defined as the continuous streak of the same events. I.e., for a link, a run is a continuous streak of successful (of 1's) or failed (of 0's) transmissions. Based on the difference between observed and expected number of *runs*, and the standard deviation, the test statistic $z$-value for the Runs-test is calculated.

$$z = \frac{r - \bar{r}}{\sigma_r} \tag{3.3}$$

where $r$ is the observed number of runs, $\bar{r}$ is the expected number of runs, and $\sigma_r$ is the standard deviation of the number of runs.

For a test statistic $z$-value above 1.96 and below -1.96 (i.e., $\mid z \mid > 1.96$), the Runs-test rejects the null hypothesis with a 5% significance level [7]. In our experiments, we calculate the p-value for the test and use a confidence interval $\alpha = 0.05$. The p-value is calculated for a two-tailed test, i.e., 0.025 on both side of the tail distribution. Thus, the Runs-test tests the hypothesis whether a given sequence is generated by a random process.

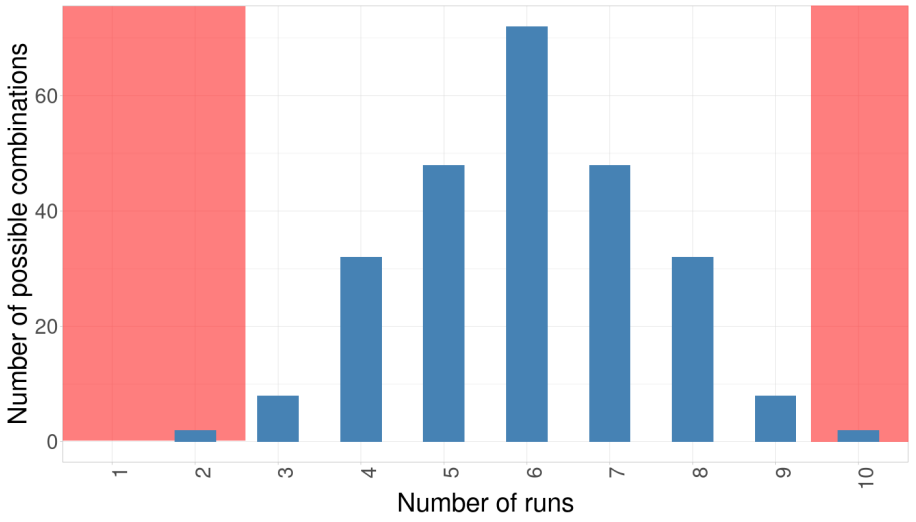



Figure 3.2: Number of possible Bernoulli sequence combinations of length 10 and success probability p = 0.5. The x-axis represents the number of runs in a Bernoulli sequence. The y-axis represents the number of possible combinations with the respective number of runs. The shaded region is the rejection area, for alpha = 0.05.

For example, consider a Bernoulli trial of length 10, i.e., sample size $N = 10$, with a success probability of 0.5 (50%), 5 successful and 5 failed events. For this example (trial of length 10), there are at most 252 $\left(\binom{10}{5}\right)$ possible sequences with different arrangements. Figure 3.2 shows the number of trials containing runs of length 2 to 10. Out of these 252 possible combinations, for instance, there are two possible ways to get a sequence containing 2 or 10 runs, namely 1111100000, 0000011111 (with 2 runs), 1010101010, or 0101010101 (with 10 runs). Under the null hypothesis that success (1's) and failure events (0's) occur randomly, the probability for a sequence with extreme number of runs to occur (i.e., a sequence with 2 or 10 runs) is $\frac{4}{252} = 0.0158$, i.e., the four extreme cases with 2 or 10 runs in a sequence.

As mentioned in the previous test (channel memory $\mu$ value), links are categorized as *oscillatory*, *bursty*, and *statistically independent*. In our analysis, we use the results from Runs-test to evaluate different types of link. For instance, in the

[7] Thus, a $z$-value above 1.96 for a sample indicates over shuffling, and below -1.96 indicates under shuffling.

aforementioned example, sequences with 2 runs are *bursty* and with 10 runs are *oscillatory* in nature. We will discuss more in Sec. 3.4. The Runs-test is used to find answer for our Question I.b. The p-value calculated from the test is used as the evidence (not probability) for considering the null hypothesis, i.e., the degree of random frame collisions. For p-values above the confidence interval $\alpha = 0.05$, the higher the p-value, the stronger the evidence for randomness. During initial experiments, we periodically conduct Runs-test for a specific history, and measure the correlation between *link boosting* and test result.

### 3.3.3 Chi-square test

"Chi-square goodness-of-fit test" is a non-parametric test that tests whether a continuous set of events are drawn independent and identically distributed (i.i.d) from a specific distribution with a success probability $p$. We use this test to check whether the outcome of transmission events are i.i.d drawn from a Bernoulli distribution.

$H_O$ : Events are drawn i.i.d from Bernoulli distribution with success probability $p$
$H_1$ : Events are not drawn i.i.d from Bernoulli distribution with success probability $p$

The test is conducted on a Bernoulli sequence of length $N$. Following is the test procedure:

1. Divide the sequence of length $N$ into sub-sequences $a_1, a_2, ..., a_S$ of size $k$. It results in $\frac{N}{k}$ sub-sequences.
2. Count the number of success events in the whole sequence and calculate $p = (\frac{\text{number of success events}}{\text{total number of events}})$.
3. Similarly, calculate the number of success events for each $\frac{N}{k}$ sub-sequences. This results in $\frac{N}{k}$ natural numbers between 0 and $k$.
4. Apply the Chi-square test to test whether the natural numbers we are observing correspond to a histogram of $\frac{N}{k}$ pulls from a Bernoulli distribution $B(k, p)$.
5. The Chi-square test calculates the Chi-square value. From the calculated Chi-square value, the p-value for the test is calculated.

Fig. 3.3 shows the observed and expected histogram pulls of the number of successful events in sub-sequences of length $k = 30$ from a sequence with a length of $N = 600$

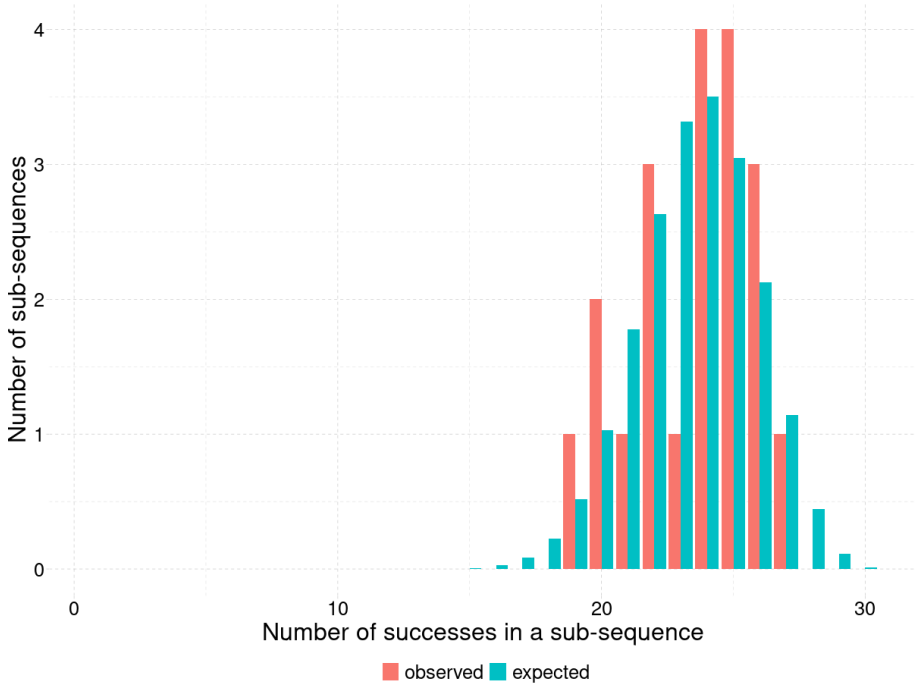



Figure 3.3: Histogram showing the observed and expected number of pulls for the number of successful events on a sequence. Sequence and sub-sequence length of $N = 600$ and $k = 30$, respectively and a success probability of 0.8.

and a success probability of 0.8. It is sub-divided into $S = 20$ sub-sequences, i.e., the length of the sub-sequence is $k = 30$. The Chi-square test is conducted to check whether the observed number of success events in all sub-sequences corresponds to a histogram of 20 pulls from a Bernoulli distribution $B(30, 0.8)$. The Chi-square value of the test is 5.042999. For a degree of freedom of 19 ($\frac{N}{k} - 1$), the p-value is 0.98, and it fails to reject the null hypothesis.

The Chi-square test has some limitations. The test result is sensitive to the length of the sequence. To minimize this effect, we apply the test on a moving window. Another disadvantage is that the order of events is not considered in the test. In other words, when calculating the success probability P(X==1) of each sub-sequence, the calculation does not consider the randomness of the sub-sequence.

We use the Chi-square test to find an answer to our Question I.c. Depending on the p-value of Chi-square test and the selected confidence interval, null-hypothesis ($H_0$) is either accepted or rejected. As mentioned earlier, the p-value of the test depends on the Chi-square value. The Chi-square value is a measure of the difference between observed and expected data distribution. I.e., the larger the Chi-square value, the larger the deviation from the null hypothesis.

For our experiments, we select a confidence interval of 95% or $\alpha = 0.05$. With a 5% significance level, we can accept the null hypothesis for a Chi-square value below the boundary value. To determine the degree of fitness with a 5% significance level, we normalize the evidence of the null hypothesis on a [0,1] scale. It is calculated as,

$$chi_q = 1 - \frac{\min(\mathrm{X}^2_{dof,table}, \mathrm{X}^2_{dof,obs})}{\mathrm{X}^2_{dof,table}} \tag{3.4}$$

where $X^2_{dof,obs}$ is the observed Chi-square value with a degree of freedom $dof$, and $X^2_{dof,table}$ is Chi-square boundary with a degree of freedom $dof$ to reject the null hypothesis. A higher value of $chi_q$ shows a stronger evidence for the establishment of the null hypothesis.

## 3.4 Experimental Assessment

To evaluate the collision property in different channel conditions, we assess data collected from different experiment settings. In our evaluation, we use data collected from three different experiments. It includes

1. synthetic link data,
2. real link data in a collision-free environment, and
3. real link data in an environment with foreign traffic (testbed data)

We apply the statistical tests to the data collected. A $\mu$ value of 0 indicates that the sequence is independent or bursty, 1 as dependent, and -1 as oscillatory. The Runs-test tests the hypotheses,

$$H_O : \text{Events are drawn from random process}$$
$$H_1 : \text{Events are not drawn from random process}$$

and the Chi-squares tests the hypotheses,

$$H_O : \text{Events are drawn i.i.d from Bernoulli distribution with success probability } p$$
$$H_1 : \text{Events are not drawn i.i.d from Bernoulli distribution with success probability } p$$

### 3.4.1 Experiment Setup

In Sec. 3.4.2, we evaluate synthetic link data. We use built-in functions of the statistical programming language R [R C17] to generate a statistically independent

Bernoulli sequence. Bursty and oscillatory sequences are synthesized using the 2-state Markov model. We synthesize by setting a $\mu$-value for the entire sequence; however, the $\mu$-value may differ for sufficiently short subsequences. In Sec. 3.4.3 and Sec. 3.4.4, we evaluate the raw data collected from the experiments offline. The experiments are conducted on nodes using the WiPS protocol stack mentioned in the previous chapter.

A WiPS frame contains the WiPS node address *snd* of the sender, and its transmission counter value *txCtr*. A node increases its transmission counter value *txCtr* for each transmission, including retransmissions. For instance, consider a node $v$ at runtime where the last frame sent to $v$' from node $v$ has a transmission counter *txCtr* value of 10. When a new frame $f$ destined to node $v$' arrives at node $v$, the WiPS MAC layer assigns 11 as the value for *txCtr*. Suppose that the frame $f$ fails the first transmission attempt. When the node attempts to retransmit the frame $f$ (if it does not exceed the maximum retransmission limit), the node increments its transmission counter value *txCtr* to 12 and updates the *txCtr* of frame *f*. A node that receives the frame adds its WiPS node address *rcv* and its local timestamp value *rxTs* when the frame arrived in the user stack. Thus, for each successfully received frame $f_{rx}$, we record a tuple $f_{rx} = (snd, rcv, txCtr, rxTs)$. From these raw data, we derive the Bernoulli sequence for a link.

Tab. 3.1 shows a sample raw data for a link and the respective Bernoulli sequence. The table shows the successfully received frames on the link. Columns with headers *snd* and *rcv* represent the sender and receiver WiPS node address of the frame, respectively. For the sample data shown in the table, the sender address *snd* is 10 and receiver node address *rcv* is 11. Column *txCtr* represents the sender transmission counter value during the frame transmission. We derive the Bernoulli sequence from the received *txCtr* values. Starting from $txCtr = 1$, we map a successfully received value to 1 and missing value/s to 0. In this example, frames with *txCtr* value 4, 7, 9, 10, 11, and 16 are missing. These failed transmission events are marked as 0 in their respective positions (starting from left) in the corresponding Bernoulli sequence. The final column *rxTs* represents the local Unix timestamp at the receiver node when a frame reaches the user stack of the receiver node.

We structure the Bernoulli sequence of link by constructing Bernoulli slots of equal size $s_{Bs}$, consisting of a sequence of $s_{Bs}$ Bernoulli trials. This would correspond to PDR slots defined in [MG21], with the difference that length is not measured in time, but in the number of Bernoulli trials. These slots are consecutively numbered, starting with 1 and represented as $n_{Bs} \in \mathbb{N}_+$. The size of the Bernoulli slot $s_{Bs}$ denotes the number of transmission event outcomes in a slot. A Bernoulli slot $n_{Bs}$

| *snd* | *rcv* | *txCtr* | *rxTs* (ms) |
|---|---|---|---|
| 10 | 11 | 1 | 1619189954100 |
| 10 | 11 | 2 | 1619189954200 |
| 10 | 11 | 3 | 1619189954700 |
| 10 | 11 | 5 | 1619189955230 |
| 10 | 11 | 6 | 1619189955400 |
| 10 | 11 | 8 | 1619189955980 |
| 10 | 11 | 12 | 1619189957100 |
| 10 | 11 | 13 | 1619189957200 |
| 10 | 11 | 14 | 1619189957560 |
| 10 | 11 | 15 | 1619189957900 |
| 10 | 11 | 17 | 1619189959300 |

**Bernoulli sequence for the corresponding raw data**
1 1 1 0 1 1 0 1 0 0 0 1 1 1 1 0 1

Table 3.1: A raw data sample for the link 10→11 logged at node 11 and its corresponding Bernoulli sequence

for each potential link $e$ consists of the transmission outcomes of frames with *txCtr* value between $s_{Bs} \cdot (n_{Bs} - 1) + 1$ and $s_{Bs} \cdot n_{Bs}$. For instance, a Bernoulli slot $n_{Bs} = 2$ with size $s_{Bs} = 30$ consists of the transmission outcome of frames with *txCtr* value from 31 to 60.

We use the conventional link quality metric Packet Delivery Ratio (PDR) to measure the link quality. Instead of measuring the PDR of a link for the entire experiment data, we calculate the PDR during a specific number of events that are structured into moving windows. A moving window consists of $s_{mw}$ Bernoulli slots (similar as in [MG21]). Like Bernoulli slots, moving windows are consecutively numbered, starting with 1. $n_{mw}$ and $s_{mw}$ denote the number and size of the moving window, respectively. We build moving windows of size $s_{mw} \geq 1$, consisting of a fixed number $s_{mw}$ of consecutive Bernoulli slots. Moving window $n_{mw}$ comprises of Bernoulli slots $n_{Bs}$, $n_{mw} - s_{mw} + 1 \leq n_{Bs} \leq n_{mw}$. This way, for $n_{mw} \geq s_{mw}$, the numbering of Bernoulli slots and moving windows are aligned. For each moving window $n_{mw}$ and each potential link $e$, the PDR $pdr^{e}_{raw}(n_{mw})$ (called as raw PDR) is calculated based on collected data. In this thesis, when we refer to the PDR of the link, it is the PDR calculated from the raw data, i.e., without any retransmissions and duplicates. It is calculated as

$$pdr^{e}_{raw}(n_{mw}) = \begin{cases} \dfrac{n^{e}_{rx}(n_{mw})}{n^{e}_{tx}(n_{mw})}, & \text{if } n_{mw} \geq s_{mw} \\ \\ \text{undefined}, & \text{if } n_{mw} < s_{mw} \end{cases} \tag{3.5}$$

where, $n^{e}_{rx}(n_{mw})$ is the number of receptions, and $n^{e}_{tx}(n_{mw})$ is the number of transmission attempts on link $e$ in moving window $n_{mw}$.

Let $F_{rx}$ be the set of frames received during an experiment, $e = (v, v')$ be a link, $n_{Bs}$ be a Bernoulli slot number. Then, the set of frames $F^{e}_{rx}(n_{Bs}) \subseteq F_{rx}$ received on link $e$ during a Bernoulli slot $n_{Bs}$ is derived as:

$$\begin{aligned} F^{e}_{rx}(n_{Bs}) = &\{f \in F_{rx} \mid f.snd = addr(v) \wedge f.rcv = addr(v') \\ &\wedge (s_{Bs} \cdot (n_{Bs} - 1) + 1) \leq f.txCtr < (s_{Bs} \cdot n_{Bs})\} \end{aligned}$$

The number of receptions $n^{e}_{rx}(n_{mw})$ and transmission attempt $n^{e}_{tx}(n_{mw})$ on link $e$ during moving window $n_{mw}$ are derived as:

$$n^{e}_{rx}(n_{mw}) = \begin{cases} \sum\limits_{n_{Bs}=n_{mw}-s_{mw}+1}^{n_{Bs}=n_{mw}} \mid F^{e}_{rx}(n_{Bs}) \mid, & \text{if } n_{mw} \geq s_{mw} \\ \\ \text{undefined}, & \text{if } n_{mw} < s_{mw} \end{cases} \tag{3.6}$$

$$n^{e}_{tx}(n_{mw}) = \begin{cases} s_{mw} \cdot s_{Bs}, & \text{if } n_{mw} \geq s_{mw} \\ \\ \text{undefined}, & \text{if } n_{mw} < s_{mw} \end{cases} \tag{3.7}$$

We apply each statistical test to each moving window to assess short-term independence between event outcomes. Our primary interest is to measure the short-term characteristics and methods to measure them. To measure long term dependencies, we apply $\beta$-metric to the entire sequence. As mentioned earlier (in Sec. 3.2), $\beta$-metric requires a large data set for a reliable result [Teg20], has high computational complexity and hence, is not a suitable metric for online evaluation. It is not a viable metric to evaluate short-term dependency [Ali+11]. We use $\beta$-metric to understand any possible long-term dependencies.

The findings from each experiment are structured as follows. After structuring the data of each link into moving windows, we apply the aforementioned statistical tests (see Sec. 3.3) to each moving window, i.e. measure the GE channel memory value $\mu$, perform the Runs-Test, and perform the Chi-square test. We also investigate for any

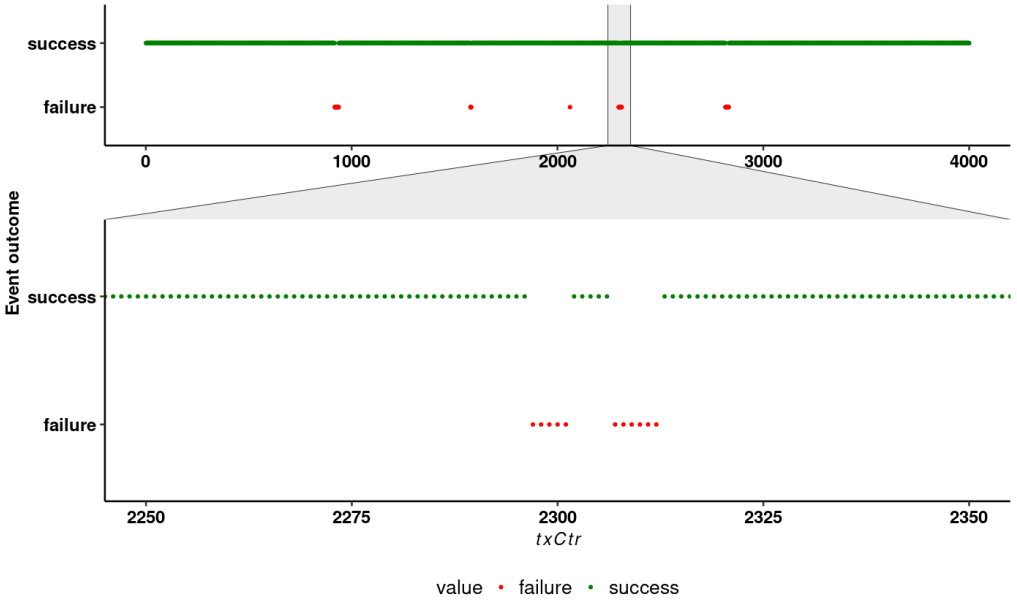
possible correlation between the results of each statistical test. Then, we evaluate the influence of using a different size $s_{mw}$ for moving windows. Before concluding our findings, we apply the $\beta$-metric to the entire link or sequence to check for any long-term dependencies. We explain our findings from analysing synthetic data, real link data in a collision-free environment, and real link data in an environment with foreign traffic (testbed data) in Sec. 3.4.2, Sec. 3.4.3, and Sec. 3.4.4, respectively.

## 3.4.2 Synthetic data

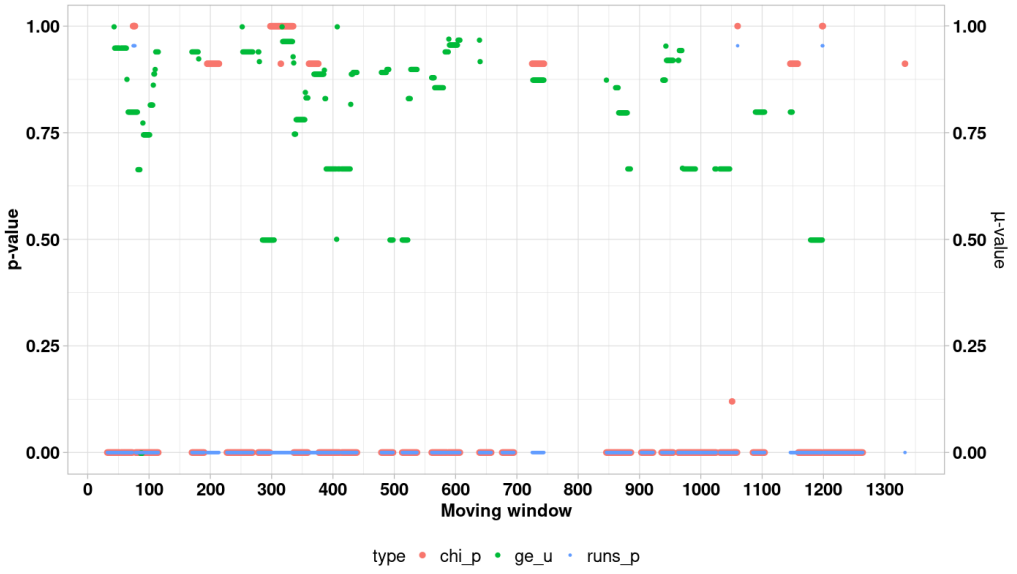
In this section, we conduct our statistical tests on the synthetic data. These experiments are conducted to examine the ground truth. I.e., we synthesize data of known types (statistically independent, bursty, or oscillatory event outcomes) and conduct statistical tests to observe whether tests provide expected results. For instance, on a synthesized statistically independent Bernoulli sequence, we expect most of the sub-sequence in a moving window to have statistically independent and random event outcomes that fit a Bernoulli distribution. Further using synthesized data, we check whether they yield the expected results in extreme cases ($\mu$ = 1, 0, -1), and calibrate the Bernoulli slot and moving window sizes.

We start with analysing three elementary Bernoulli sequences of type *bursty*, *statistically independent*, and *oscillatory* with a length of 40000. Bursty and statistically independent Bernoulli sequences have a success ratio (or raw PDR), $p = 0.99$. The oscillatory Bernoulli sequence has a success ratio, $p = 0.5$. Bursty and oscillatory sequences of length 40000 are synthesized with the Markov model. The $\mu$-values for the bursty and oscillatory sequences are close to 0.89 and -1, respectively. The synthesized bursty sequence has a $p_{bg}$ value of 0.105 (state transition probability from bad to good state) and a $p_{gb}$ value of 0.001 (state transition probability from good to bad state). The oscillatory sequence has a $p_{bg}$ and a $p_{gb}$ value of 1, i.e. an elementary sequence with alternative 1's and 0's. As stated earlier, the $\mu$-values may differ for sufficiently short sub-sequences. This, too, applies to the statistically independent Bernoulli sequences synthesized with built-in R functions.

First, let us look into results for moving windows with the following configuration, $s_{Bs} = 30$ and $s_{mw} = 20$. This configuration results in a total of 600 Bernoulli trials or event outcomes in a moving window. The Chi-square test uses a sub-sequence size of $k = 30$. It results in 20 (600/30) sub-sequences for the Chi-square test in a moving window. Fig. 3.4, Fig. 3.5, and Fig. 3.6 show the results of the bursty, statistically independent and oscillatory sequences, respectively. The first sub-figures, Fig. 3.4a, Fig. 3.5a, and Fig. 3.6a show the event outcomes of the first 4000 events. The second sub-figures, Fig. 3.4b, Fig. 3.5b, and Fig. 3.6b show the statistical test



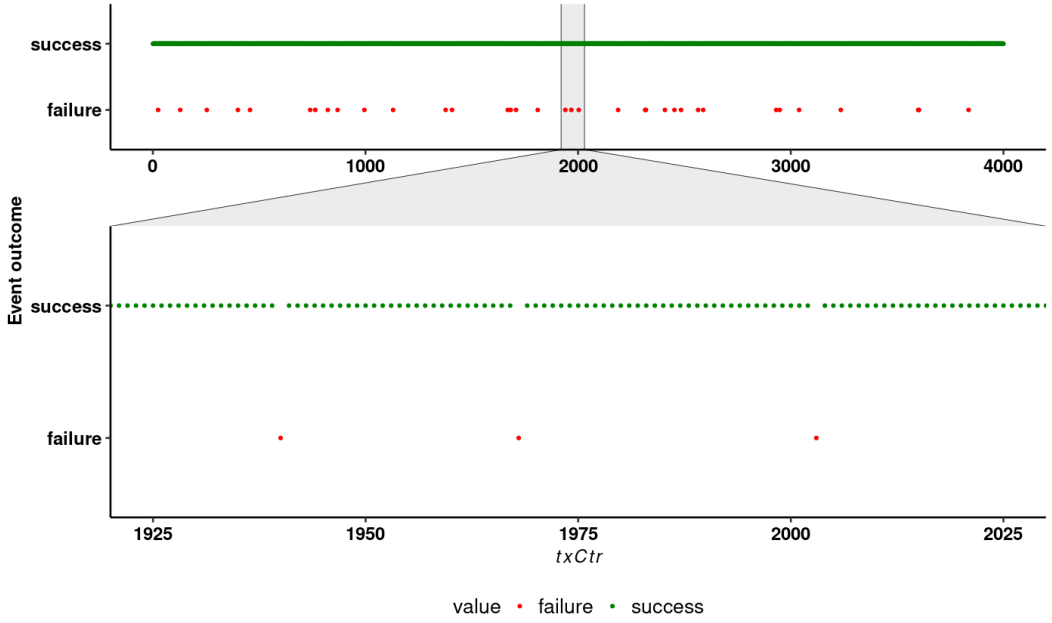

(a) First 4000 trials of the bursty Bernoulli Sequence and magnifying a region where failed events occurred.



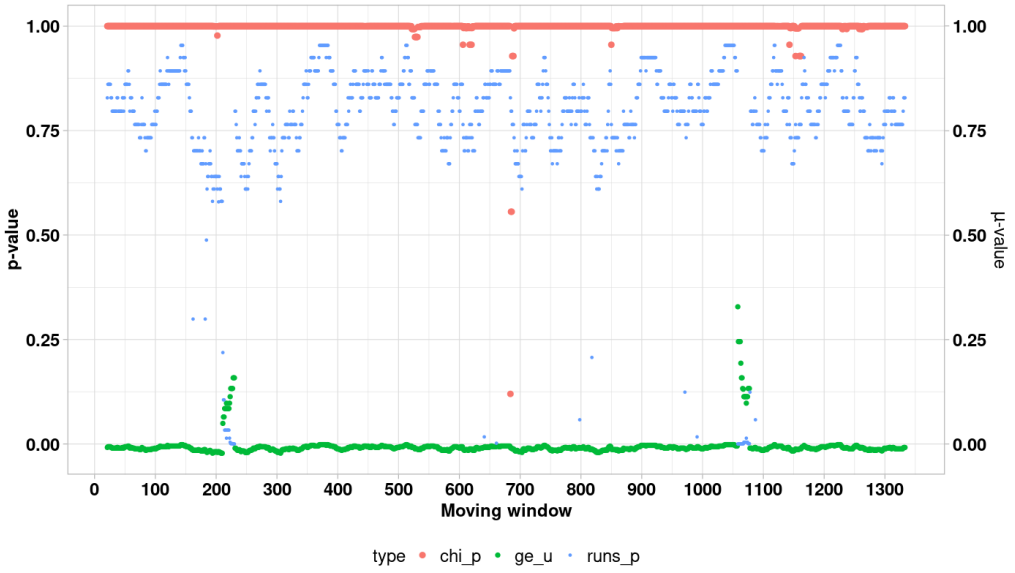

(b) Results of statistical methods applied to moving windows

Figure 3.4: A bursty Bernoulli Sequence with raw PDR = 0.99. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$.

results in a moving window. In statistical test results, the red coloured and the blue coloured dot represent the p-value for the Chi-Square test and the Runs-test, respectively. The green coloured dot represents the *channel memory* $\mu$ value during the moving window. In the plot, there are moving windows with no dots. In those windows, there are not enough event outcomes to conduct the test. For instance, to




(a) First 4000 trials of the statistically independent Bernoulli Sequence and magnifying a region where failed events occurred.

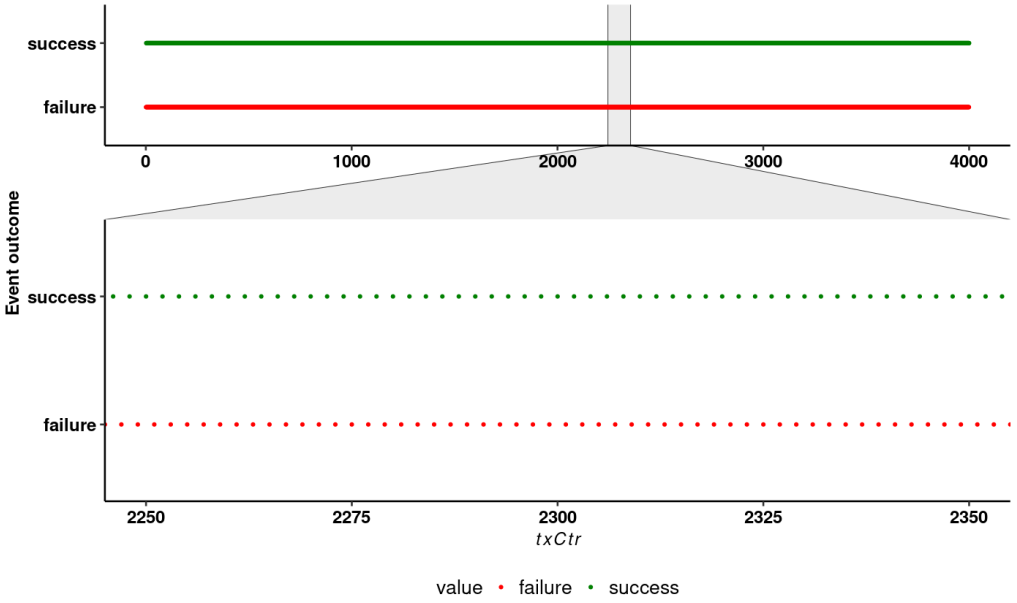



(b) Results of statistical methods applied to moving windows

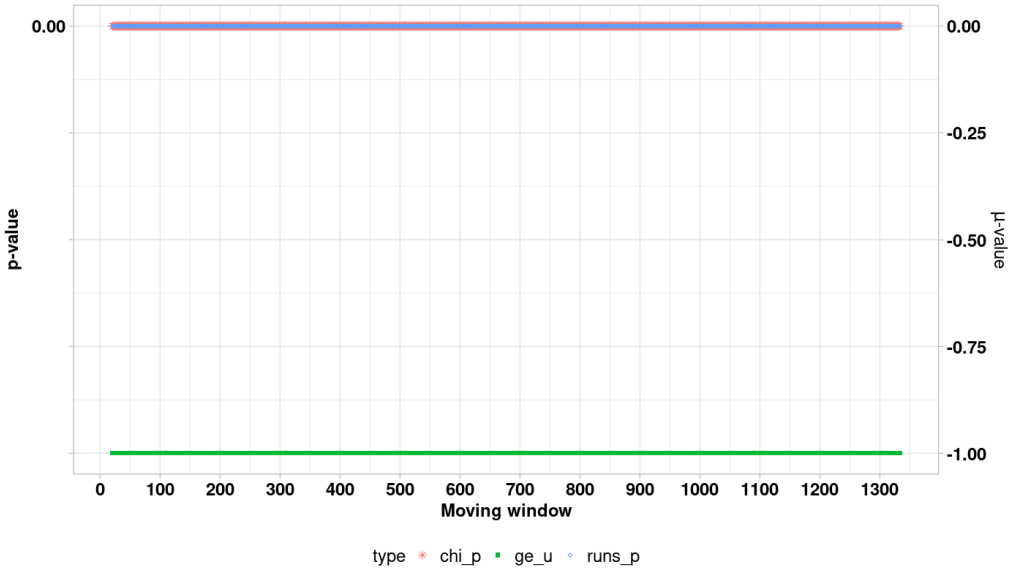
Figure 3.5: A statistically independent Bernoulli Sequence with raw PDR = 0.99. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$.

calculate the $\mu$ value, the sub-sequence must contain at least one event from good to bad (a loss following a success) and from bad to good (a success following a loss).

Fig. 3.4 shows the results for a bursty Bernoulli sequence. The $\mu$ value for the whole sequence is 0.89. However, for the sub-sequences in a moving window, it




(a) First 4000 trials of the oscillatory Bernoulli Sequence and magnifying a region where failed events occurred.




(b) Results of statistical methods applied to moving windows

Figure 3.6: An oscillatory Bernoulli Sequence with raw PDR = 0.5. Statistical methods are applied to moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$.

differs. For the vast majority of moving windows, the $\mu$ values are above 0.70. Thus, we can conclude that the vast majority of sub-sequences are also tested as bursty or statistically dependent similar to the entire sequence. Similarly, non-parametric tests such as the Runs-test and the Chi-square reject their null hypothesis for a majority of moving windows. I.e., the Runs-test and the Chi-square test show

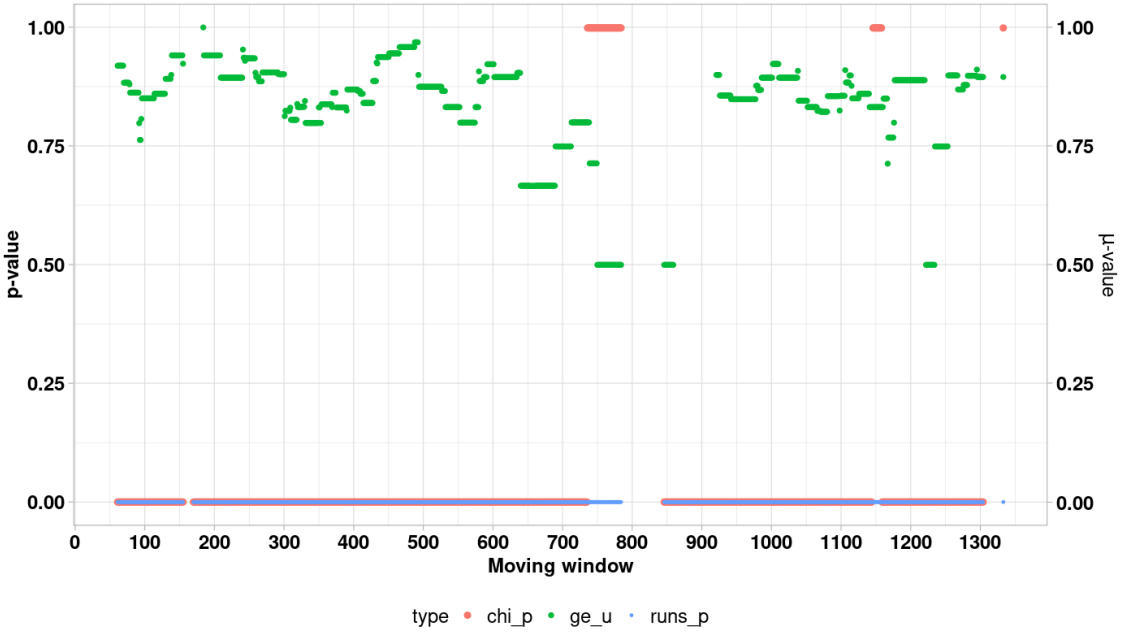
substantial evidence to reject randomness claim and fitness to Bernoulli distribution, respectively, with a 95% confidence interval ($\alpha = 0.05$). There are seldom windows (for instance, between 70-100) where the $\mu = 0$, indicating the sub-sequences in those windows has independent loss events. Likewise, the Runs-test and the Chi-square failed to reject their null hypothesis for a few sub-sequences. It shows that even though the entire sequence is bursty, few sub-sequences might have a different characteristic depending on the sub-sequence length.

Fig. 3.5 shows the results for a statistically independent Bernoulli sequence. For a vast majority of moving windows, the $\mu$ value is close to 0 providing strong evidence for statistical independence. Seldom, the $\mu$ value jumped from 0. The Runs-test and the Chi-square test results also fit the ground truth. Agreeing with the $\mu$ value, the p-values of the non-parametric tests failed to reject the null hypothesis for the vast majority of moving windows. There are few moving windows where they reject their null hypothesis. Another noticeable result is the correlation between different test results. In those few moving windows where the $\mu$ value jumped from 0, the p-values of the runs-test also reacted to it. I.e., when the $\mu$ value shows some signs of dependency in event outcomes in a sub-sequence of a moving window, the Runs-test also rejected the null hypothesis by showing strong evidence. For instance, in Fig. 3.5b, between moving windows 1050 and 1100. We investigate it further in the following analyses.

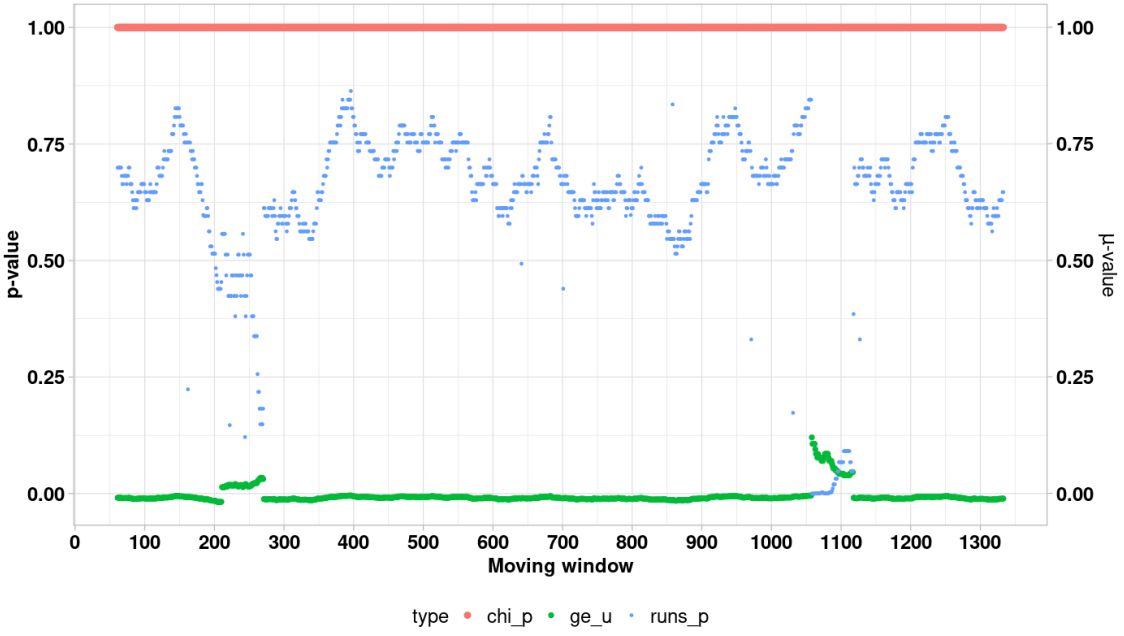
Fig. 3.6 shows the result for an oscillatory sequence. The whole sequence has a $\mu$ value of -1. The non-parametric test results also convey the ground truth. The Chi-square test and the Runs-test reject the null hypothesis for all sub-sequences, i.e., the sub-sequences are neither random nor fit a Bernoulli distribution with the observed success probability of the sub-sequence.

Next, we analyze those sequences for moving windows with a higher size $s_{mw} = 60$. The sizes of the Bernoulli slots and sub-sequence size for the Chi-square test remain the same. In this configuration, each moving window consists of 1800 trials. For the Chi-square test, a sub-sequence size $k = 30$ results in 60 (1800/30) sub-sequences for the test.

Fig. 3.7a shows the result for the bursty sequence using a larger moving window. In comparison to Fig. 3.4b, there are fewer moving windows without results (no dots). In this sample sequence with a success probability of 0.99, increasing the window size reduced the number of windows without test results. Another noticeable effect is that increasing the sample size results in more averaged result. For instance, consider the moving windows from $n_{mw} = 30$ to $n_{mw} = 200$ in Fig. 3.4b and the moving windows from $n_{mw}$ 100 to $n_{mw} = 200$ in Fig. 3.7a. In those regions, using a

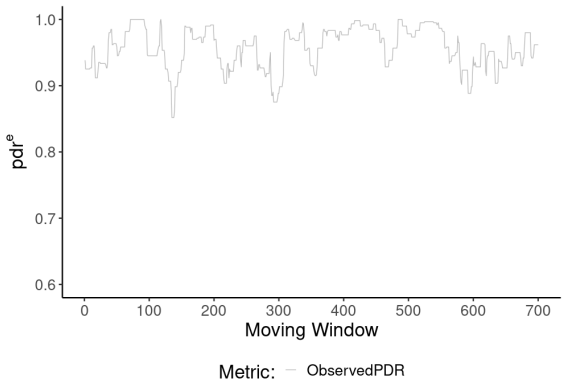



(a) Results of bursty Bernoulli sequence (Fig. 3.4)

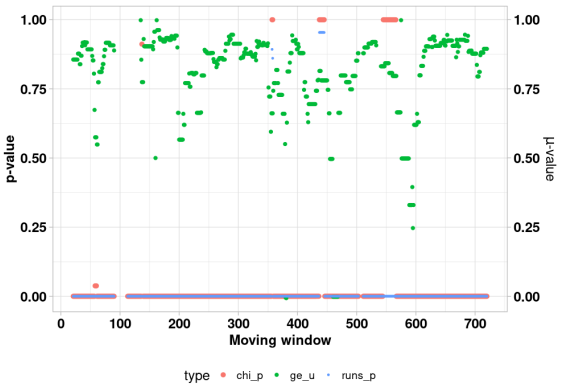



(b) Results of statistically independent Bernoulli sequence (Fig. 3.5)

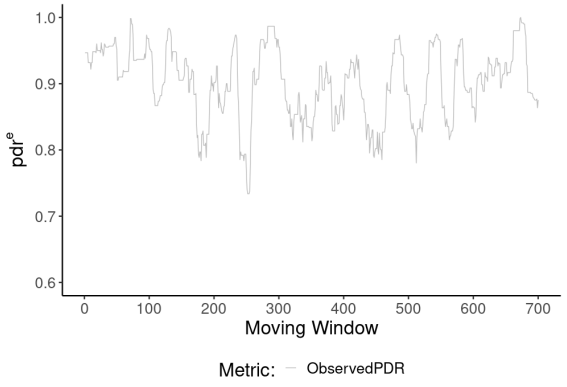
Figure 3.7: Statistical methods are applied to a different moving configuration $s_{mw} = 60$ and $s_{Bs} = 30$ to the sequences used in Fig. 3.4, and Fig. 3.5



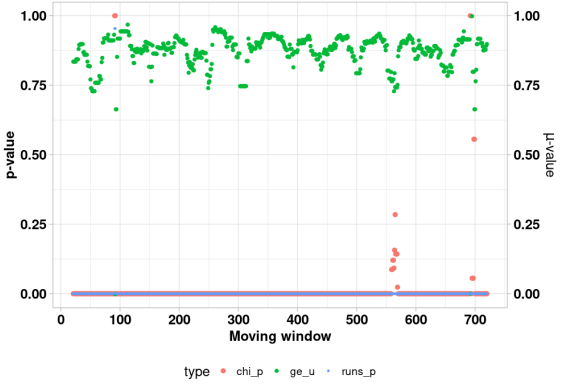

(a) PDR over the windows (sequence 1)



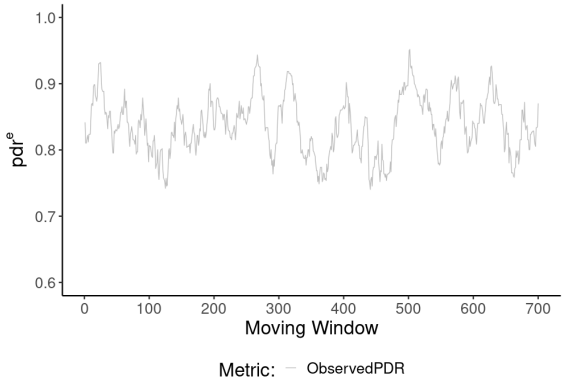

(b) Test results of sequence 1



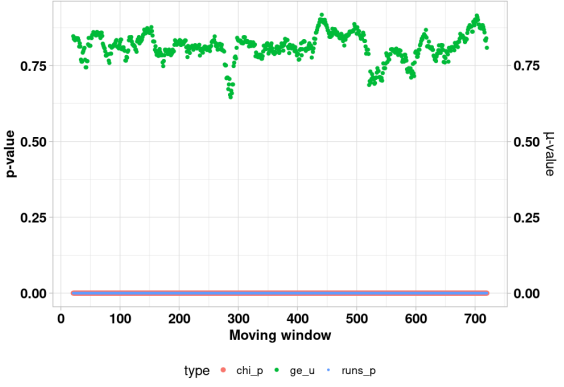

(c) PDR over the windows (sequence 2)




(d) Test results of sequence 2

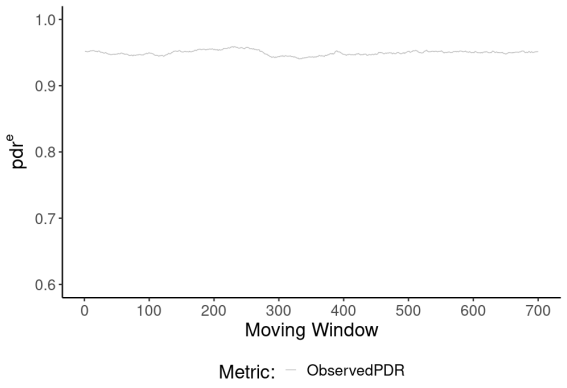



(e) PDR over the windows (sequence 3)

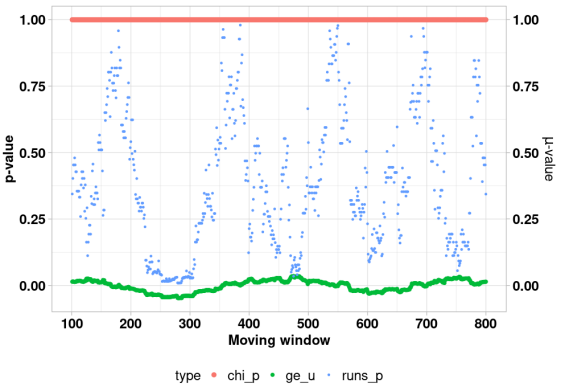



(f) Test results of sequence 3

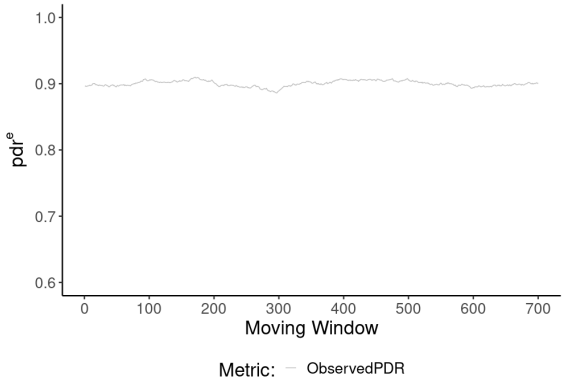
Figure 3.8: Sample bursty (statistically dependent) sequences with different success probabilities $pdr^e_{raw}$ generated using 2-state Markov model and test results with moving windows of size $s_{mw} = 20$ and $s_{Bs} = 30$.



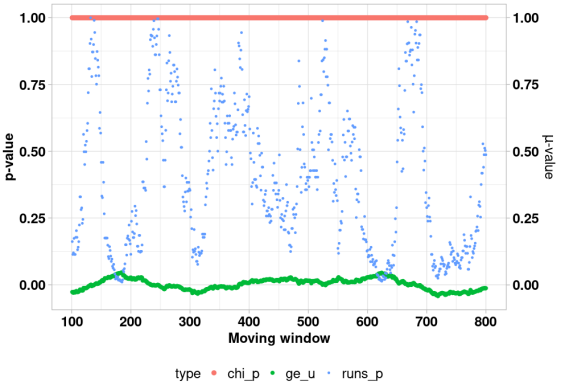

(a) PDR over the windows (sequence 1)



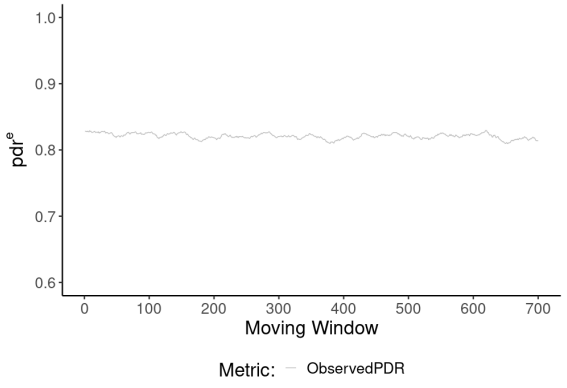

(b) Test results of sequence 1



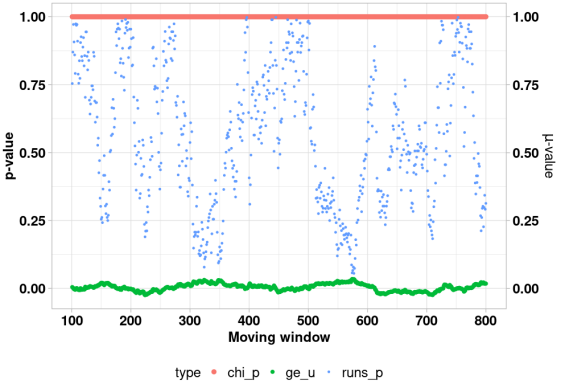

(c) PDR over the windows (sequence 2)




(d) Test results of sequence 2




(e) PDR over the windows (sequence 3)




(f) Test results of sequence 3

Figure 3.9: Sample statistically independent sequences with different success probabilities $pdr^e_{raw}$ generated using built-in R methods and test results with moving windows of size $s_{mw} = 100$ and $s_{Bs} = 30$.

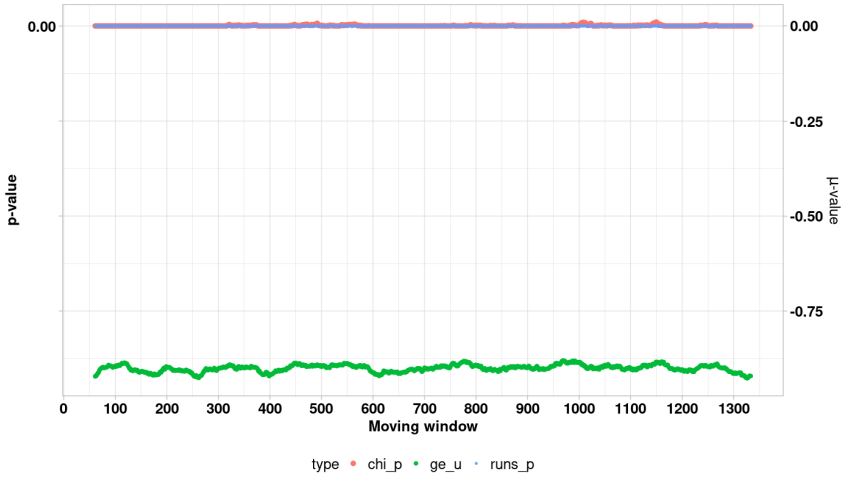
smaller window size results in a $\mu$ value varying from 0.65 to 0.99. By increasing the window size, the $\mu$ values averages over the window. The results of non-parametric tests have a similar effect. The Runs-test shows strong evidence to reject the null hypothesis, and so does the Chi-Square test.

We observe similar effects when applying statistical tests to the statistically independent and oscillatory sequences. When we increased the window size for statistically independent sequence, the short spikes in the $\mu$ value average during windows between 200 to 250 and 1050 to 1100 (see Fig. 3.5b and Fig. 3.7b). The correlation between $\mu$ values and the test result of the Runs-test is better visible in Fig. 3.7b. I.e., between moving window 1050 to 1100 where the $\mu$ value jumped from 0, the Runs-test rejects the null hypothesis. For the elementary oscillatory sequence, increasing window size did not have any impact since, in each moving window, the pattern is the same, i.e., alternating 1's and 0's.

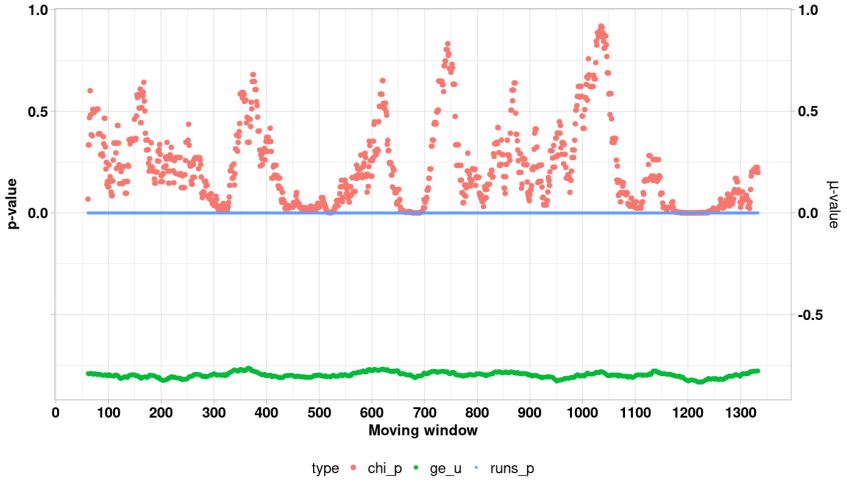
Next, we calculate the $\beta$-metric [Sri+08] value for the entire sequences to identify any possible long term dependencies. A $\beta$-metric value close to region 1 indicates statistically dependent, and close to 0 indicates statistically independent links. The $\beta$-metric of the bursty sequence is 0.99, matching the ground truth. For statistically independent and oscillatory sequences, the values of $\beta$-metric are -0.5 and -3, respectively. As mentioned earlier, the negative region of the $\beta$-metric is not limited, i.e., a link can have $\beta$-metric value -1, -5 or even -10. So a negative $\beta$ value does not provide much information, rather the presence of a negative correlation and the higher the negative $\beta$ value, the higher the negative correlation.

We evaluated several other synthesized data. Fig. 3.8 shows the results of different generated bursty Bernoulli sequences. Figures on the left and right side show their raw PDR values (success ratio $p$) in each moving window and the test results, respectively. Fig. 3.9 shows the results of different synthesized statistically independent Bernoulli sequences.

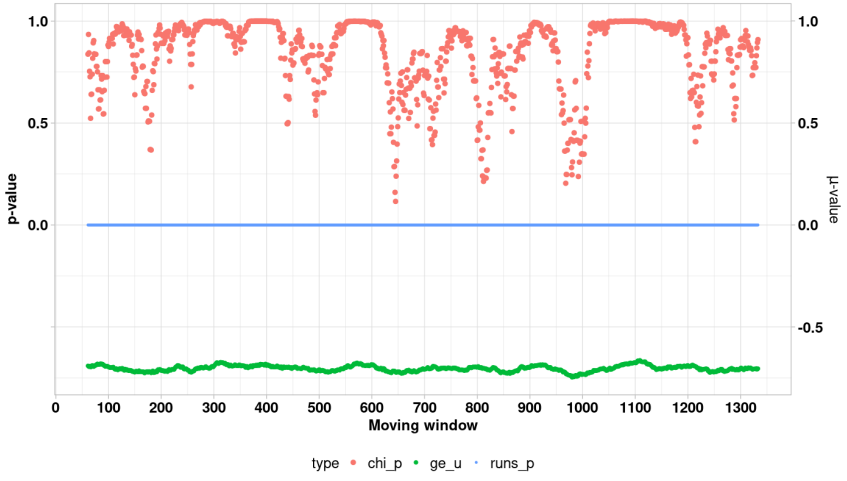
Fig. 3.10 shows the results of different oscillatory sequences with a success probability of 0.55. Fig. 3.10a, Fig. 3.10b, and Fig. 3.10c show the results for sequences with $\mu$ values -0.9, -0.8, and -0.7. The $\mu$ values of each sequence during the moving windows are close to the $\mu$ values of the entire respective sequence. For instance, in Fig. 3.10b, the $\mu$ value for the entire sequence is -0.8. The $\mu$ values in moving windows vary between -0.76 and -0.85. For each sequence, the Runs-test rejects the null hypothesis (p-value $< \alpha$) for every moving window. As mentioned in Sec. 3.3, for the Runs-test (two-tailed), bursty and oscillatory sequences fall in the rejection region. The results of the Chi-square test reflect the oscillatory characteristics of the sequence. As the $\mu$ value of the entire sequence increases from -1, the Chi-square




(a) Results of oscillatory sequence with $\mu$ value of -0.9




(b) Results of oscillatory sequence with $\mu$ value of -0.8




(c) Results of oscillatory sequence with $\mu$ value of -0.7

Figure 3.10: Statistical methods are applied to a moving configuration $s_{mw} = 60$ and $s_{Bs} = 30$ to oscillatory sequences with different $\mu$ values

test shows evidence for the fitness to Bernoulli distribution. In other words, as the sequence becomes oscillatory, the Chi-square rejects the null hypothesis.

After analysing several sequences, we found that statistical methods confirm the ground truth and react towards changes in the sub-sequence, thus answering Question II (see Sec. 3.1). We observed a relationship between the $\mu$ value and the outcome of the Runs-test. For all sequences with the $\mu$ value 0, the Runs-test shows evidence for randomness. For instance, in the case of statistically independent sequences, if the sub-sequence in a moving window shifts from the statistically independent region (the shift in $\mu$ value from 0 towards 1 or -1), for the same sub-sequence, the Runs-test shows evidence to reject the null hypothesis (p-value $< \alpha$). In [Hum77], the author discusses the existing tight bond or relationship between independence and randomness as (the author argue for a loosening of the bond between them);

> "*The concepts of randomness and independence have always been closely connected in probability theory. Definitions of a random sample generally insist that the successive elements be independent and definitions of random sequences making use of independence are common. There are varying degrees with which randomness and independence have been connected. An extreme position makes independence sufficient for randomness. Other positions retain independence as a necessary condition for randomness. Of course, there is no doubt that independence is present in many of the cases where we have randomness.*" - Paul W. Humphreys

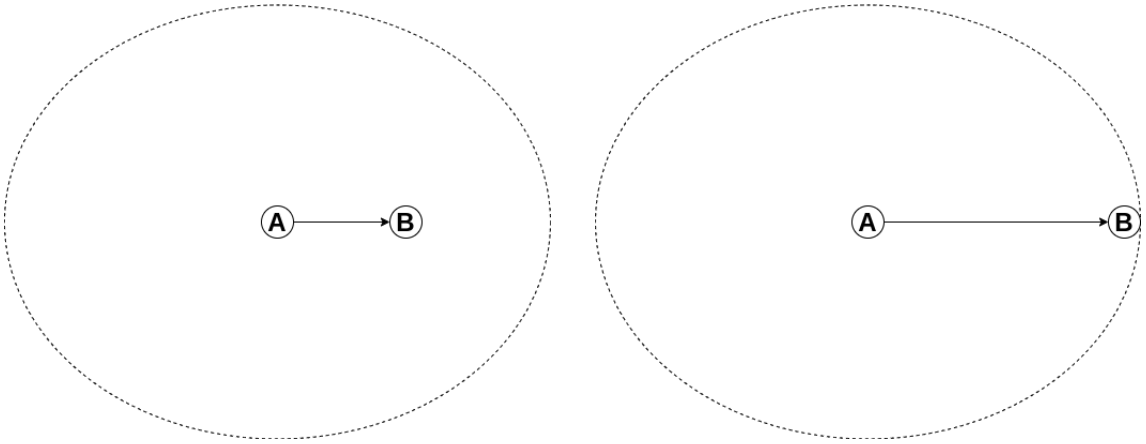
The Chi-square test rejects the null hypothesis for sequences with $\mu$ values close to 1 and -1 (i.e., a lack of evidence for fitness to a Bernoulli distribution with success probability $p$) for statistically dependent (bursty) and oscillatory sequences.

Concerning the selection of moving window size, a smaller moving window magnifies the short-term dependency and other effects. On the other hand, a larger moving window averages the result. In our analysis, we used Bernoulli slot of size $s_{Bs}$ and moving windows of size $s_{mw}$ 30, 60, and 100. Regarding computational complexity, calculating the $\mu$ value is the least complex among the three and easy to implement.

### 3.4.3 Real link data from a collision-free environment

This section evaluates link data from a collision-free environment (empty channel). We conduct this set of experiments to understand the characteristics of transmis-

sion events due to spatial properties. In this environment, a failure event or failed transmission occurs due to the spatial properties of the network.




(a) Node B is located inside the communication range of node A

(b) Node B is located at the border of the communication range of node A

Figure 3.11: Node placement for the experiments in collision-free environment. The dotted circle denotes the communication range of node A.

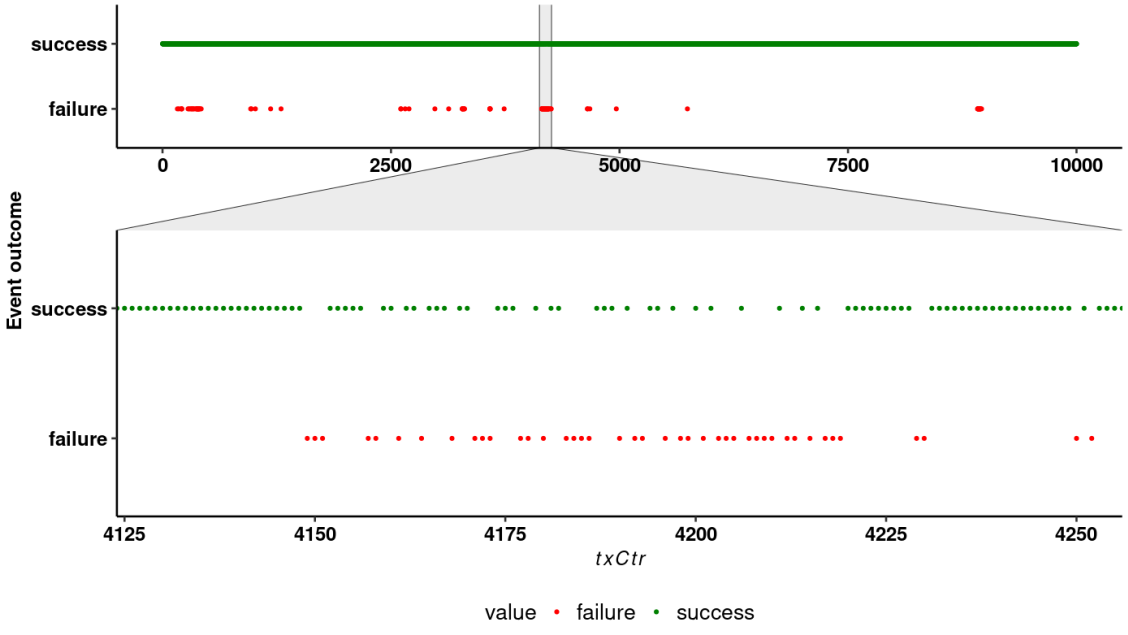



Figure 3.12: First 10000 event outcomes on link A→B during the second experiment.

Fig. 3.11 shows the node placement for the experiments. In this experiment, Nodes A and B are the only sender and receiver, respectively. Unlike in Fig. 3.11, the nodes are not placed in the line of sight. We conduct two experiment runs on the link A→B with different transmission power. For the first experiment, node B is located well inside the communication range of node A. And, for the second

experiment, node B is located at the communication range border of node A. For the latter, we reduce the transmission power of node A to decrease its communication range.

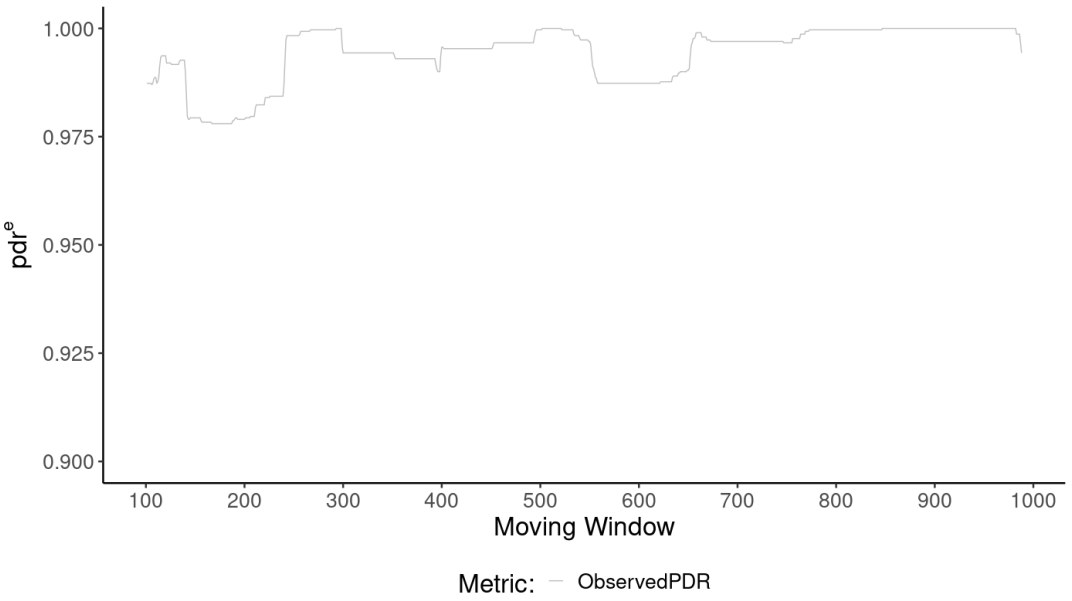
In both experiments, the sender node A periodically (every 10 ms) broadcasts frames with constant payload size at a transmission rate of 6 Mbps. We conducted experiments for 5 minutes. During an experiment run, node A sends around 30000 frames.

In the first experiment, the receiver node received frames with an average frame RSSI of -60 dBm. For the first experiment where node B is well inside the communication range of node A, the link reported zero failure events, resulting in a PDR of 1. Since there are no failure events for the first experiment, it is excluded from further analysis. For the second experiment, the receiver node received frames with an average RSSI of -89 dBm. In this experiment, the link reports several failure events.

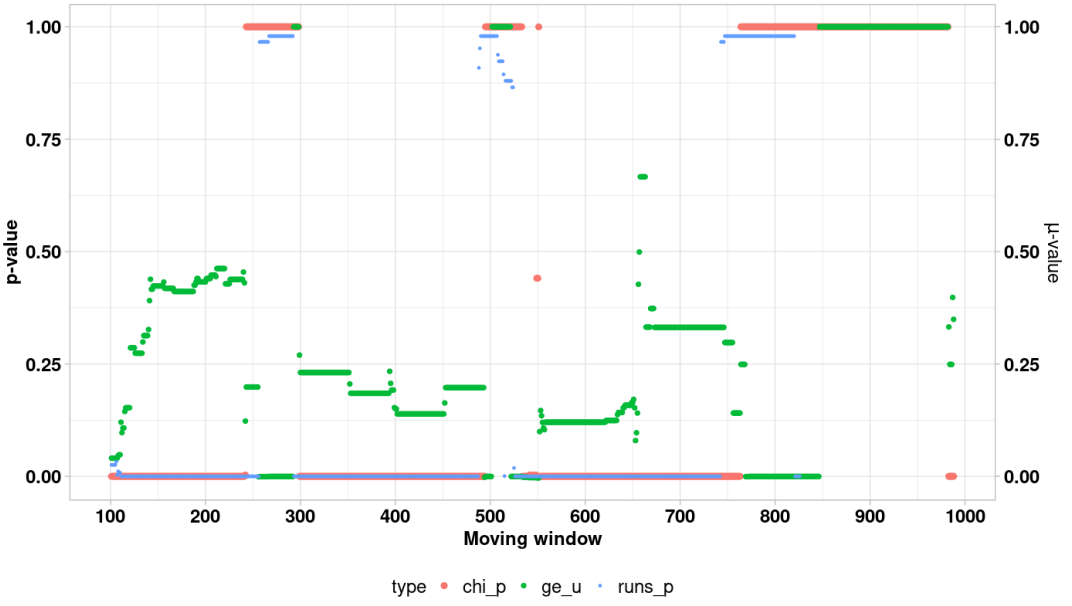
In our evaluation, we use Bernoulli slots of size $s_{Bs} = 30$ and moving windows of size $s_{mw} = 100$. Similar to the evaluation of synthetic data, in each moving window, we calculate the channel memory $\mu$ value from the GE model, p-values of the Chi-square test, and the Runs-test. The Chi-square test uses a sub-sequence size of $k = 30$. In a moving window, it results in 100 (3000/30) sub-sequences for the Chi-square test.

Fig. 3.12 shows the first 10000 transmission outcomes on link A→B for the second experiment. The PDR of the link during the entire experiment is 0.99. Transmission event outcomes show that frame losses are not statistically independent. Fig. 3.13a and Fig. 3.13b show the PDR and the statistical test results in each moving window, respectively. Out of moving windows with at least one failure, 78% of the windows rejected the null hypothesis for the Chi-square test, and 79% of windows reject the null hypothesis for the Runs-test with at least 95% confidence interval. The $\mu$ values for the vast majority of the moving windows also concludes it as not statistically independent. We calculated the $\beta$-metric for the entire observed data and obtained a $\beta$-metric value of $\approx 0.75$. It indicates that the outcomes of the events on the link (more probable) to be statistically dependent.

In [Sri10], the authors evaluated wireless links in a collision-free environment. Their study concludes that most of the links they analyzed in their experiments (in a collision-free channel) are bursty or dependent. Our limited experimental study shows signs supporting their findings. We primarily focus on studying the links in non-deterministic networks with foreign traffic. Thus, we limit our experiments with links in a collision-free environment.




(a) PDR during each moving window




(b) Statistical test results in each moving window

Figure 3.13: Results for the experiment where node B is located at the communication range border of node A. Evaluation uses moving windows of size $s_{mw} = 100$ and $s_{Bs} = 30$.

### 3.4.4 Environment with foreign traffic

The temporal properties of the network are a significant factor that influences the characteristics of a link in a real-world scenario, a network where collisions exist. In this set of experiments, we analyze the data collected from the testbed operating in the presence of non-deterministic external traffic. We arrange the experiments into two parts, 1) the nodes in the testbed network generate aperiodic traffic, and 2) periodic traffic. In the former part, we evaluate the links under different network loads by increasing the traffic generated by the testbed. Furthermore, we analyze the links while the network generates non-random or periodic traffic in the latter part.

| Setting | Min | Max delay | Estimated bandwidth usage (in notion of channel time usage) by the testbed |
|---|---|---|---|
| *S.A.1* | 100 ms | 200 ms | 3 % |
| *S.A.2* | 50 ms | 100 ms | 6 % |
| *S.A.3* | 5 ms | 50 ms | 20.2 % |
| *S.A.4* | 2 ms | 20 ms | 41.5 % |

Table 3.2: Frame generation settings used in aperiodic traffic experiments

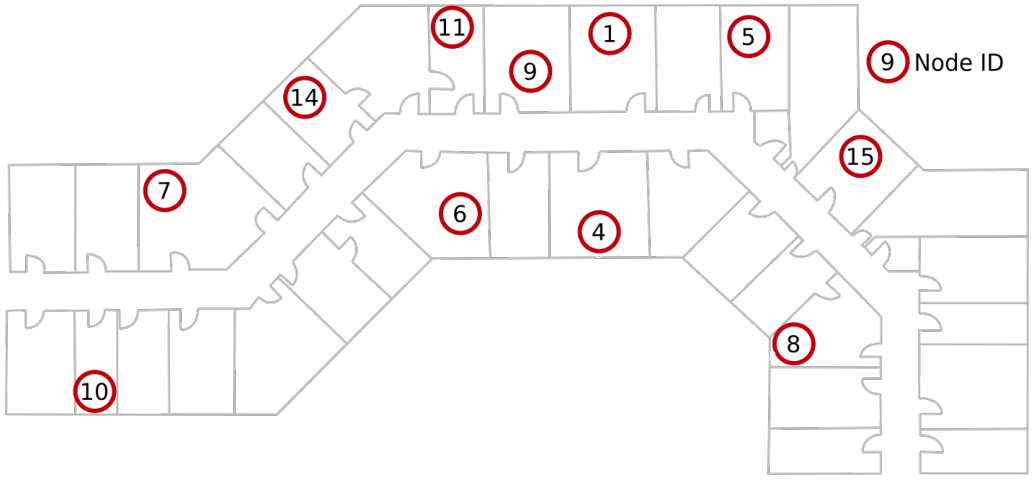



Figure 3.14: Floor plan of the AGVS testbed

In the first part, we conduct multiple experiments in four traffic generation settings. Tab. 3.2 shows the configuration for each setting and the estimated network load generated by the testbed traffic. The delay between each transmission is selected from uniform distributed intervals. For instance, setting *S.A.1* has a minimum and a maximum delay of 100 and 200 ms; the node selects a random delay before the next transmission, uniformly distributed in the intervals of 100 and 200 ms. When the testbed nodes operate in this setting, they consume 3% of the total usable network

bandwidth. Using settings *S.A.2*, *S.A.3*, and *S.A.4* the nodes generate a higher network load, the spread of possible delays between two consecutive transmissions is reduced. For instance, using *S.A.1* generates less testbed traffic compared to using *S.A.2*. Furthermore, using *S.A.1*, a node could delay the next transmission by a value between 100 ms and 200 ms, i.e., a spread range of 100 ms. Meanwhile, using *S.A.2*, the possibility of the next delay is between 50 ms and 100 ms, i.e., a spread range of 50 ms. As the spread range for the possible delay between consecutive transmissions decreases, the probability of multiple nodes selecting similar delays between consecutive transmissions increases.

Fig. 3.14 shows the floor plan of the testbed. Experiments are conducted during working hours. In each setting, we conduct multiple experiments. The nodes transmit frames at a transmission rate of 6 Mbps with a payload of constant size. We

Figure 3.15: Link 8 → 15 during experiment using setting *S.A.1*

start by discussing the results using the lowest network load, using setting *S.A.1*. In the lowest network load, approximately 3 % of the traffic is generated by the testbed. We observed that most links have statistically independent and random event outcomes in this setting. For instance, Fig. 3.15 shows the results of the link from nodes 8 to 15. The GE model channel memory parameter, $\mu$ throughout moving windows was 0, and outcomes of non-parametric tests, the Runs-test and the Chi-square fail to reject their null hypotheses with 95% confidence interval (a p-value $\leq \alpha$, where $\alpha = 0.05$). I.e., throughout the experiment, the event outcomes were statistically independent, random, and fit a Bernoulli distribution.

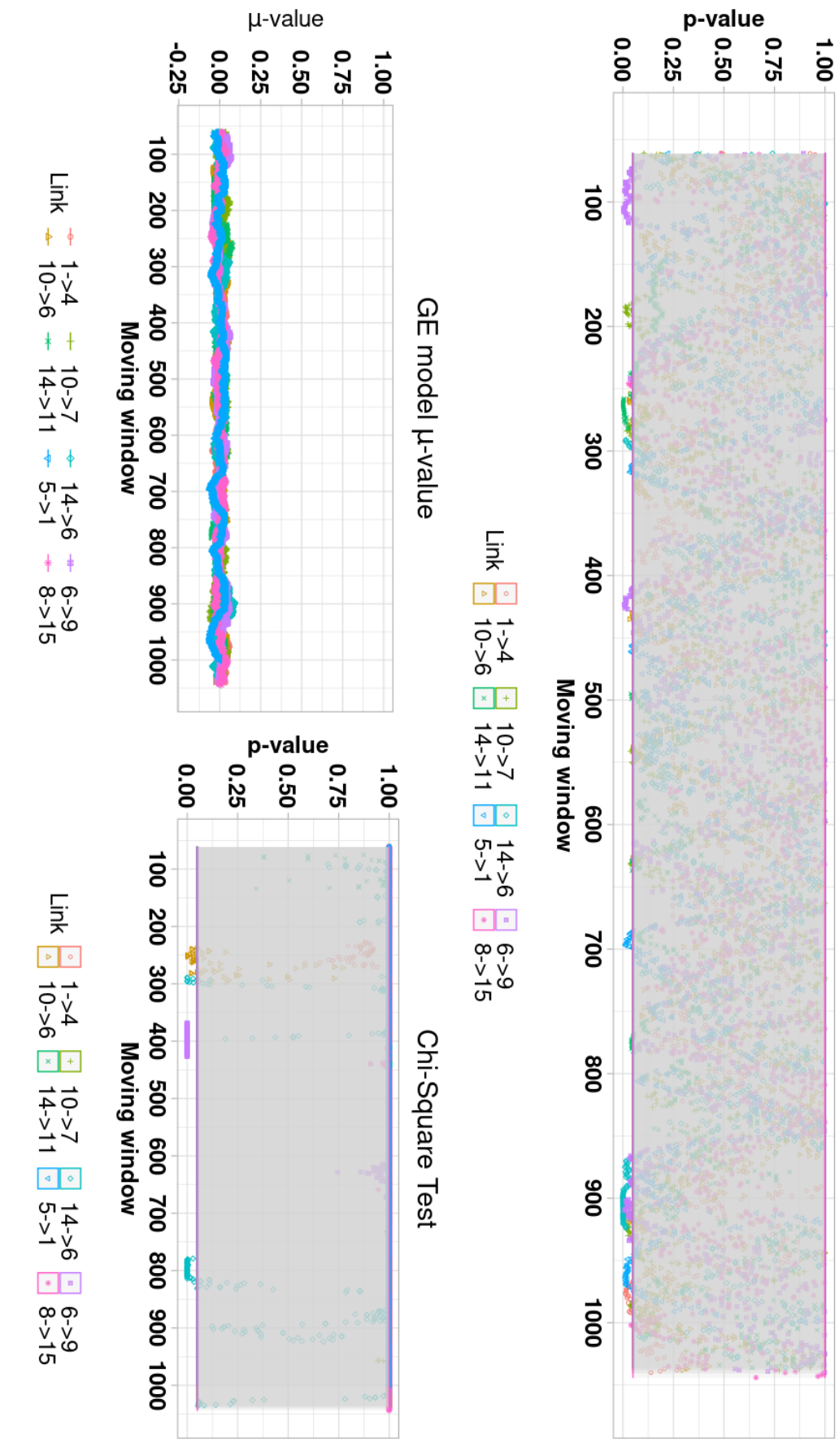
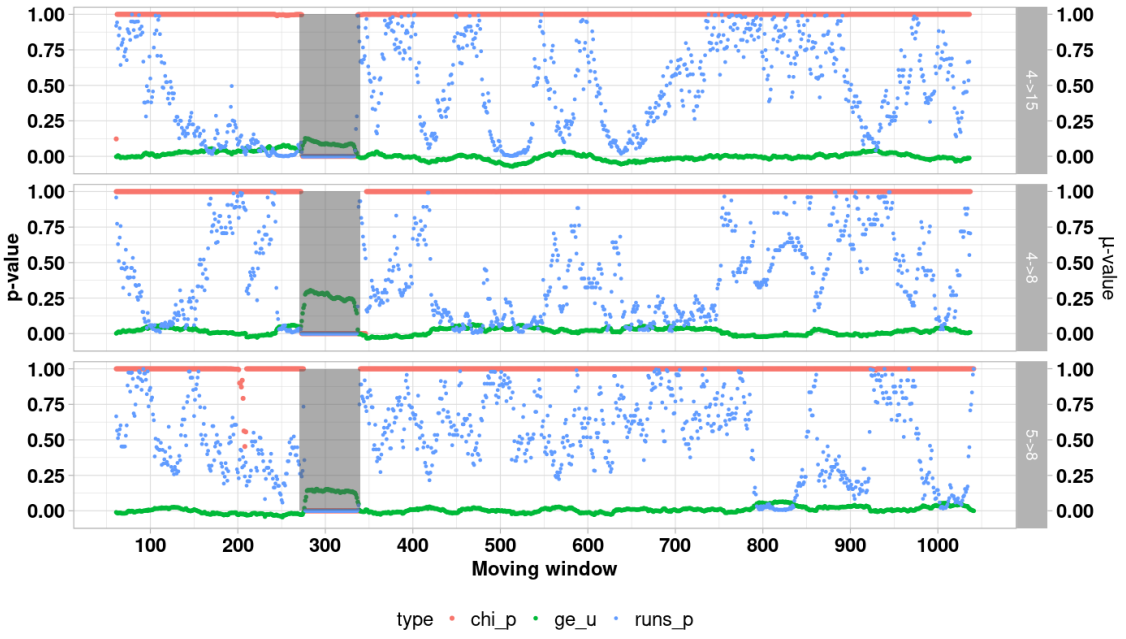
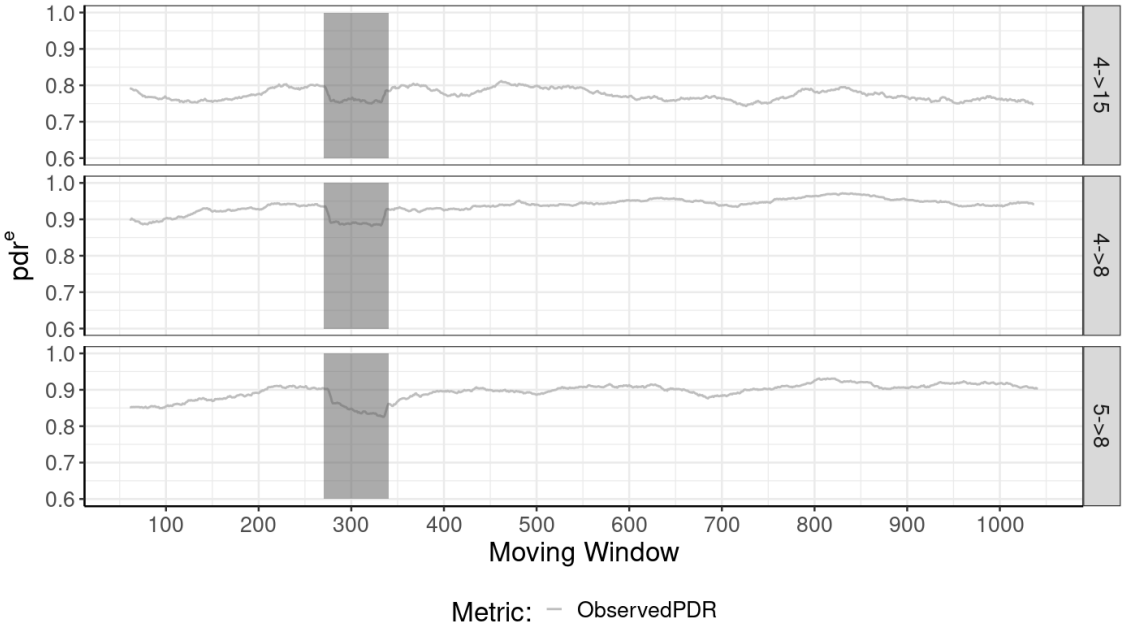



Figure 3.16: Statistical test results for a set of links during an experiment using traffic generation setting *S.A.1.* Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{B_s}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$).




(a) Statistical test results




(b) PDR over the windows

Figure 3.17: A set of links where non-random and statistically dependent (bursty) frame losses were observed during an experiment using setting *S.A.1*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30.

Fig. 3.16 shows the results of few links during the experiment. Plots contain a link from, or to, at least one of the nodes in the testbed. The non-shaded region represents the rejection zone with a confidence interval of 95%, i.e., with a 95% confidence interval, we can reject the null hypothesis of the respective non-parametric tests. In rare cases, results from the Runs-test show evidence of non-randomness, with a confidence interval of 95%, i.e., a p-value below 0.05. Similarly and rarely, the Chi-square tests, which evaluate the fitness to a Bernoulli distribution, show evidence to reject the null hypothesis. The $\mu$ values throughout the moving windows for the links are close to 0, showing that the event outcomes of sub-sequences are statistically independent. When we look back to the results of synthesized statistically independent Bernoulli sequences in Sec. 3.4.2, there are rare outliers, i.e., non-random sub-sequences were observed. In these experiments, using setting *S.A.1*, a node's transmission interval between each consecutive transmission is random, resulting in transmissions at random time points. Although it decreases the possibility of dependent transmission event outcomes, it does not fully eliminate the possibility. These non-random event outcomes occur either due to frame collisions between testbed nodes or collisions with external nodes' frames [8]. For instance, in Fig. 3.17, between moving windows 290 and 320, we observed several non-random event outcomes. We observed that a set of links was affected during the same time interval.

Fig. 3.17 shows results for links 4→15, 4→8, and 5→8. Between moving windows 290 and 320, the Runs-test and the Chi-square test results show that the sub-sequences are non-random and do not fit a Bernoulli distribution. Furthermore, the $\mu$ values during those windows increased from 0. As mentioned earlier, it could be either due to occasional synchronised transmission by the nodes in the testbed, resulting in a collision between frames sent by the nodes or due to a rapid surge in external traffic. An external device's high demand in bandwidth usage might affect the links in a testbed, in varying magnitude, depending on their location. We believe it is more likely due to the latter since three known WiFi Access Points are located within the premises of the most affected links.

As we increase the network load, we observe an increase in the number of moving windows with sub-sequences as dependent or bursty and non-random transmission event outcomes. For instance, Fig. 3.18 shows results for the same set of links as in Fig. 3.16 using setting *S.A.3*, with a higher network load and less delay spread between consecutive transmissions. Compared to experiments with lower testbed traffic, we observed a relative increase in the number of moving windows with non-

[8] including Access Points (AP) and other devices operating in the same frequency.

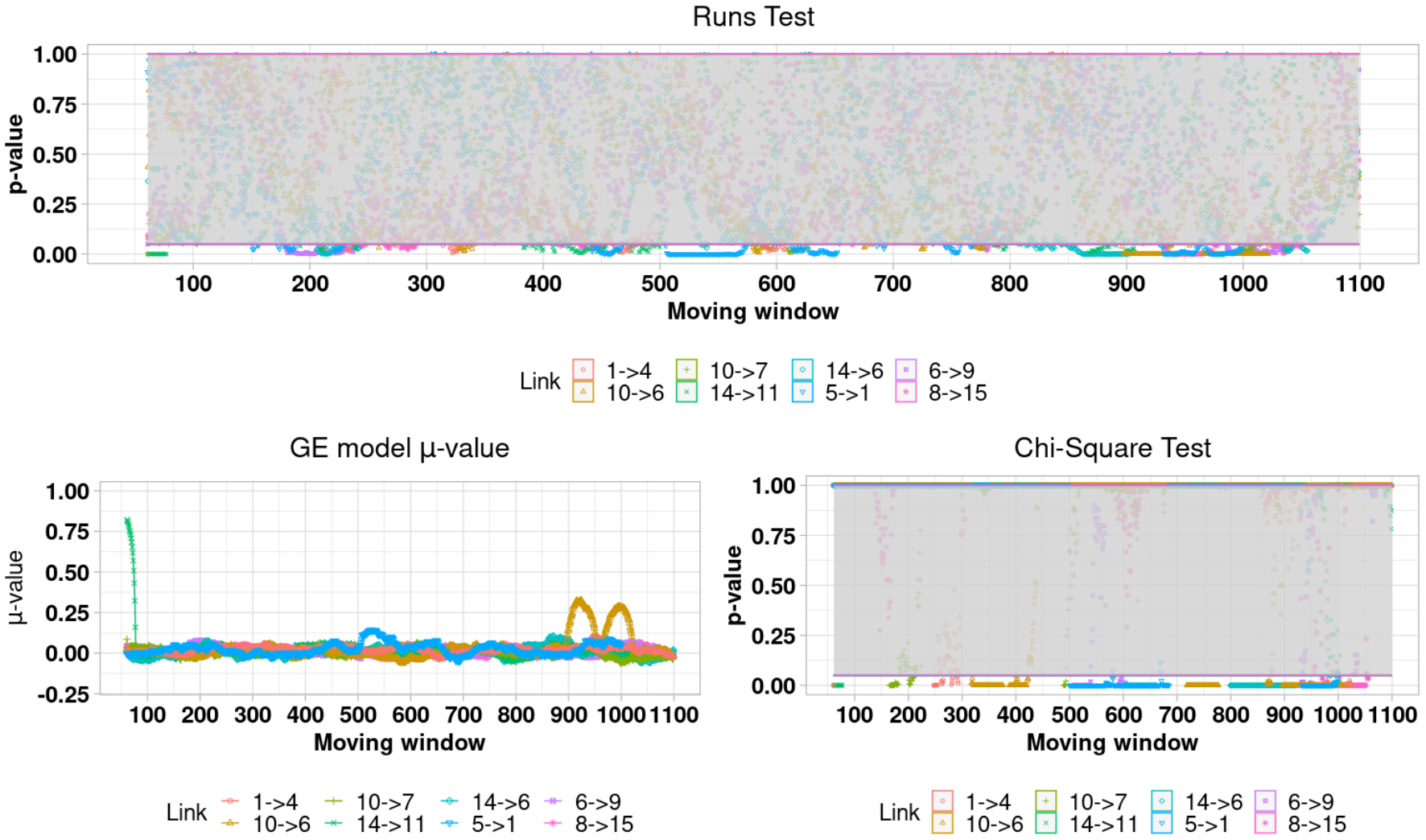



Figure 3.18: Statistical test results for a set of links during an experiment using traffic generation setting *S.A.3*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$).

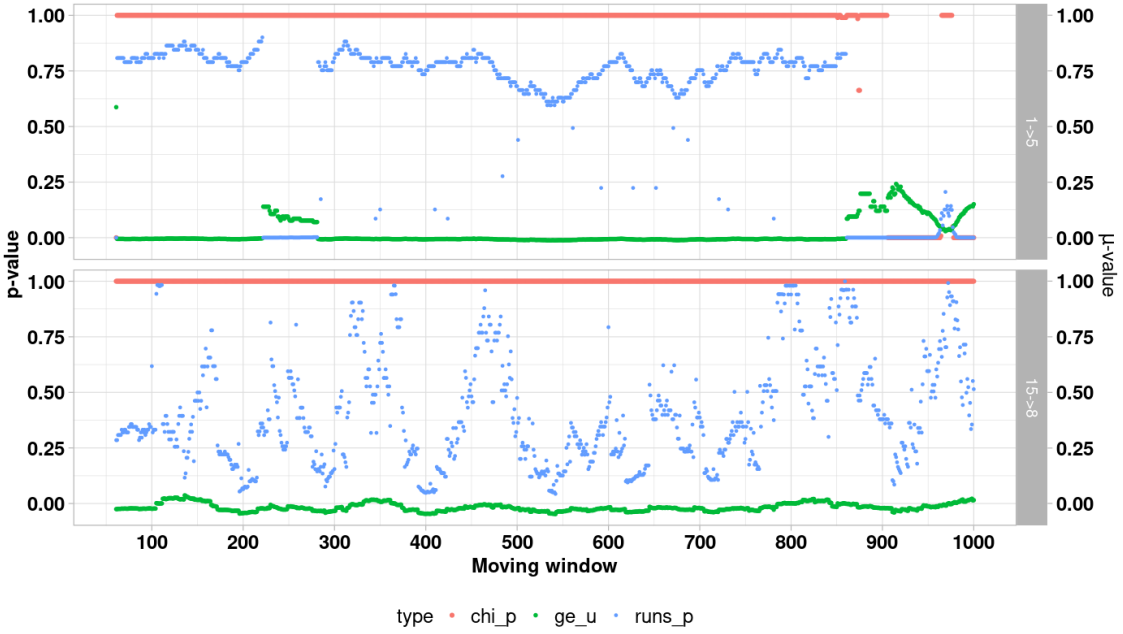



Figure 3.19: Links during experiment using setting *S.A.3*. Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30.

random sub-sequence, and evidence of not fitting to a Bernoulli distribution. Also, we observed a slightly higher $\mu$ value for most links. Furthermore in some links, the $\mu$ value surged to a higher value during several moving windows.

Nevertheless, we observed few links where the event outcomes during the moving windows were random and independent for most of the time, even at a higher load. For instance, Fig. 3.19 shows the results for two links with PDR above 0.95 from the experiment using setting *S.A.3*. Link 1→5 (upper diagram) observed random and statistically independent event outcomes, except between windows 250 to 280, and 860 to 960. Meanwhile, link 15→8 (lower diagram) observed random and statistically independent event outcomes for almost all windows. Overall, as we increased the network load, we observed an increase in the number of links, where the frequency of non-random and bursty subsequences increases. For instance, Fig. 3.20 shows the results for two links with PDR above 0.95 from the experiment using setting *S.A.4*. Compared to link 1→11 (upper diagram), 4→5 (lower diagram) observed a higher number of windows where the non-parametric tests show evidence to reject the null hypothesis.

Next, we investigate possible long term dependencies by calculating the $\beta$-metric [Sri+08] for each link during the entire experiment and in each setting. Fig. 3.21 shows the PDR on a link and their respective $\beta$-metric value calculated from the experiment data in different settings. The plot shows calculated $\beta$-metric value of

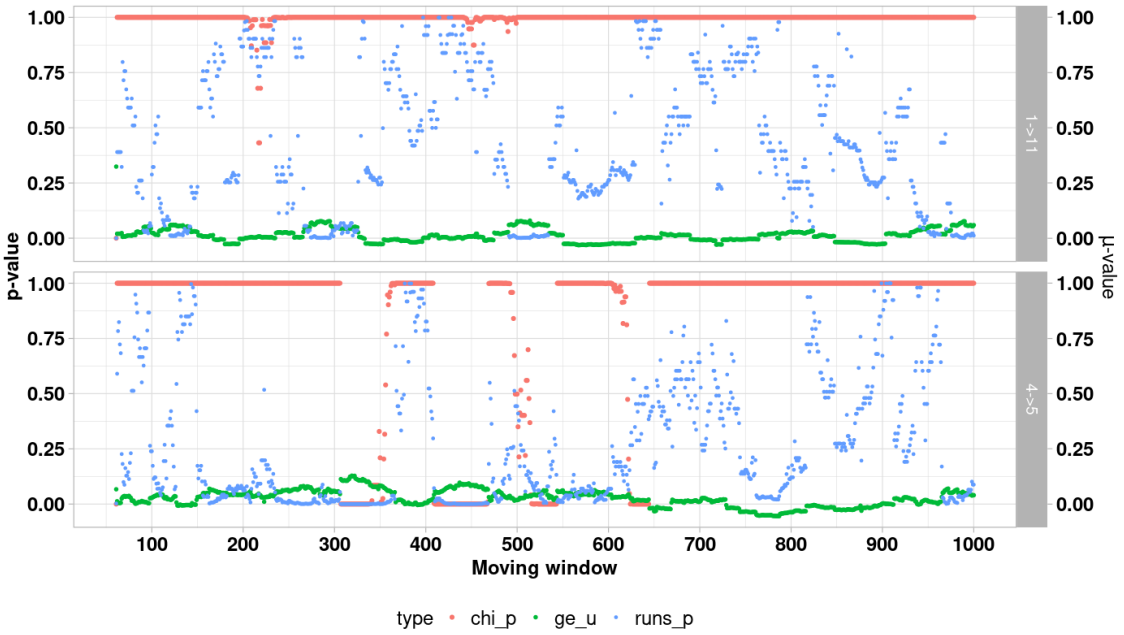



Figure 3.20: Links during experiment using setting *S.A.4*. Evaluation uses moving windows of size $s_{mw}$ = 60 and $s_{Bs}$=30.

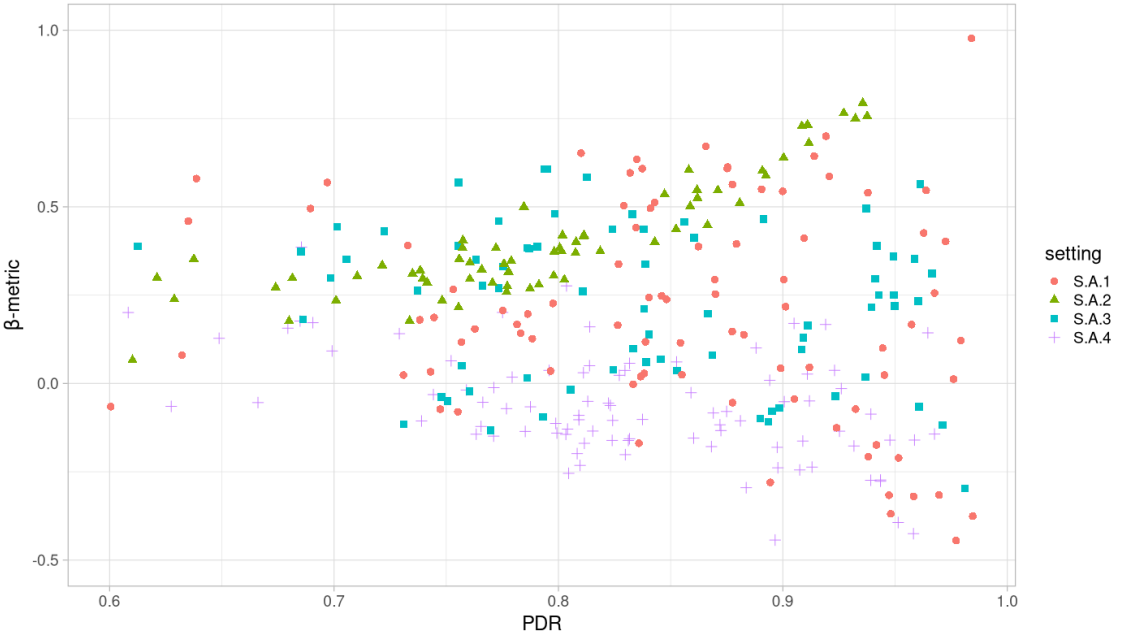



Figure 3.21: $\beta$-metric for links during the entire experiment

the links with an PDR above 0.60. The results demonstrate that when the testbed traffic increases, there is a slight increase in the $\beta$-metric of certain links, i.e., the likelihood of dependent transmission event outcomes increases. Using setting *S.A.1*, with estimated testbed traffic using 3% of the total available bandwidth, more than 90% of the links in the testbed observed a $\beta$-metric value below 0.25. When the testbed traffic is increased, using settings *S.A.2*, *S.A.3*, and *S.A.4*, certain links observed a higher $\beta$ value. In those experiments, we discovered links between various characteristics. For instance, there are links as the network load increases, that

- observed a similar PDR [9] and $\beta$-metric value,
- observed a similar PDR and an increase in $\beta$-metric value,
- observed a similar PDR and varying $\beta$-metric values during each run (either close to 0 or higher), and
- observed a drop in their PDR and an increase in their $\beta$-metric value.

To conclude, in this set of experiments with aperiodic traffic from testbed node, together with external traffic, we observed that at traffic settings *S.A.1* and *S.A.2*, most of the links hold the statistical independence property. In those settings, the overall network load is low compared to settings *S.A.3* and *S.A.4*. Furthermore, using the former settings, the delay spread between consecutive transmissions is higher than the latter. These factors, low network load and random delay between transmissions, together with external traffic, and the underlying random backoff mechanism of 802.11, produced an independent *conditional collision probability* for links. We also observed periods where the event outcomes became bursty and dependent. As we moved into settings *S.A.3* and *S.A.4*, we observed a higher likelihood to observe non-random and bursty event outcomes. Examining the results from applying the $\beta$-metric to collected data, we observed that when we increase the network load above a certain range, it increases the likelihood of increasing the number of links in a network with long term dependent transmission event outcomes.

As per 802.11 protocol [80216], when a node attempts a new transmission, it senses the medium, and if the medium is not idle, the node does a random backoff before transmission. However, on the other hand, if the medium is idle, it will immediately transmit without any random backoff delay. Thus, in those experiments, the characteristic of the application traffic, where nodes were transmitting at aperiodic intervals and the random backoff mechanism of WiFi protocol contributed to the

[9] PDR during the entire experiment

independence property of links. What if the application traffic generated by nodes is not aperiodic?

Next, we discuss the results of experiments where nodes transmit frames in a periodic interval. In this scenario, the nodes periodically transmit frames, unless when the node attempts the transmission, a random backoff happens due to a busy medium. However, as the experiments are conducted in an uncontrolled environment, the likelihood of random backoffs due to external traffic is higher. We conduct multiple experiments in four traffic settings. In settings *S.P.1*, *S.P.2*, *S.P.3*, and *S.P.4*, all nodes transmit at a periodic interval of 150ms, 75ms, 27ms, and 13ms, respectively. The settings *S.P.1*, *S.P.2*, *S.P.3*, and *S.P.4* contribute approximately 3%, 6%, 20%, and 41%, respectively, of the total medium capacity, equivalent to the traffic generated during experiments with aperiodic traffic patterns using settings *S.A.1*, *S.A.2*, *S.A.3*, and *S.A.4* (see Tab. 3.2).

In those settings where frames are transmitted in periodic intervals, we observed links with properties in both spectra, i.e., statistically independent and random, and statistically dependent and non-random or bursty. However, unlike in the scenario with the aperiodic frame transmissions and nodes operating under a similar network load, more links had bursty and statistically dependent event outcomes, with long bursty frame losses. For instance, Fig. 3.22 shows results for the same set of links as in Fig. 3.18, during an experiment using traffic setting *S.P.1*. In the former, the traffic is generated periodically, while in the latter, in an aperiodic manner. The amount of traffic generated by the testbed in both experiments is similar, approx 3%. For instance, the link 1→4 observed several non-random sub-sequences or moving windows and a $\mu$ value in the region -0.125. Similarly, other links also recorded more non-random sub-sequences and a small surge in $\mu$ values.

Fig. 3.23 shows the results for link 11→5, from an experiment using setting *S.P.1* with lowest testbed load. The link observed an average PDR of 0.81 considering the entire experiment duration. It observed multiple bursty frame losses of different lengths between moving windows 100 and 700. As we increase the network load, i.e., experiments using settings S.P.2, S.P.3, and S.P.4, we observed a higher number of links with non-random and bursty subsequences. Moreover, we observed that certain links were affected in most of the experiments, and their $\mu$ values increased and showed strong evidence to reject the Runs-test's null hypothesis as the experiment settings moved from S.P.1 to S.P.4 (higher testbed traffic).

Examining the data from multiple experiments, we observed some signs of bursty or dependent frame losses due to periodic transmission intervals even at a low network load. To a great extent, the presence of external traffic and other factors

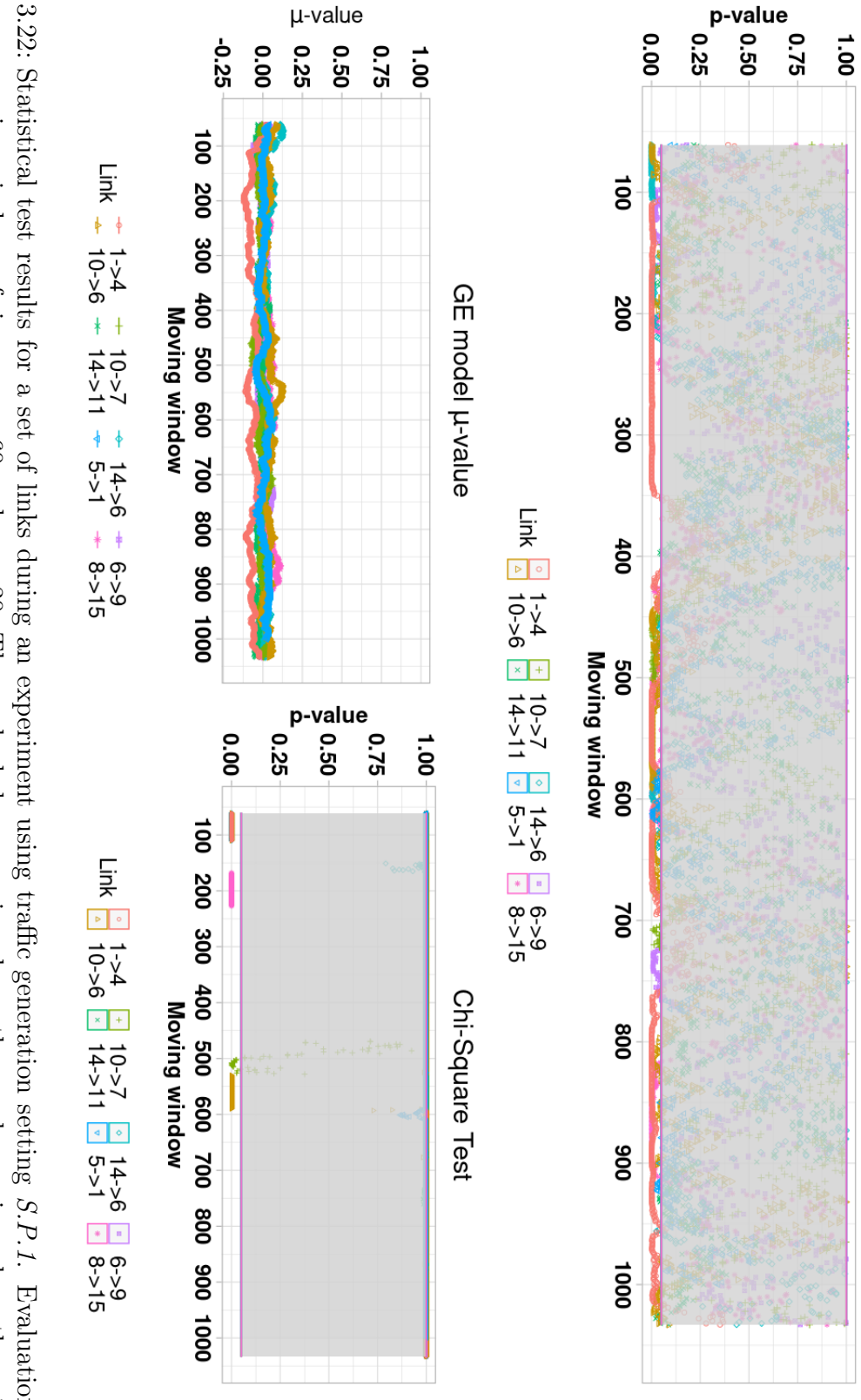



Figure 3.22: Statistical test results for a set of links during an experiment using traffic generation setting *S.P.1.* Evaluation uses moving windows of size $s_{mw} = 60$ and $s_{Bs}$=30. The unshaded grey region shows the p-value region where the evidence to reject the null hypothesis is rejected with 95% confidence interval ($\alpha = 0.05$).

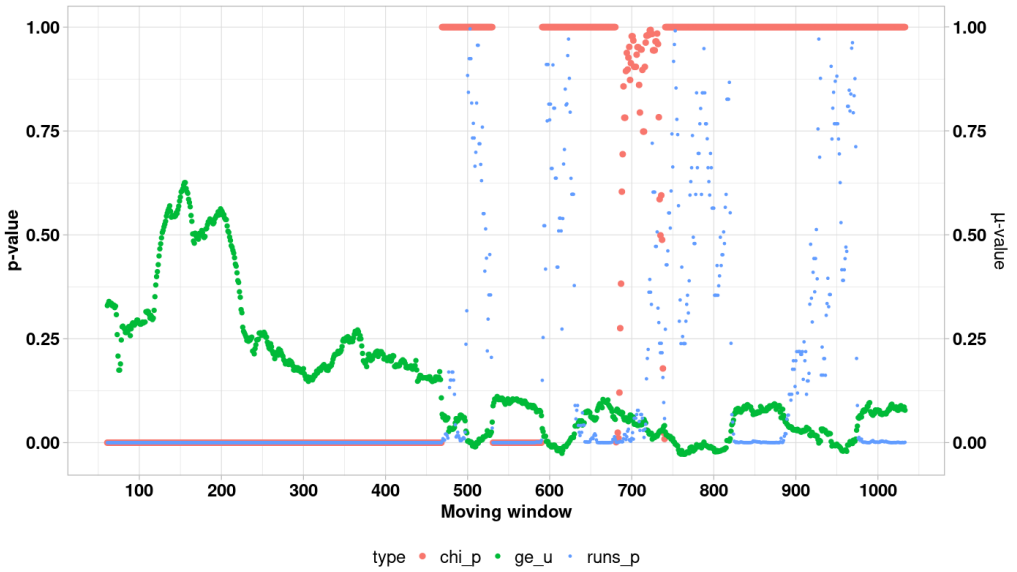



Figure 3.23: Statistical test results for the link 11→5. Evaluation uses moving windows of size $s_{mw}$ = 100 and $s_{Bs}$=30.

[10] might mitigate the effects of periodic transmission. These results lead to the question:

- What if the testbed or a network operates in the absence of external traffic, and the network generates traffic in the periodic pattern? In such a scenario, whether the underlying random backoff is adequate to assume that the conditional collision probability on links is independent?

We look into the above question in Chapter 5 (see Sec. 5.3.3).

## 3.5 Conclusion

In this chapter, we examined suitable methods to determine the statistical properties of a link and conducted experimental assessments using synthetic data, data from a collision-free environment, and data from an environment with foreign traffic. Looking back into the questions presented at the beginning of this chapter (see Sec. 3.1), in this chapter, we answered Question II and Question III. To measure the various statistical properties of a wireless link (Question II), we discussed possible statistical tests in Sec. 3.3. Further, we assessed the quality of those statistical tests using synthesized data in Sec. 3.4.2. Instead of applying the statistical tests to the entire sequence, we applied the tests to sub-sequences inside a moving window to assess a link's short-term characteristics.

[10]Such us hardware bus delay or periodic timer inaccuracies

We use the $\mu$ parameter from the Gilbert-Elliot [Gil60][Ell63] model to measure the independence between outcomes of successive transmission events. We applied the non-parametric test, the Runs-test [Bra68], to understand the randomness of transmission event outcomes, i.e., to check whether frame losses occur randomly. Further, we applied another non-parametric test, the Chi-square test for goodness to fitness, to measure the fitness of observed Bernoulli trials (generated from the results of transmission outcomes) to a Bernoulli distribution with success probability $p$. Furthermore, we borrowed the $\beta$-metric [Sri+08] to examine any long-term dependency in transmission outcomes. To evaluate the quality of the aforementioned statistical tests, we applied the tests on synthetic data. I.e., tests are applied to the sub-sequences of synthesized data of known types, which contains statistically independent and random, statistically dependent or bursty, and oscillatory Bernoulli sequences. Results show that the statistical tests infer the ground truth.

To discover the characteristics of wireless links in different medium conditions, i.e., Question III, we applied the statistical tests to link data gathered from experiments in a collision-free environment and an environment with foreign traffic, in Sec. 3.4.3 and Sec. 3.4.4, respectively. Our limited experimental study on data gathered from experiments in a collision-free environment shows that the likelihood of transmission outcomes being dependent is higher than being independent. To understand the characteristics of links in a real-world scenario, we evaluated the link data from a testbed in the presence of external traffic. We conducted experiments where the testbed nodes generate aperiodic and periodic traffic. In both cases, as we increased the traffic generated by the testbed, we noticed a higher likelihood of observing non-random and bursty event outcomes. Compared to experiment results where the testbed nodes generate aperiodic traffic (operating under similar network load), more links had bursty and statistically dependent event outcomes, while the nodes generate periodic traffic. The periodic transmission intervals might have affected certain links. We look further into the impact of traffic patterns on link quality in later chapters.

By answering Question II and Question III, we lay the foundation to explore the solutions for Question I, i.e., *what are the statistical properties required to improve link reliability by retransmissions*? To find an answer, we need to evaluate the relationship between the statistical test outcomes, i.e., when a sequence holds the property and the efficacy of link boosting via retransmissions. Also, it leads to further questions. In the following chapter, we examine those questions. Further, we cannot use the p-values from a non-parametric test to compare the statistical link properties. We discuss this too further in the following chapter.

# 4

# Applying Statistical Methods to Identify *Link Boosting* Capability of Wireless Links

## Contents


**4.1 Introduction** . . . . . **82**

**4.2 Link Boosting** . . . . . **82**

4.2.1 Methodology . . . . . 84

4.2.2 Evaluation techniques . . . . . 88

**4.3 Experimental Assessment** . . . . . **89**

4.3.1 Evaluation using Synthetic data . . . . . 89

4.3.2 Evaluation using Real world link data . . . . . 105

**4.4 Measuring the "margin of safety" required to tolerate variation in statistical properties of a link** . . . . . **109**

4.4.1 Foundations . . . . . 110

4.4.2 Influence of sample size . . . . . 113

4.4.3 Influence of target reliabilities . . . . . 115

**4.5 Conclusion** . . . . . **119**

## 4.1 Introduction

In the previous chapter, we devised foundations to investigate statistical properties required to improve link reliability by retransmissions, i.e., to perform link boosting. Following those groundworks, this chapter examines the necessity of certain statistical properties such as statistically independent transmission event outcomes, randomness, or fitness to a Bernoulli distribution, to perform a successful link boosting. Thus, the following question is investigated: *What statistical properties are required to improve link reliability by retransmissions*? I.e., revisiting the Question I mentioned in the previous chapter.

The chapter is structured as follows: to begin with, in Sec. 4.2, we discuss the link boosting process on a link for a route, the steps to perform it, and introduce a metric to measure the effectiveness of link boosting. Furthermore, we also revisit the evaluation techniques used to measure the statistical properties of a Bernoulli sequence. Then, in Sec. 4.3, we present the experiment results on synthetic data and data collected from the testbed. It includes experiments on synthetic data with the motivation to understand the role of statistical independence, randomness, and sequences' fitness to a Bernoulli distribution for successful link boosting. Moreover, we introduce the idea of "*Margin of Safety*" [11] in link boosting. Later, we analyze the link data (transmission outcomes) from the testbed and investigate their statistical properties. Then, in Sec. 4.4, we evaluate the margin of safety required to tolerate variations in the statistical properties of a link for successful link boosting. Finally, in Sec. 4.5, we conclude the findings.

## 4.2 Link Boosting

*Link boosting* is the practice of increasing the link reliability to a target reliability by retransmissions. To examine the relationship between effective *link boosting* and the necessity of certain statistical properties, we compare the result of *link boosting* in each moving window with the outcomes of statistical methods. In this part, we further extend our network model introduced earlier in the thesis and extended in Sec. 3.4.1 before describing our methods and metrics used to measure the effectiveness of *link boosting*.

A successfully received WiPS frame $f_{rx}$ is recorded as tuple $f_{rx} = (snd, rcv, txCtr, rxTs)$, where *snd* is the WiPS node address of the sender, *rcv* is the WiPS node address of the receiver, *txCtr* is the transmission counter value at the sender, and

[11] The terminology is borrowed from [Gra03]

*rxTs* is the local timestamp value when the frame arrives in the user stack of the receiver. To monitor the link reliability of a link during the operation of a route, the sender appends an extra parameter called frame identifier *fId*. Unlike the *txCtr*, the sender keeps a separate *fId* counter for each route identified using route identifier *rId* through the link. Further, the *fId* is not incremented for retransmissions. For instance, consider a node $v_1$ at runtime where the last frame sent from node $v_1$ to $v_2$ has a transmission counter *txCtr* value at 10, and the frame identifier *fId* value for a route with *rId* 101 is also 10. When the next frame $f$ for the route with *rId* 101 arrives at node $v_1$, the WiPS MAC layer assigns 11 as the *txCtr* (assuming, in between, no other frames from node $v_1$ is sent) and *fId*. Assume that node $v_1$ fails the first transmission attempt of frame *f*. When the node attempts retransmission (if it does not exceed the maximum retransmission limit), the node increments its transmission counter value *txCtr* to 12 and updates the *txCtr* in retransmitting frame *f*, while the frame identifier *fId* remains unchanged.

| *snd* | *rcv* | *txCtr* | *fId* | *rId* | *rxTs* (ms) |
|---|---|---|---|---|---|
| 10 | 11 | 1 | 1 | 1 | 1619189954100 |
| 10 | 11 | 2 | 2 | 1 | 1619189954200 |
| 10 | 11 | 3 | 3 | 1 | 1619189954700 |
| 10 | 11 | 5 | 4 | 1 | 1619189955230 |
| 10 | 11 | 6 | 5 | 1 | 1619189955400 |
| 10 | 11 | 8 | 6 | 1 | 1619189955980 |
| 10 | 11 | 12 | 8 | 1 | 1619189957100 |
| 10 | 11 | 13 | 9 | 1 | 1619189957200 |
| 10 | 11 | 14 | 10 | 1 | 1619189957560 |
| 10 | 11 | 15 | 11 | 1 | 1619189957900 |
| 10 | 11 | 17 | 12 | 1 | 1619189959300 |

**Bernoulli sequence for the corresponding raw data**
1 1 1 0 1 1 0 1 0 0 0 1 1 1 1 0 1

Table 4.1: A raw data sample on link 10→11 with $n^e_{max} = 2$

For instance, Tab. 4.1 shows the recorded frame transmissions on link 10→11, using single retransmission in case of a failed transmission or $n^e_{max}$ equals 2. In practice, we attempt $n^e_{max}$ transmissions to boost the link reliability and, when an acknowledgement is received, we avoid further retransmissions. It ensures that link reliability is boosted to the target if other necessary conditions [12] are met. In this example, the first registered frame loss is the frame with a *txCtr* and *fId* value of

[12] such as the outcome of upcoming events are statistically independent and the estimated reliability is less than or equal to the observed boosted reliability.

4 for the route with *rId* of 1. Since we assigned a maximum of two transmissions per payload, the payload is retransmitted in the frame with a *txCtr* and *fId* value of 5 and 4, respectively. Further in this example, there is only one failed payload delivery: the transmission of frames with a *fId* value of 7, i.e., the payload in frames with *txCtr* values 10 and 11.

### 4.2.1 Methodology

We conduct our study on synthetic data and data collected from the testbed. The synthetic dataset contains synthesized Bernoulli sequences with varying statistical properties. The data from the testbed contains link data under different traffic patterns. Similar to our previous study (see Sec. 3.4.1), we structure the Bernoulli sequences from synthesized and testbed data into moving windows. Recalling, a moving window $n_{mw}$ of size $s_{mw}$ consists of Bernoulli slots of equal size $s_{Bs}$. Further, a Bernoulli slot $n_{Bs}$ with size $s_{Bs}$ consists of a sequence of $s_{Bs}$ Bernoulli trials. On collected data, we examine the effectiveness of *link boosting* in the following manner:

***Step* 1:** Set the minimum target reliability $r^e_{target}$ required through the link for the route. In this chapter and in our evaluation, we operate only one route per link.

***Step* 2:** For each moving window, proceed with the following steps:

***Step* 2.1:** Calculate the raw PDR $pdr^e_{raw}$ during the moving window using Eqn. 3.5 (see Chapter 3).

$$pdr^e_{raw}(n_{mw}) = \begin{cases} \dfrac{n^e_{rx}(n_{mw})}{n^e_{tx}(n_{mw})}, & \text{if } n_{mw} \geq s_{mw} \\ \\ \text{undefined}, & \text{if } n_{mw} < s_{mw} \end{cases} \qquad \text{(Eqn. 3.5 revisited)}$$

where, $n^e_{rx}(n_{mw})$ and $n^e_{tx}(n_{mw})$ are the number of correctly received frames and the number of transmission attempts, respectively, on link $e$ during the moving window $n_{mw}$.

***Step* 2.2:** Using the Eqn. 3.1, find the maximum number of transmissions $n^e_{max}$ required in the moving window $n_{mw}$ to achieve the minimum target reliability $r^e_{target}$, i.e., $r^e_{target} \leq q^e_{n^e_{max}}(n_{mw})$.

$$q^e_{n^e_{max}}(n_{mw}) = 1 - (1 - q^e(n_{mw}))^{n^e_{max}} \qquad \text{(Eqn. 3.1 revisited)}$$

where the function $q^e(n_{mw})$ returns the aggregated link quality of link $e$ in the moving window $n_{mw}$, i.e., $q^e : \mathbb{N}_+ \rightarrow \mathbb{R}_{0,1}$. In practice, $q^e$ of the link $e$ is estimated using different link reliability metrics. Thus in practice, the effectiveness of link boosting also depends on the estimation accuracy of the selected link reliability metric. In Chapter 6, we discuss further on this topic. In our evaluation, to calculate $n^e_{max}$, $q^e(n_{mw})$ returns the raw PDR (or success probability) $pdr^e_{raw}(n_{mw})$ in a moving window, i.e., $q^e(n_{mw}) \mapsto pdr^e_{raw}(n_{mw})$. Therefore, in our evaluation, a failed *link boosting* happens due to the absence of certain statistical properties of the observed events.

***Step* 2.3:** Measure the observed boosted reliability PDR $pdr^e_{proc}(n_{mw}, rId)$ on the link by performing up to $n^e_{max}$ - 1 retransmissions for a failed frame transmission in the moving window $n_{mw}$. $pdr^e_{proc}(n_{mw}, rId)$, the observed boosted reliability on link $e$ for route $rId$ in moving window $n_{mw}$ is calculated as:

$$pdr^e_{proc}(n_{mw}, rId) = \frac{\sum_{n_{Bs}=n_{mw}-s_{mw}+1}^{n_{mw}} pdr^e_{proc}(n_{Bs,rId})}{s_{mw}} \tag{4.1}$$

where, $pdr^e_{proc}(n_{Bs,rId})$ is average PDR on link $e$ for route $rId$ during a Bernoulli slot $n_{Bs}$ without duplicates.

$$pdr^e_{proc}(n_{Bs,rId}) = \frac{\mid F^e_{rx,nod}(n_{Bs}, rId) \mid}{s_{Bs}}$$

where:

- $F^e_{rx}(n_{Bs}, rId)$ is the subset of received frames on link $e = (v, v')$ in Bernoulli slot $n_{Bs}$ carrying route identifier $rId$. It is defined as:

$$\begin{aligned} F^e_{rx}(n_{Bs}, rId) = \{ & f \in F_{rx} \mid \\ & f.snd = addr(v) \wedge \\ & f.rcv = addr(v') \wedge \\ & (n_{Bs} - 1) \cdot s_{Bs} + 1 \leq f.txCtr \leq n_{Bs} \cdot s_{Bs} \wedge \\ & f.rId = rId \} \end{aligned}$$

  where, $F_{rx}$ is the set of received frames observed during the entire experiment.

- $F^e_{rx,nod}(n_{Bs}, rId)$ is subset of $F^e_{rx}(n_{Bs}, rId)$ without any duplicates. It is defined as:

$$F^e_{rx,nod}(n_{Bs}, rId) = \{f \in F^e_{rx}(n_{Bs}, rId) \mid \forall f' \in F^e_{rx}(n_{Bs}, rId). \\ (f \neq f' \implies f.fId \neq f'.fId)\}$$

Throughout this chapter, the Bernoulli sequence (*event outcomes*) derived from the recorded data is referred to as the original sequence. Further, in our evaluation, we operate only one route per link. For instance, consider the following statistically independent Bernoulli sequence with a raw PDR of 0.75 in a moving window with a size $s_{mw}$ of 20 and the maximum number of transmissions $n^e_{max}$ of 1. Tab. 4.2

| ***txCtr*** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Event Outcomes** | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| ***fId*** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

Table 4.2: Bernoulli sequence with outcome of transmissions with *txCtr* between 1 and 20 for some link $e$ for $n^e_{max} = 1$.

shows the transmission counter value *txCtr*, event outcomes (represented as Bernoulli trials), and frame identifier *fId*. To boost the link reliability of a link that observed such event outcomes to a minimum target reliability $r^e_{target}$ of 0.93 , retransmissions of failed attempts are necessary. I.e., Step 1, we set $r^e_{target}$ to 0.93. Next, Step 2.1, we knew that the raw PDR ($pdr^e_{raw}$) or success probability of the sequence equals 0.75. Using Eqn. 3.1, if we assign a maximum of two transmission attempts per frame ($n^e_{max} = 2$) to a link with a minimum PDR of 0.75 (a success probability greater equal to 0.75), the link's reliability is boosted to 0.9375, i.e., Step 2.2. A single retransmission ($n^e_{max} = 2$) for a failed transmission increases the link reliability from 0.75 to 0.9375.

$$\begin{aligned} q^e_{n^e_{max}} &= 1 - (1 - q^e)^{n^e_{max}} \\ q^e_2 &= 1 - (1 - 0.75)^2 \\ &= 1 - (0.0625) \\ &= 0.9375 \end{aligned}$$

Next is Step 2.3, to calculate observed boosted reliability $pdr^e_{proc}(n_{mw}, rId)$. In practice, the *fID* of a frame is assigned by the sender. In our offline analysis, the *fID* value is modified depending on the evaluation requirements. For instance, Tab. 4.3 shows the scenario if the sender uses a maximum of two transmission attempts

| ***txCtr*** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Event Outcomes** | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| ***fId*** | $1^1$ | $2^1$ | $3^1$ | $4^1$ | $4^2$ | $5^1$ | $6^1$ | $7^1$ | $8^1$ | $9^1$ | $10^1$ | $11^1$ |

| ***txCtr*** | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|
| **Event Outcomes** | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| ***fId*** | $12^1$ | $12^2$ | $13^1$ | $14^1$ | $14^2$ | $15^1$ | $\underline{15^2}$ | $16^1$ |

Table 4.3: Scenario when retransmissions are applied ($n^e_{max} = 2$). Superscript denotes the number of transmission attempts.

per frame ($n^e_{max} = 2$) for each frame transmission. The green colour represents a successful transmission, while the red colour denotes an unsuccessful transmission. The superscript number inside *fId* cells counts the transmission attempts. Underlined *fId* in a red coloured cell shows a dropped frame that exceeds the maximum transmission attempts. In this example, out of 16 frames with unique *fId*, one failed to deliver. This results in an observed boosted reliability $pdr^e_{proc}$ of 0.9375.

To explore any possible correlation between *link boosting* and the results of a statistical test (certain statistical properties), we introduce the metric *Target Missed*. *Target Missed*, $r^e_{\Delta,missed}(n_{mw}, rId)$ in a moving window $n_{mw}$ for a route with identifier $rId$ is calculated as the difference between target reliability and observed boosted reliability. I.e.,

$$r^e_{\Delta,missed}(n_{mw}, rId) = \begin{cases} r^e_{target} - pdr^e_{proc}(n_{mw}, rId), & \text{if } r^e_{target} \geq pdr^e_{proc}(n_{mw}, rId) \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

As mentioned earlier, in our evaluation, we consider only one route per link. If $r^e_{\Delta,missed}(n_{mw})$ in a moving window $n_{mw}$ is negative, it implies observed boosted reliability is above target reliability. Since we focus on moving windows that failed to reach target reliability, we replace the values of $r^{\Delta}_{e,missed}$ below zero by zero. The *link boosting* for the window $n_{mw}$ is considered as:

$$\begin{cases} Success, & \text{if } r^e_{\Delta,missed}(n_{mw}) = 0 \\ Failure, & \text{if } r^e_{\Delta,missed}(n_{mw}) > 0 \end{cases}$$

In our evaluation, we apply each statistical test on the original sequences derived from event outcomes inside a moving window. For each moving window, we com-

pare the $r^{\Delta}_{e,missed}$ with the result of statistical tests. For instance, we compare the $r^{\Delta}_{e,missed}(n_{mw})$ during the moving window $n_{mw}$ with the *channel memory* $\mu$ of the original sequence.

## 4.2.2 Evaluation techniques

Here, we revisit the statistical methods that we apply to the data and discusses how we use them in our further evaluation.

### $\mu$-value for statistical independence

Sec. 3.3.1 discussed the discrete-time 2-state Markov model known as the Gilbert-Elliot model and its channel memory parameter $\mu$. We evaluate the correlation between the $\mu$ value and Target missed $r^{\Delta}_{e,missed}$ during each moving window in our evaluation. It helps to understand the change in the effectiveness of link boosting with respect to statistically dependent or bursty, statistically independent, and oscillatory event outcomes. The $\mu$ value is calculated as,

$$\mu = 1 - p_{gb} - p_{bg} \qquad \text{(Eqn. 3.2 revisited)}$$

where $p_{gb}$ and $p_{bg}$ are the transition probability from good state to bad state and from bad state to good state, respectively(see Sec. 3.3.1).

### Runs-test to measure randomness

In Sec. 3.3.2, we discussed the Wald-Wolfowitz Runs-test used to measure the randomness of a Bernoulli sequence. In our previous analysis, we used the p-value calculated from the $z$-value to decide whether the sequence is randomly generated or not. In our further analysis, instead of using the p-value, we compare the $z$-values with Target missed $r^{\Delta}_{e,missed}$ during each moving window. The p-value from a Runs-test provides evidence or probability to reject the null hypothesis that a random process generates the sequence. However, if we use the $z$-value from the Runs-test, we get more information about the randomness of the sequence. Also, comparing the p-values of multiple tests is invalid. Recalling the equation to calculate $z$-value,

$$z = \frac{r - \bar{r}}{\sigma_r} \qquad \text{(Eqn. 3.3 revisited)}$$

where $r$ is the observed number of runs, $\bar{r}$ is the expected number of runs, and $\sigma_r$ is the standard deviation of the number of runs. A negative $z$-value is observed if the sequence is "*undershuffled*", i.e., the observed number of runs is less than expected,

and a positive z-value is observed if the sequence is "*overshuffled*", i.e., the observed number of runs is greater than expected. It helps to understand the change in the effectiveness of link boosting with respect to the randomness of event outcomes.

#### Chi-square to measure the fitness with a Bernoulli distribution

We use the Chi-square goodness-of-fit test to evaluate whether the observed event outcomes fit a Bernoulli distribution with a certain success probability. In Sec. 3.3.3, we discuss the steps to perform the test. As discussed earlier, the test has several drawbacks. For instance, for a Bernoulli sequence, the test returns a different p-value depending on the degree of freedom. And, degree of freedom depends on the length of the sub-sequence ($k$). In our evaluation, we calculate the $chi_q$.

$$chi_q = 1 - \frac{\min(\mathrm{X}^2_{dof,table}, \mathrm{X}^2_{dof,obs})}{\mathrm{X}^2_{dof,table}} \qquad \text{(Eqn. 3.4 revisited)}$$

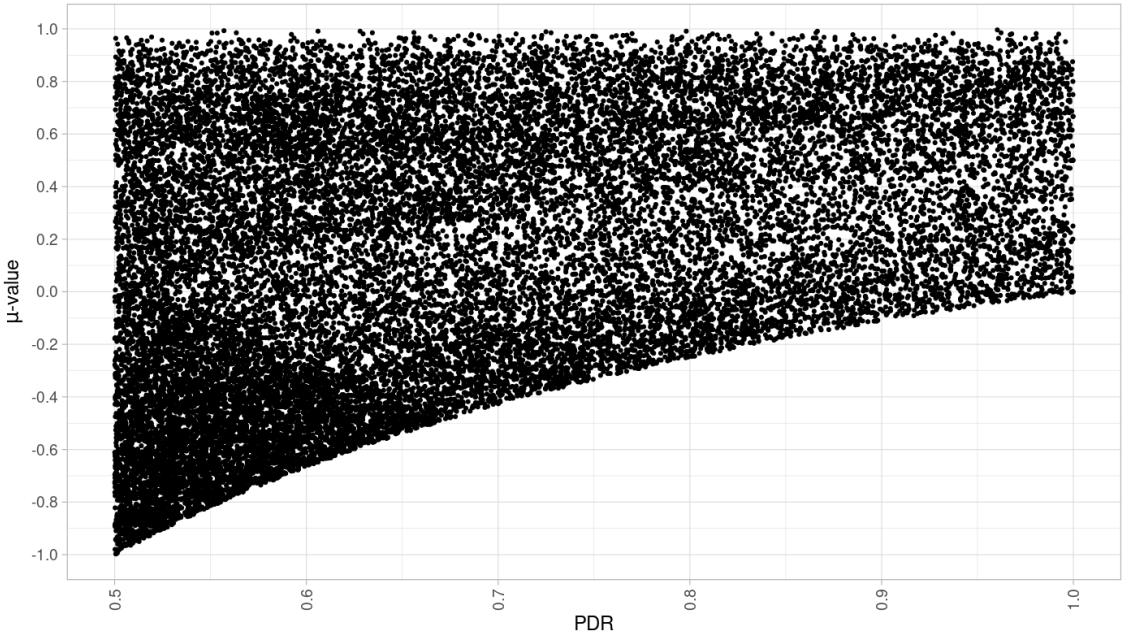
where $X^2_{dof,obs}$ is the observed Chi-square value with a degree of freedom $dof$, and $X^2_{dof,table}$ is Chi-square boundary with a degree of freedom $dof$ to reject the null hypothesis with a confidence interval of 95%. A higher value of $chi_q$ shows a stronger evidence for the establishment of the null hypothesis. In our evaluation, we compare the test results with multiple sub-sequence sizes (resulting in multiple degrees of freedom).

## 4.3 Experimental Assessment

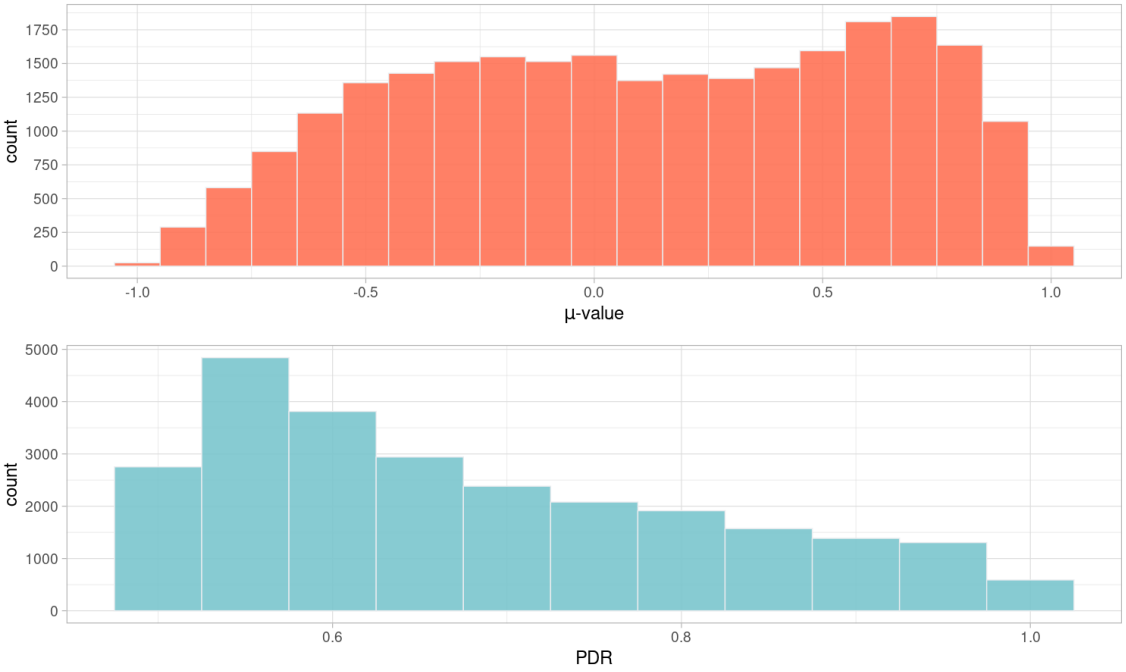
We use synthetic data and data collected from the testbed to study the relationship between the effectiveness of link boosting (in terms of $r^e_{\Delta,missed}$) and statistical properties of the Bernoulli sequence generated from transmission outcomes on a link. In this section, we discuss our test results and findings from analyzing the data. Sec. 4.3.1 evaluates the applicability of metrics (derived from the aforementioned statistical tests) to explore possible relationships. Later, in Sec. 4.3.2, we perform offline link boosting on the data collected from links in the testbed and compare the metric outcomes.

### 4.3.1 Evaluation using Synthetic data

Synthetic data used in our study is synthesized using the Markov model. The synthesized data set contains various Bernoulli sequences of size 20000 and $\mu$-values ranging from -1 to 1. In our evaluation, we consider the Bernoulli sequences with




(a) Each point denotes the success probability or raw PDR and the $\mu$-value of synthesized Bernoulli sequences (of size 20000) in our evaluation data set.




(b) Histograms showing the count of Bernoulli sequences in our evaluation data set with respective $\mu$-values and success probability or raw PDR.

Figure 4.1: Characteristics of the synthesized Bernoulli sequences in our evaluation data set.

a success probability (or raw PDR $pdr^e_{raw}$) higher than 0.50. The filtered data set contains more than 18000 Bernoulli sequences with broader statistical characteris-

tics. Fig. 4.1 shows the characteristics of the synthesized data used in our study. Fig. 4.1a shows the success probability of a synthesized Bernoulli sequence and its $\mu$-value. Further, Fig. 4.1b shows the histograms showing the distribution of synthesized Bernoulli sequence in terms of $\mu$-values and success probability.

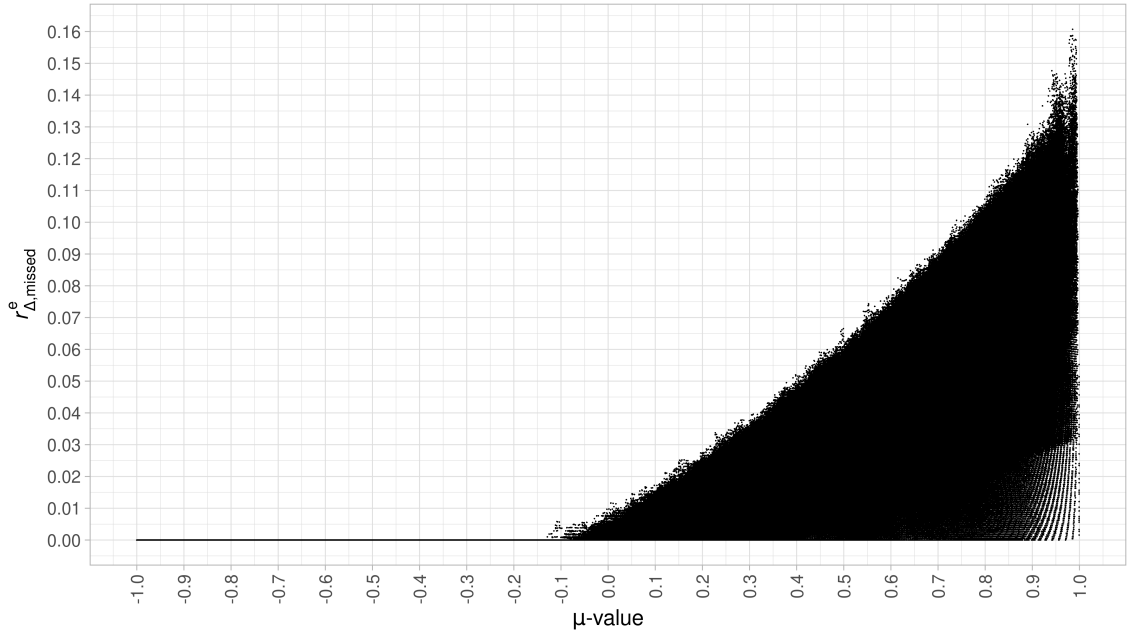
We use multiple moving window settings in our analysis. In the following part, we discuss the results from applying a moving window configuration of $s_{Bs} = 30$ and $s_{mw} =$60, and $s_{Bs} = 30$ and $s_{mw} = 100$, resulting in 1800 and 3000 Bernoulli trials in a moving window, respectively. In our evaluation, we set a target reliability $r^e_{target}$ of 0.98, and we only consider the moving windows where the success probability (observed raw PDR) is below 0.979. The moving window with the lowest success probability has a success probability of 0.50 and required up to 6 transmissions ($n^e_{max} = 6$) to meet the target reliability $r^e_{target}$. Next, we discuss the results from evaluating the role of statistical independence and the relationship between $\mu$-value and the effectiveness of link boosting. Following that, we discuss the findings using $z$-value and $chi_q$ on synthetic data.

#### 4.3.1.1 Using $\mu$-value to understand the role of statistical independence for link boosting

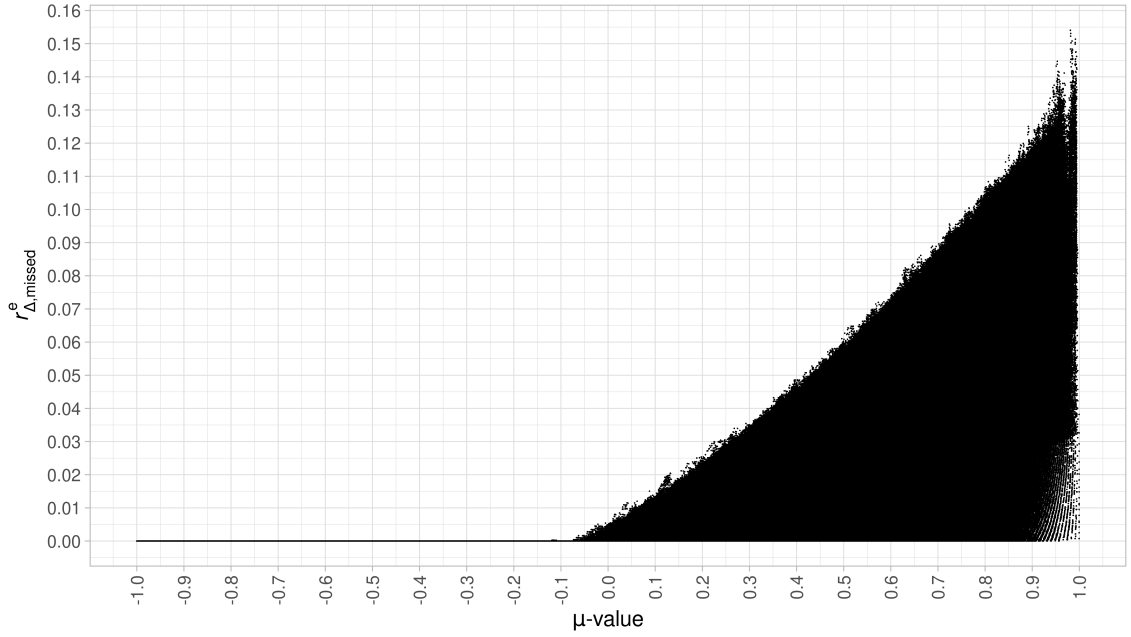
We start by examining the relationship between the $\mu$-value and the effectiveness of link boosting during each moving window, i.e., $r^e_{\Delta,missed}(n_{mw})$. Fig. 4.2 plots the $\mu$-value and Target missed $r^e_{\Delta,missed}$ during each moving window. Fig. 4.2a and Fig. 4.2b plot the results using moving windows with $s_{Bs} = 30$ and $s_{mw} =$60, and $s_{Bs} =$ 30 and $s_{mw} = 100$, respectively. These configurations result in more than 10 million moving windows or data points for our evaluation.

The link boosting is successful, or $r^e_{\Delta,missed} = 0$ on almost all sequences with a $\mu$-value close to and below 0, i.e., for those with statistically independent and oscillatory event outcomes. Notably, link boosting is successful for all sequences with a $\mu$-value below -0.1 or oscillatory event outcomes. These preliminary results show that the reliability of a link with the statistically independent, particularly on oscillatory event outcomes, can be increased by retransmissions. Oscillatory sequences are not statistically independent and may or may not be random. We further discuss oscillatory sequences when examining the relationship between $z$-values from the Runs-test and link boosting.

Furthermore, plots show that the Target missed $r^e_{\Delta,missed}$ increases as the $\mu$ value increases, i.e., the link boosting fails, and the margin of failure increases as the $\mu$-value increases. However, there are windows where the $\mu$-values are close to 1 (implying bursty event outcomes), and the link boosting is successful, i.e., $r^e_{\Delta,missed} =$




(a) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 60$.



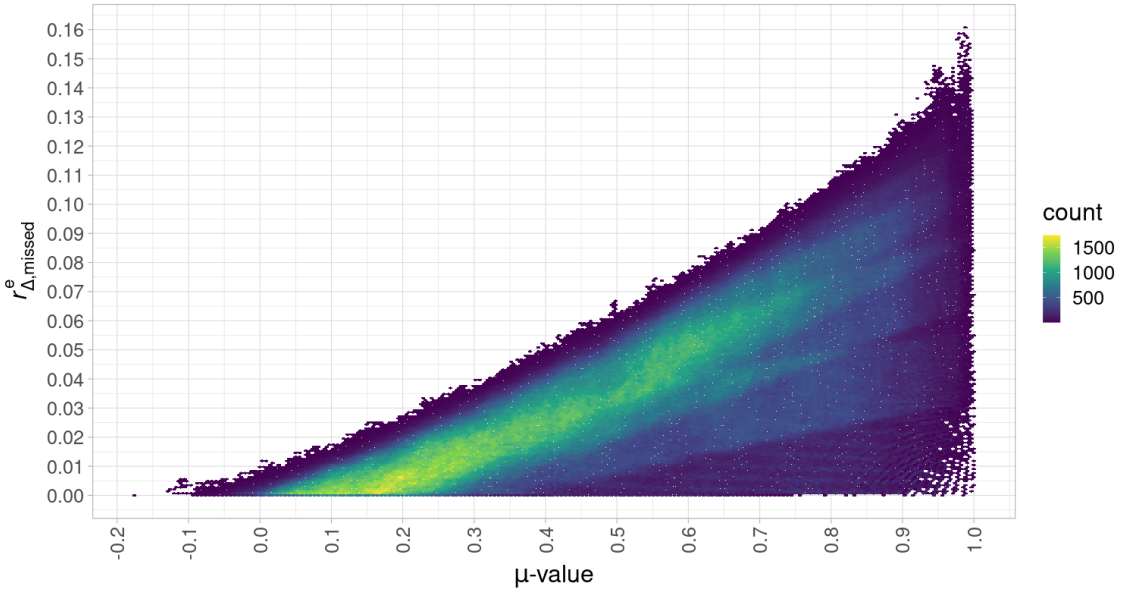

(b) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 100$.

Figure 4.2: Plots showing the $\mu$-value and Target missed $r^e_{\Delta,missed}$ in each moving window from the entire data set.

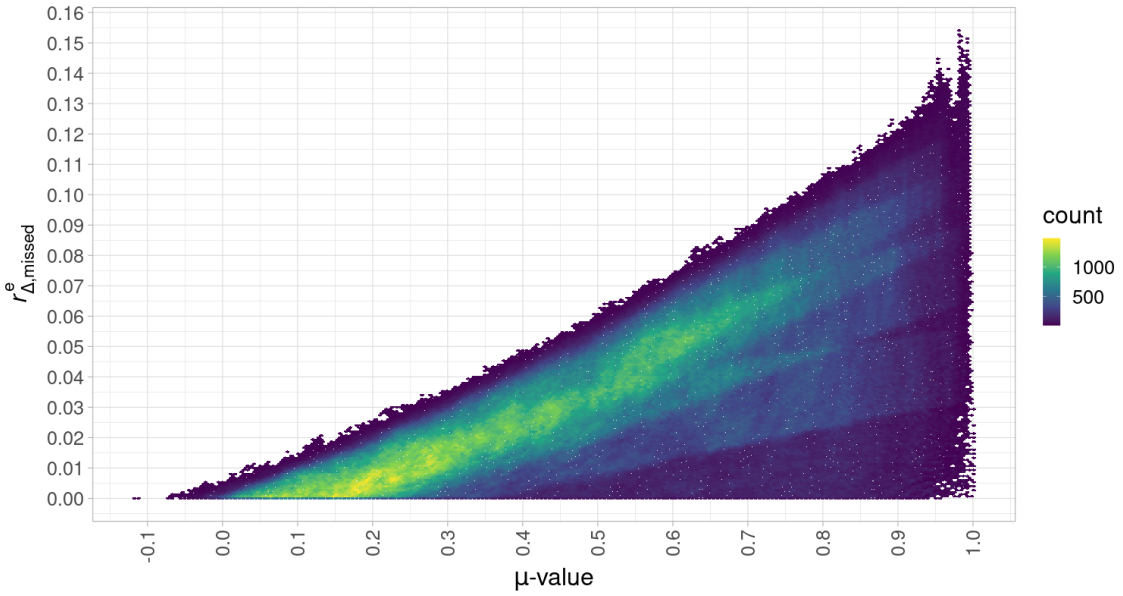
0. To help us comprehend the results better, we further filter the data by removing the windows where Target missed $r^e_{\Delta,missed}$ equals zero. Fig. 4.3 shows the heat plot (or heat map) of Target missed $r^e_{\Delta,missed}$ and $\mu$-values for moving windows where link boosting fails, i.e., $r^e_{\Delta,missed} > 0$. The brighter the points in the plot, the higher

the number of moving windows or sequences with the same result. Both window configurations show the same pattern as the $\mu$-value increases, the Target missed $r^e_{\Delta,missed}$ too increase. Furthermore, we also observed a similar pattern using other window configurations.

After grouping the moving windows according to their $\mu$-values, Fig. 4.4 displays the box plot of target missed $r^e_{\Delta,missed}$ for each group. The plot only contains the




(a) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 60$.




(b) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 100$.

Figure 4.3: Plots showing the $\mu$-value and Target missed $r^e_{\Delta,missed}$ in moving windows where the $r^e_{\Delta,missed} > 0$. I.e., windows were link boosting failed.

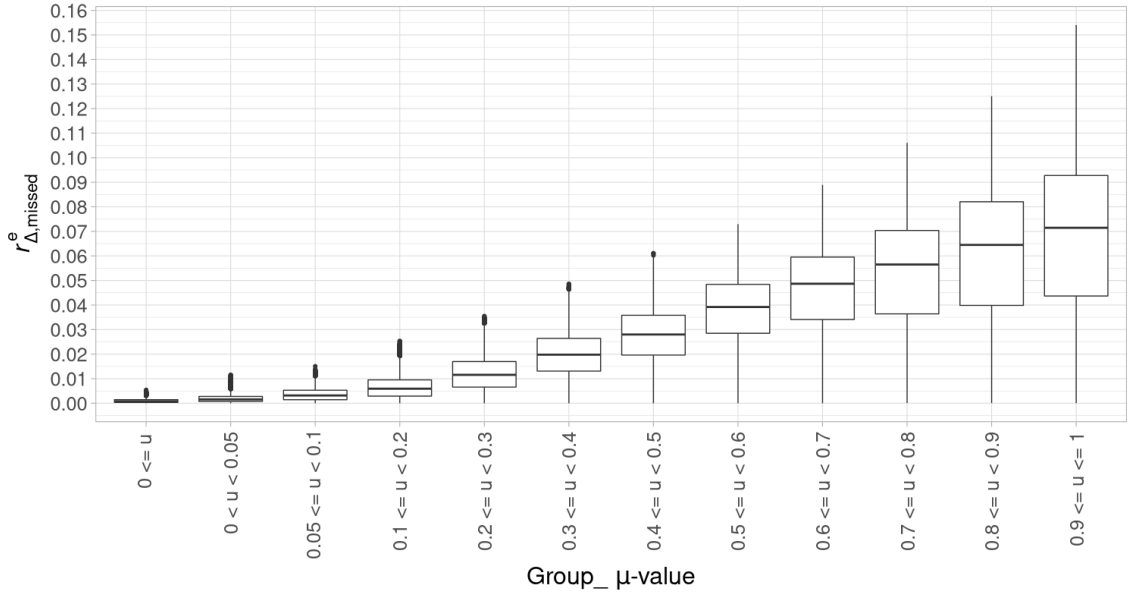



Figure 4.4: Box plot showing the data distribution of Target missed $r^e_{\Delta,missed}$ in moving windows for a specific range of $\mu$-values where $r^e_{\Delta,missed} > 0$.

result where the link boosting was a failure ($r^e_{\Delta,missed} > 0$). For instance, in Fig. 4.4, the x-axis with label "$0 < \mu < 0.05$" and "$0.05 <= \mu < 0.1$" shows the $r^e_{\Delta,missed}$ for the moving windows where the $\mu$-values during the windows are $0 < \mu < 0.05$ and $0.05 \leq \mu < 0.1$, respectively. In the set of moving windows where the $0 < \mu < 0.05$, windows have a median Target missed $r^e_{\Delta,missed}$ close to 0 ($\approx 0.001$). Similarly, for the set of moving windows where the $0.4 \leq \mu < 0.5$ (in the plot with label "$0.4 <= \mu < 0.5$") have a median $\mu$-value of 0.027. These results can be used to select a $\mu$-value boundary to categorize the links depending on the acceptable margin of error. For instance, if the acceptable worst-case margin of error for target missed is below 0.015, a $\mu$-value of 0.1 can be set as the boundary. Seldom, these worst-case occur. For a $\mu$-value of 0.1, in an unsuccessful link boosting, which rarely occurs (probability of less than 2%), the observed median target missed $r^e_{\Delta,missed}$ is 0.0025.

However, some of the questions remain unanswered. Why do some windows or sequences with higher $\mu$-value have successful link boosting? Why do windows with the same $\mu$-value observe different Target missed $r^e_{\Delta,missed}$? Recall that in this evaluation, we set the target reliability $r^e_{target}$ to 0.98. To achieve a minimum target reliability, windows with a raw PDR $pdr^e_{raw}$ between 0.859 and 0.979 require two transmissions, or $n^e_{max} = 2$. Meanwhile, a sequence with raw PDR below 0.859 (and higher than 0.73) requires three transmissions. Consider the former, where the windows require two transmissions to meet the target; a window (or Bernoulli sequence)

with raw PDR 0.93 has a higher "margin of safety" [13] than a window with raw PDR 0.86. The "margin of safety" $r^e_{mos}(n_{mw})$ in a moving window $n_{mw}$ is defined as the difference between boosted link reliability and minimum target reliability. i.e.,

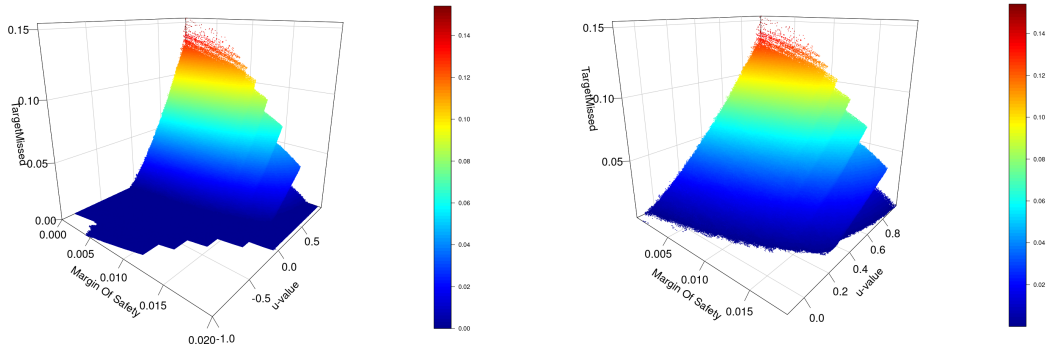
$$r^e_{mos}(n_{mw}) = q^e_{n^e_{max}}(n_{mw}) - r^e_{target} \tag{4.3}$$

For instance, the "margin of safety" for a window with raw PDR 0.93 and meets the minimum target reliability of 0.98 is 0.0151.

$$\begin{aligned} r^e_{mos} &= (1 - (1 - 0.93)^2) - (0.98) \\ &= 0.9951 - 0.98 \\ &= 0.0151 \end{aligned}$$

Similarly, the "margin of safety" for a window with raw PDR of 0.86 is 0.0004.

$$\begin{aligned} r^e_{mos} &= (1 - (1 - 0.86)^2) - (0.98) \\ &= 0.9804 - 0.98 \\ &= 0.0004 \end{aligned}$$




(a) Windows from the entire data.

(b) With windows where $r^e_{\Delta,missed} > 0$.

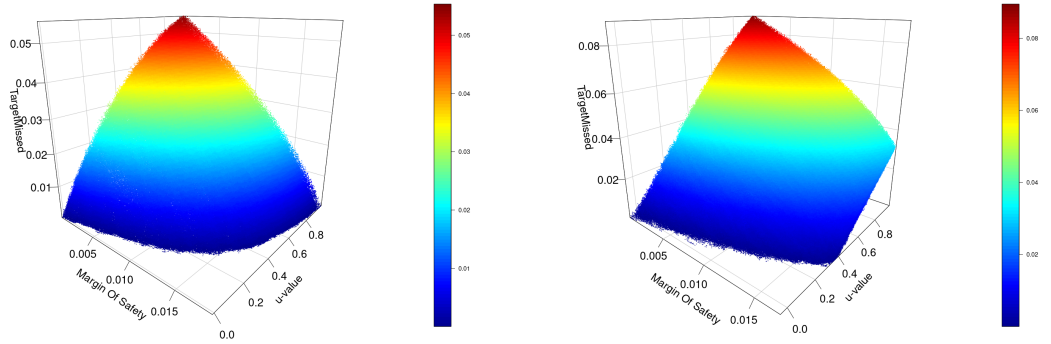
Figure 4.5: 3D Scatter plot showing Target missed $r^e_{\Delta,missed}$ (z-axis), the margin of safety $r^e_{mos}$ (y-axis), and the $\mu$-value (x-axis) during each moving window.

[13] The term "margin of safety" was first introduced in the field of investing by Benjamin Graham [Gra03].

In our evaluation, the windows or sequences with a $\mu > 0.5$ and successful link boosting are windows with a very high "margin of safety". For instance, with windows were $\mu > 0.5$ and $r^e_{\Delta,missed} = 0$, the filtered data set contains no windows with a raw PDR below 0.94. It means that link boosting on a window or sequence with a higher "margin of safety" can tolerate weakness in statistical properties. It also means that the windows or sequences with a lower "margin of safety" require stronger statistical properties for successful link boosting. For instance, considering the windows with raw PDR between 0.859 and 0.979 (where $n^e_{max}$ of 2 to meet a $r^e_{target}$ of 0.98), successful link boosting of a window with a raw PDR $pdr^e_{raw}$ of 0.86 requires a $\mu$-value below 0.05, i.e., a strong statistical independence. Meanwhile, a window with raw PDR of 0.92 requires can tolerate to $\mu$-value up to 0.20, i.e., it will tolerate a small degree of statistically dependent event outcomes.

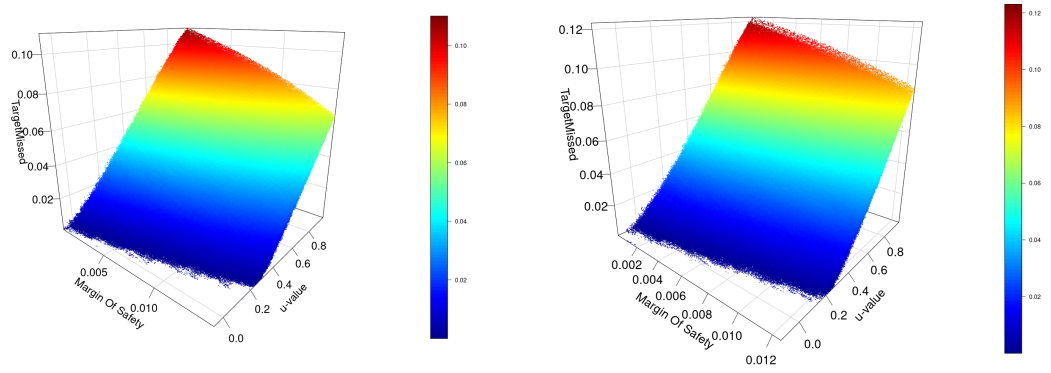
Fig. 4.5a shows the Target missed $r^e_{\Delta,missed}$, the margin of safety $r^e_{mos}$, and the $\mu$-value during each moving window (for the entire data set). The range of the "margin of safety" axis depends on the $r^e_{target}$. In this evaluation, it is between 0 and 0.02, i.e., between 0 and $1 - r^e_{target}$. Moreover, Fig. 4.5b shows the windows where link boosting failed i.e., $r^e_{\Delta,missed} > 0$. Supporting our verdicts, as the "margin of safety" increases, the windows tolerate a small degree of bursty event outcomes. For instance, link boosting is successful for all windows with a "margin of safety" greater than 0.015 and a $\mu$-value below 0.2. I.e., windows with a "margin of safety" higher than 0.015 can tolerate up to a $\mu$-value of 0.2. Conversely, the windows with a small "margin of safety" require strong statistical independence (a $\mu$-value close to 0).

Another observation is the multiple "layers" in Fig. 4.5. Fig. 4.6 shows the Target missed $r^e_{\Delta,missed}$, the margin of safety, and the $\mu$-value of windows, depending on the maximum number of transmissions, $n^e_{max}$. For instance, Fig. 4.6a plots the windows where the maximum transmissions required to meet the target reliability ($r^e_{target} = 0.98$), $n^e_{max}$ is 2. I.e., the windows where the raw PDR is between 0.86 and 0.979. Similarly, Fig. 4.6b, Fig. 4.6c, and Fig. 4.6d plot the windows where the maximum transmissions required to meet the target reliability ($r^e_{target} = 0.98$), $n^e_{max}$ is 3, 4, and 5, respectively. The range of the z-axis, Target missed $r^e_{\Delta,missed}$ increases as the the maximum transmissions required to meet the target, $n^e_{max}$ increases. For windows that require two transmissions to meet the target reliability ($n^e_{max} = 2$), the observed upper bound of target missed is 0.055 (see Fig. 4.6a ). While for windows that require $n^e_{max}$ of 3, 4, and 5, the observed upper bound ranges to 0.085, 0.105, and 0.122, respectively. As the raw PDR decreases, the difference between raw PDR and target reliability increases, resulting in increased target missed in absolute number. This raises the followings questions. Is it possible to normalize the Target missed




(a) Windows that require $n^e_{max} = 2$

(b) Windows that require $n^e_{max} = 3$




(c) Windows that require $n^e_{max} = 4$

(d) Windows that require $n^e_{max} = 5$

Figure 4.6: 3D Scatter plot showing Target missed $r^e_{\Delta,missed}$ (z-axis), the margin of safety $r^e_{mos}$ (y-axis), and the $\mu$-value (x-axis) during each moving window depending on $n^e_{max}$.

$r^e_{\Delta,missed}$ for windows with different $n^e_{max}$, between 0 and 1? Similarly, is it possible to normalize the "margin of safety" for different experiment settings, i.e., depending on the $r^e_{target}$, as the range varies? Thus, is it possible to derive an equation (or a system model) to find the tolerable range of $\mu$-value or weakness in statistical independence depending on the raw PDR $pdr^e_{raw}$ and the minimum target reliability $r^e_{target}$? We will answer these questions in Sec. 4.4.

Concluding our results, empirical evaluation of synthesized data shows that, irrespective of the "margin of safety", successful link boosting requires that the transmission event outcomes be either statistically independent or oscillatory. The results show that as the event outcomes become more statistically dependent, it is more likely that the link boosting becomes unsuccessful, and the possibility for a higher target missed $r^e_{\Delta,missed}$ depending on the "margin of safety". Based on our findings,

a correlation exists between the effectiveness of link boosting and the $\mu$-value, i.e., statistical independence property. $\mu$-value is a potential candidate to filter suitable links for routing. Further, moving windows tolerate a small weakness in the statistical independence property depending on the "margin of safety" and target reliability $r^e_{target}$. We study further on this relationship in the coming section.

#### 4.3.1.2 Using *z*-value to understand the role of randomness in link boosting

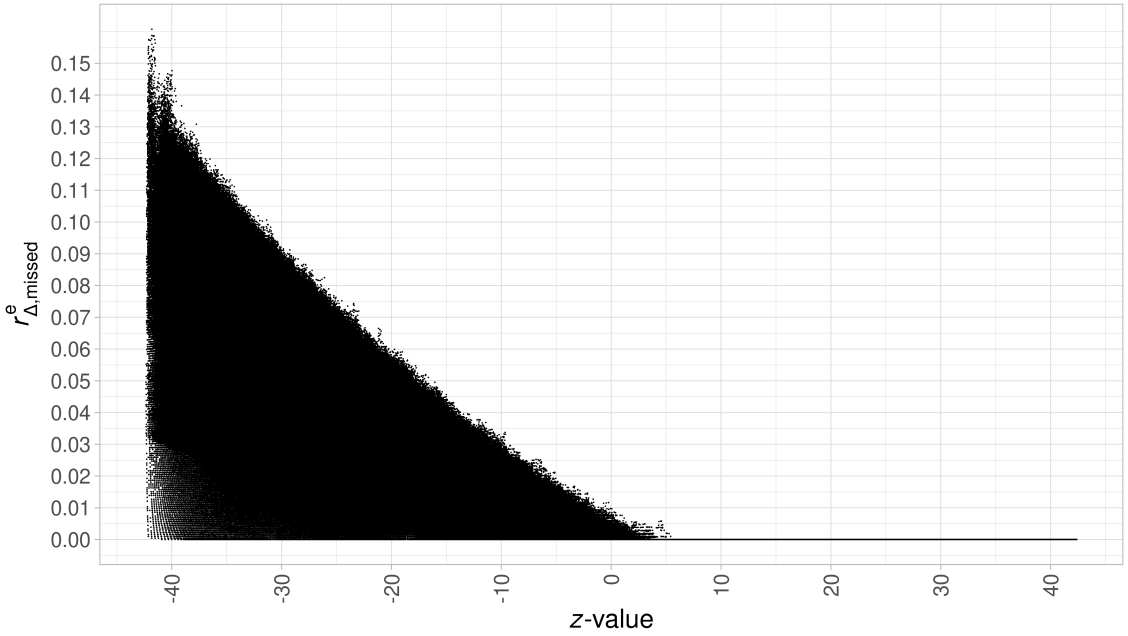
Next, we discuss the relationship between the $z$-value calculated from the Runs-test and link boosting effectiveness for each moving window. Likewise, in the previous evaluation, we used two moving window configurations. Fig. 4.7a and Fig. 4.7b shows result using moving windows with $s_{Bs} = 30$ and $s_{mw} =$60, and $s_{Bs} = 30$ and $s_{mw} = 100$, respectively. The plot's $z$-value range (x-axis range in the plots) differs, as it depends on the length of the Bernoulli sequence. The higher the sample length, the higher the $z$-value range.

Nonetheless, the Runs-test rejects the null hypothesis with 95% confidence interval for a $z$-value below -1.96 and above 1.96. I.e., a window or sequence with $\mid z \mid < 1.96$ or, $-1.96 < z < 1.96$ is considered random. Fig. 4.7 shows that for two different moving configurations, except for few outliers (see Fig. 4.8), the link boosting was successful for most sequences with $\mid z \mid < 1.96$ ($-1.96 < z < 1.96$ ). Notably, link boosting was successful for all sequences with a positive $z$-value (or $z > 1.96$), i.e., "*overshuffled*" sequences, such that the observed number of runs is higher than the expected number of runs. Conversely, for most sequences, as the $z$-value becomes negative (or $z < -1.96$), i.e., for "*undershuffled*", link boosting becomes a failure, and the Target missed $r^e_{\Delta,missed}$ increases. These results are comparable to the results from our previous evaluation, i.e., comparing the $\mu$-value and target missed $r^e_{\Delta,missed}$ during each window.

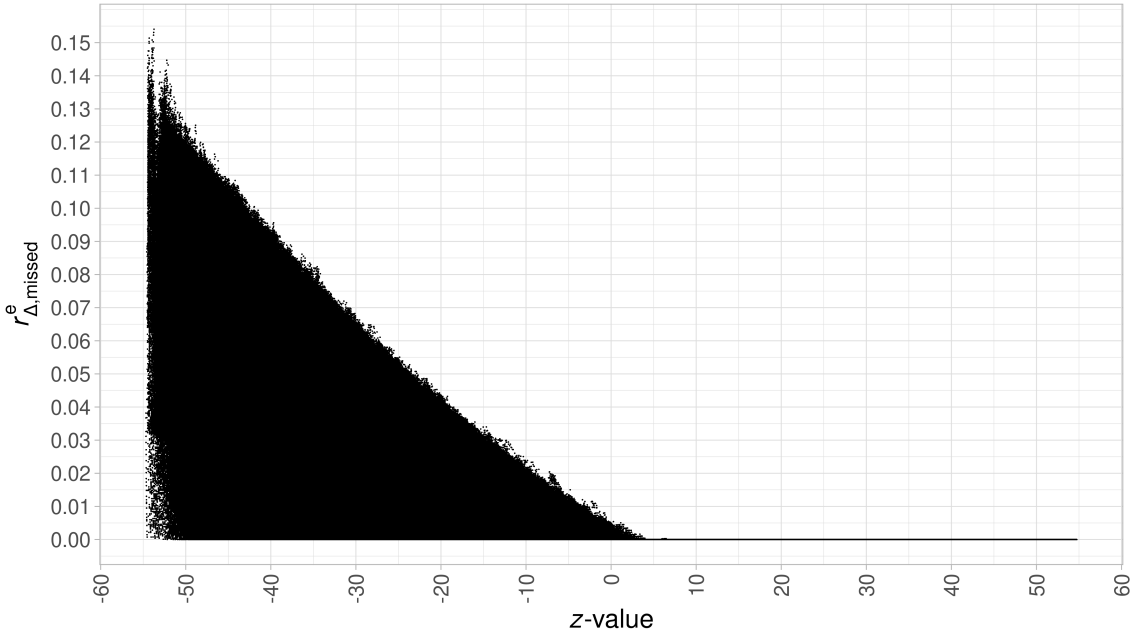
Similar to Fig. 4.4, after grouping the moving windows according to their $z$-values, Fig. 4.8 displays the box plot of target missed $r^e_{\Delta,missed}$ for each group. Further, Fig. 4.8 excludes the windows where link boosting is successful. The result are comparable with the results from box plot of $\mu$-values. As the $z$-value decreases, the margin of error from link boosting increases. Moreover, the outliers, i.e., the windows with successful link boosting and "*undershuffled*" (low or negative $z$-values), to an extent the high "margin of safety", tolerated the weakness of random outcome.

Fig. 4.9 shows the $\mu$-value and $z$-value calculated during moving windows. The plot shows that there exists a strong correlation between both. Especially, there exists a strong correlation between sequences with negative $\mu$-values (0 to -1) and positive $z$-values, i.e., between oscillatory and "*overshuffled*" sequences. We ob-

served the same result (correlation) for different moving configurations. Further, the sequences where the $\mid z \mid < 1.96$ (the region where evidence to reject the null hypothesis is below the confidence interval) observed a $\mu$-value between -0.005 and




(a) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 60$.




(b) Applying a moving window configuration, $s_{Bs} = 30$ and $s_{mw} = 100$.

Figure 4.7: Plots showing the $z$-value and Target missed $r^e_{\Delta,missed}$ in each moving window.

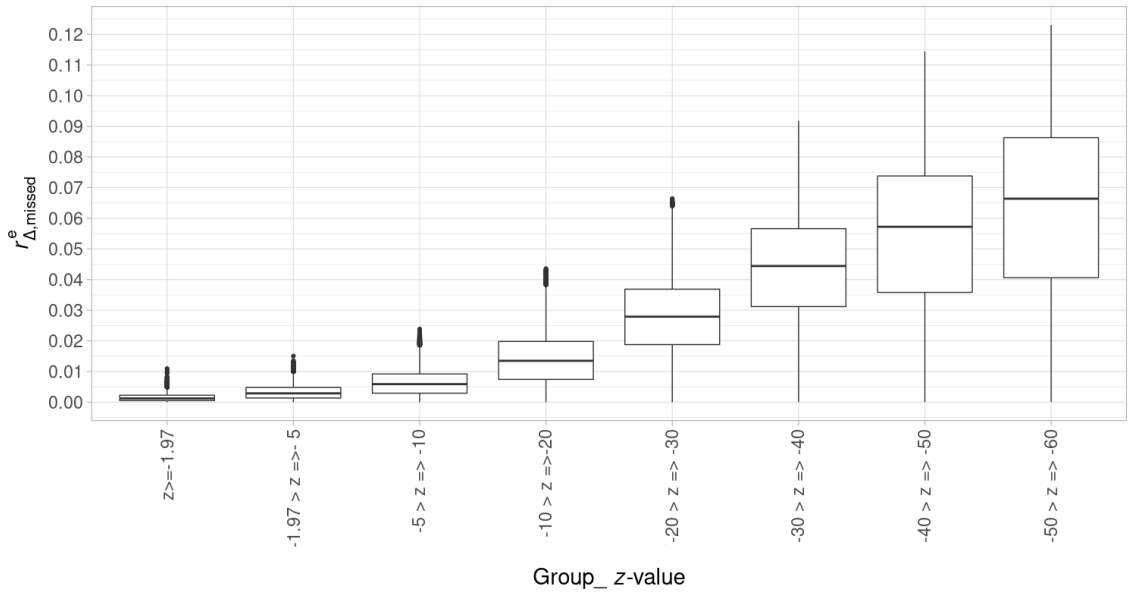



Figure 4.8: Box plot showing the data distribution of Target missed $r^e_{\Delta,missed}$ in moving windows for a specific range of $z$-values where $r^e_{\Delta,missed} > 0$.

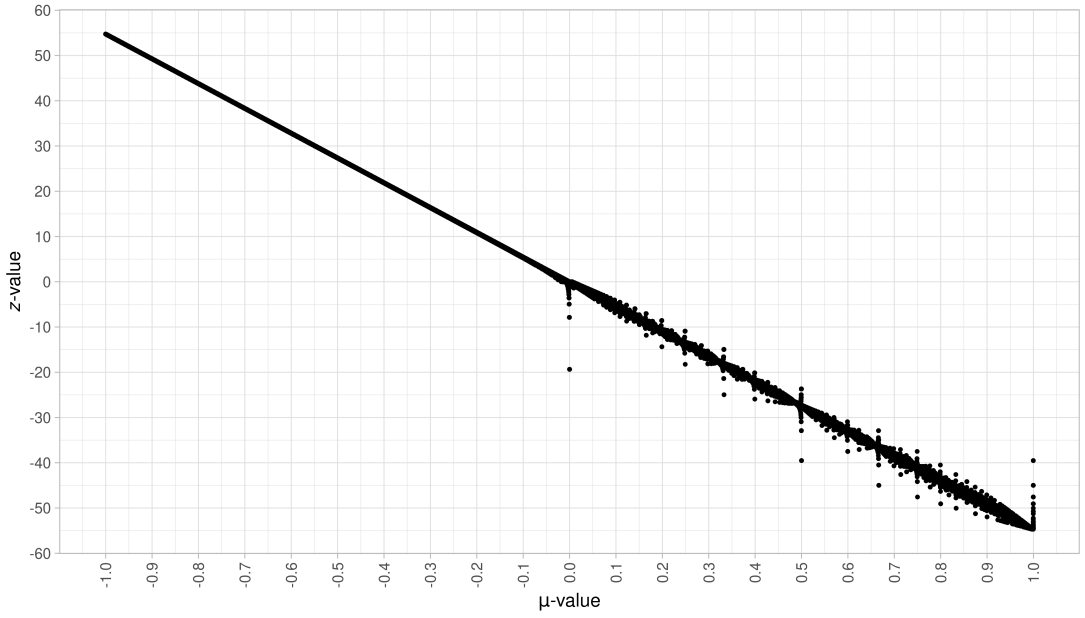



Figure 4.9: Correlation between the $\mu$-value and $z$-value during an evaluation with moving window configurations, $s_{Bs} = 30$ and $s_{mw} = 100$.

0.005. These results show the relationship between randomness and statistical independence.

Our evaluation shows a relationship between the degree of randomness and link boosting. It is similar to the results from the study on statistical independence,

i.e., the relationship between $\mu$-values and link boosting. Link boosting is successful on sequences with a $\mid z \mid < 1.96$ (or random sequences). Notably, link boosting is successful on all sequences with a $z \geq 1.96$. Even though the sequence with a $z \geq 1.96$ is not considered as "random" by the Runs-test, the "*overshuffling*" property (higher number of runs than expected) results in successful link boosting. Further, we found a strong correlation between the $z$-value and $\mu$-value, i.e., randomness (in terms of the number of runs) and statistically independent event outcomes.

#### 4.3.1.3 Using $chi_q$ to understand the role of sequences' fitness to a Bernoulli distribution for a successful link boosting

In this part, we examine the relationship between the observations from the Chi-square goodness-of-fit test and link boosting. To examine the role, we use the metric $chi_q$, calculated from the Chi-square value of the test, where a higher value for $chi_q$ indicates stronger evidence for establishing the null hypothesis, i.e., the sequence is drawn i.i.d from a Bernoulli distribution with success probability $p$. Fig. 4.10 shows the $chi_q$ and Target missed $r^e_{\Delta,missed}$ during each moving window with a window configuration, $s_{Bs} = 30$ and $s_{mw} = 100$. We observed similar results using other window configurations, and therefore, exclude those results in further discussions. We categorize the results from Fig. 4.10 into two parts: 1) moving windows where $0 < chi_q \leq 1$ referred to as "$chi_q$ greater than 0", and 2) moving windows where $chi_q = 0$. The former, $chi_q$ greater than 0 implies the establishment of the null

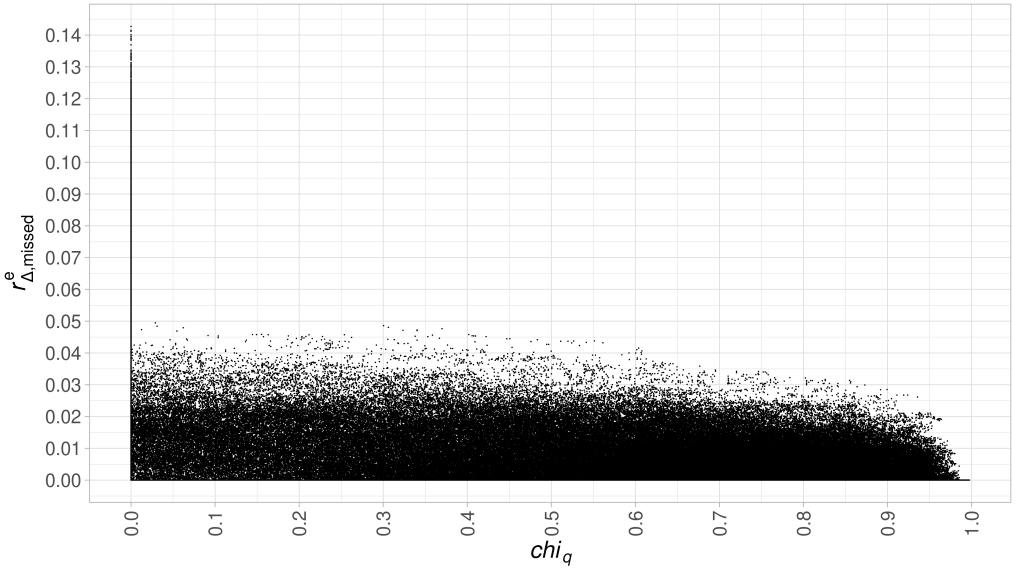



Figure 4.10: Plot showing the $chi_q$ and Target missed $r^e_{\Delta,missed}$ during each moving window with a configuration, $s_{Bs} = 30$ and $s_{mw} = 100$.

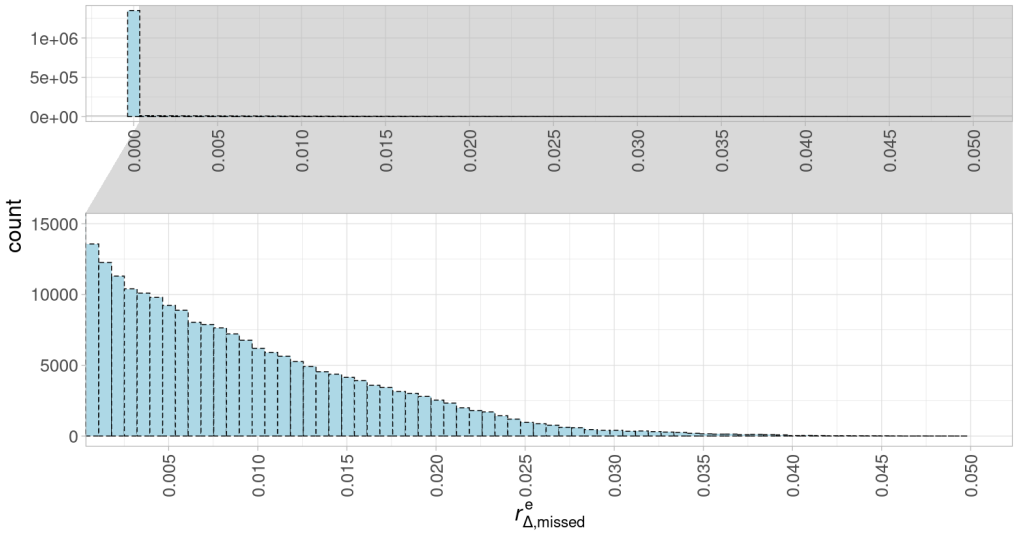



Figure 4.11: Histogram showing the distribution of Target missed $r^e_{\Delta,missed}$ in moving windows with a configuration, $s_{Bs} = 30$ and $s_{mw} = 100$, where $chi_q > 0$ .

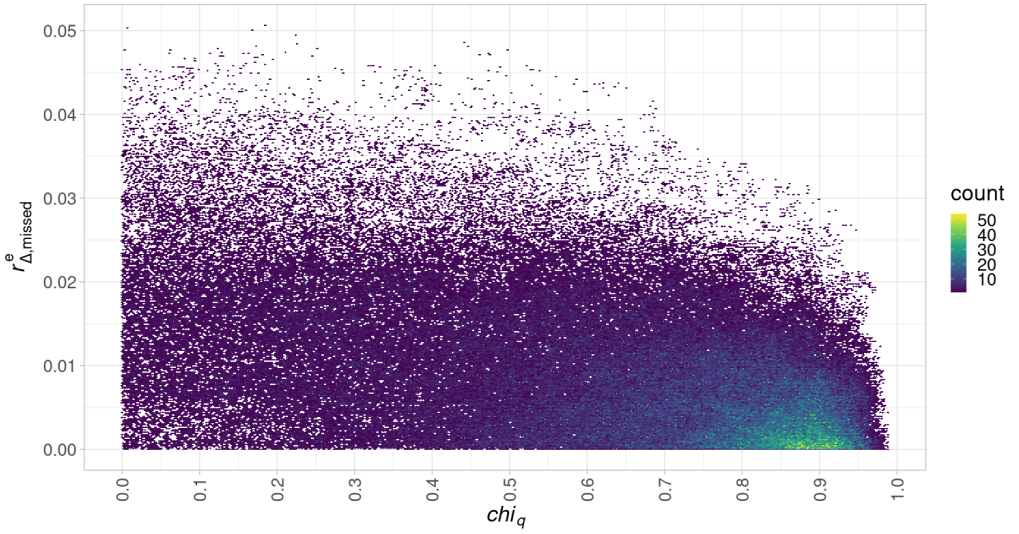



Figure 4.12: Plots showing the $chi_q$ and Target missed $r^e_{\Delta,missed}$ in moving windows where the $r^e_{\Delta,missed} > 0$ and $chi_q > 0$. I.e., windows were link boosting failed. Moving window uses a configuration, $s_{Bs} = 30$ and $s_{mw} = 100$.

hypothesis with a 95% confidence interval. And the latter, $chi_q$ equal 0 implies evidence to reject the null hypothesis with a 95% confidence interval. Furthermore, the higher value of $chi_q$ the stronger evidence for the establishment of the null hypothesis. Next, we discuss the findings of the former subset followed by the latter.

Fig. 4.10 contains more than 3 million sample points or moving window results. Of those 3 million data points, 51% of the moving windows recorded a $chi_q$ greater than

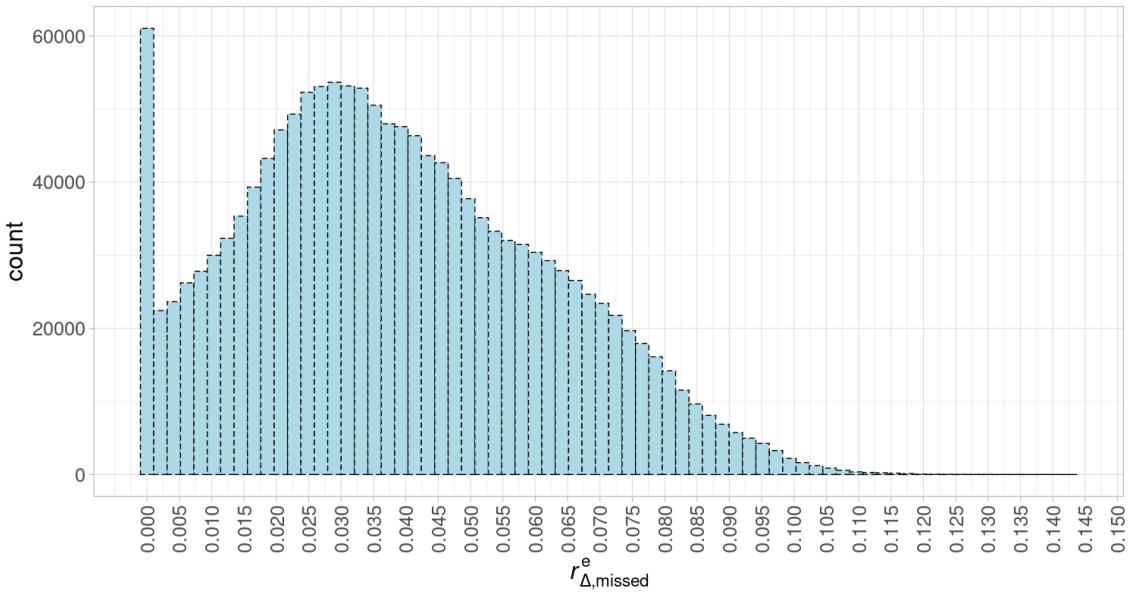



Figure 4.13: Histogram showing the distribution of Target missed $r^e_{\Delta,missed}$ in moving windows with a configuration, $s_{Bs} = 30$ and $s_{mw} = 100$, where $chi_q = 0$.

0. Among those windows (where $0 < chi_q \leq 1$), the link boosting was successful ($r^e_{\Delta,missed} = 0$) for 86% of the moving windows. The remaining 14% recorded a Target missed $r^e_{\Delta,missed}$ in the range, $0 < r^e_{\Delta,missed} < 0.05$. Fig. 4.11 shows the histogram for the data distribution of Target missed $r^e_{\Delta,missed}$ for windows where $chi_q$ is greater than 0. The magnified region shows the data distribution for the 14%, where the Target missed is greater than 0, $r^e_{\Delta,missed} > 0$. In those windows or sequences, the vast majority of the windows observed a Target missed $r^e_{\Delta,missed}$ close to 0. Moreover, the number of windows with higher target misses is seldom. Fig. 4.12 shows the heat plot for the moving windows where $chi_q$ and $r^e_{\Delta,missed}$ is greater than 0, i.e., $0 < chi_q \leq 1$ and $r^e_{\Delta,missed} > 0$. The figure shows that as the $chi_q$ values increases, the probability of having target miss close to 0 increases. I.e., in case of failed link boosting for a moving window with a $chi_q$ value greater than 0.7, it is more likely of having a target missed $r^e_{\Delta,missed}$ below 0.01 than a moving window with a $chi_q$ value below 0.7.

Of those 3 million data points, the remaining 49% of the moving windows recorded a $chi_q$ equals 0. Among those windows, only 3% of the windows recorded a target missed equals 0. I.e., the link boosting failed ($r^e_{\Delta,missed} > 0$) for 97% of the moving windows where $chi_q$ equals 0. Fig. 4.13 shows the histogram for the data distribution of target missed for the windows where the $chi_q$ equals 0.

| Tolerance level | Success ratio for windows with $0 < chi_q \leq 1$ | Success ratio for windows with $chi_q = 0$ |
|---|---|---|
| $r^e_{\Delta,missed} = 0$ | 86% | 3% |
| $r^e_{\Delta,missed} \leq 0.005$ | 90% | 6% |
| $r^e_{\Delta,missed} \leq 0.01$ | 94% | 10% |
| $r^e_{\Delta,missed} \leq 0.02$ | 98% | 22% |

Table 4.4: The table shows the percentage of moving windows (out to complete data set) with successful link boosting for different tolerance levels.

Comparing the two histograms, Fig. 4.11 (windows with $chi_q > 0$) and Fig. 4.13 (windows with $chi_q = 0$), and from other evaluations using different moving window configurations, the following are our conclusions:

- When $0 < chi_q \leq 1$, i.e., inadequate evidence to reject the null hypothesis with 95% confidence, link boosting was successful for most windows (more than 85% windows); however, the link boosting failed for 15% of the Bernoulli sequences or moving windows, with a target missed $r^e_{\Delta,missed}$ measures up to 0.05.

- When $chi_q = 0$, i.e., strong evidence (with 95% confidence) to reject the null hypothesis, less than 3% of the windows recorded a successful link boosting. Further, sequences or moving windows with $chi_q$ =0 shows a higher likelihood for a failed link boosting by a higher margin of error.

What will be the numbers if we tolerate a target miss up to 0.005? Implies that we tolerate a Target missed $r^e_{\Delta,missed}$ below 0.005, i.e., $r^e_{\Delta,missed} \leq 0.005$. Tab. 4.4 shows the values for multiple tolerance levels using a moving window configuration $s_{Bs}$ =30 and $s_{mw} = 100$. For instance, for a tolerance level of 0.005 or $r^e_{\Delta,missed} \leq 0.005$, the link boosting is successful for 90% of the moving windows where the $chi_q$ is greater than 0. Further, the link boosting is successful for 6% of the moving windows where the $chi_q$ equals 0. We also observed similar results for other moving configurations.

In our evaluation, we observe a relationship between the Chi-square test results, $chi_q$ and link boosting. In the windows where the test fails to show the evidence to reject the null hypothesis with 95% confidence or $chi_q$ greater than 0, the link boosting is successful for most of the windows. However, for moving windows or sequences where link boosting and the $chi_q$ is greater than 0, the margin of error ranges up to 0.05. Conversely, in the windows with $chi_q$ equals 0, only a small percentage (up to 3%) of the windows recorded a successful link boosting.

Empirical analysis using synthetic data shows the importance of statistical independence and randomness for link boosting. Further, our results show the relation-

ship between randomness and statistical independence in this particular scenario. By using the metrics discussed above, it is possible to select links that fit the requirements of a network. In Sec. 4.4, we further examine the tolerance range for statistical independence property (in terms of $\mu$-value) and "undershuffled" sequences, depending on the "margin of safety".

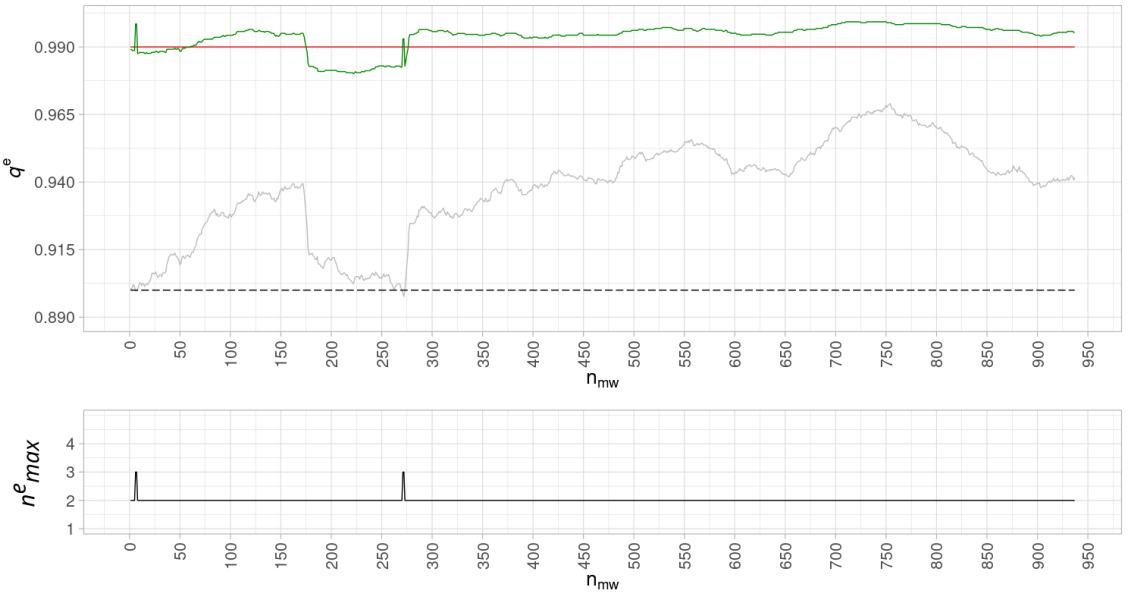
### 4.3.2 Evaluation using Real world link data

Previously, in Chapter 3, we performed statistical analysis on link data collected from the testbed under different traffic conditions. In this section, we perform link boosting on the link data and assess our metrics. In our analysis, we set the target reliability $r^e_{target}$ as 0.99. Initially, we discuss the results from link boosting and reactiveness of metrics in the experiments. Later, we discuss the results of a link in sensitive and non-sensitive regions.

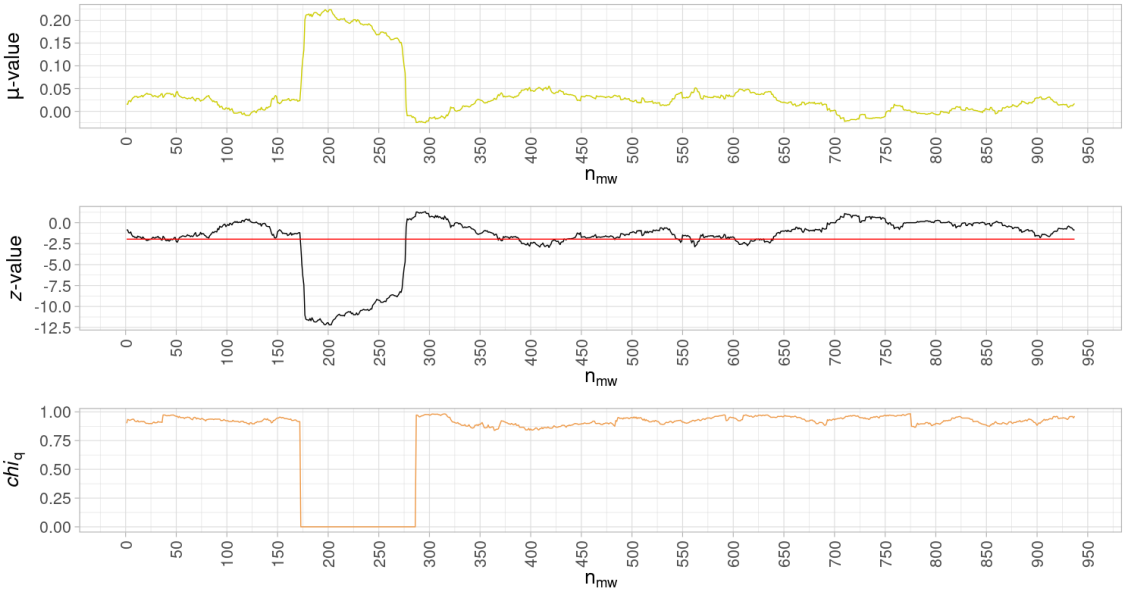
After analyzing the data collected from the testbed, the event outcomes on most of the links in the presence of external traffic possess the statistical properties to perform successful link boosting. We discuss the findings from our analysis. Fig. 4.14 shows the results of performing link boosting in link $4 \rightarrow 8$. Fig. 4.14a shows the results of link boosting along with the number of transmissions during each window. In the upper plot, the grey line denotes the raw PDR, and the red and green lines denote the Target reliability $r^e_{target}$ and the observed boosted reliability $pdr^e_{proc}$ (after link boosting), respectively, during each moving window. The lower plot denotes the number of transmissions $n^e_{max}$ during each window. Fig. 4.14b shows the values of each metric in each moving window. The target reliability $r^e_{target}$ is set to 0.99 (the red line Fig. 4.14a).

Between the moving windows 175 and 275, the link reliability dropped due to bursty frame losses. The event outcomes during those periods are bursty and non-random. The values of the metrics, $\mu$-value, $z$-value, and $chi_q$, reacts to the changes in the statistical properties of the event outcomes. In those windows, the $\mu$-value increases, denoting the statistically dependent or bursty event outcomes. Likewise, the $z$-value decreases, denoting the non-random "*undershuffled*" event outcome. And, the $chi_q$ returns a value of 0, denoting strong evidence to reject the existence of transmission event outcomes of type Bernoulli distribution with a success probability calculated from the raw PDR. In this scenario, all the metrics showed the signs of possible link boosting failure. However, in our analysis, there are cases where metrics, especially $chi_q$, return false negatives, i.e., windows where $chi_q$ equal 0, and link boosting was successful. As discussed in the previous section, where we

evaluated the relationship between $chi_q$ and the effectiveness of link boosting, these false negatives for $chi_q$ are rare.

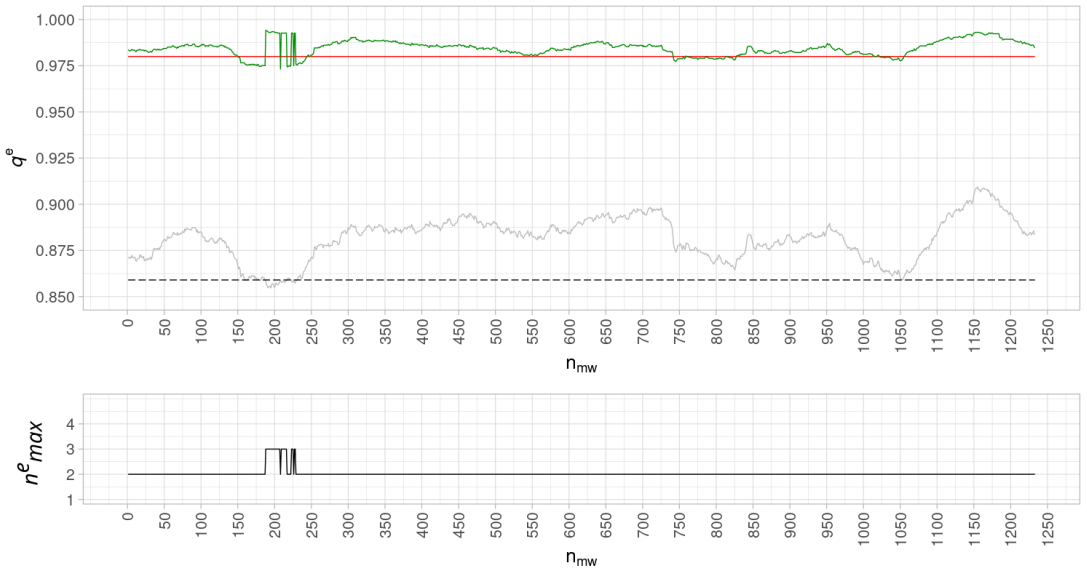



(a) The upper plot shows the raw PDR $pdr^e_{raw}$ (grey line), target reliability $r^e_{target}$ (red line), and observed boosted reliability $pdr^e_{proc}$ (green line) during each moving window. The lower plot shows the number of transmission required to meet the minimum target reliability $n^e_{max}$.

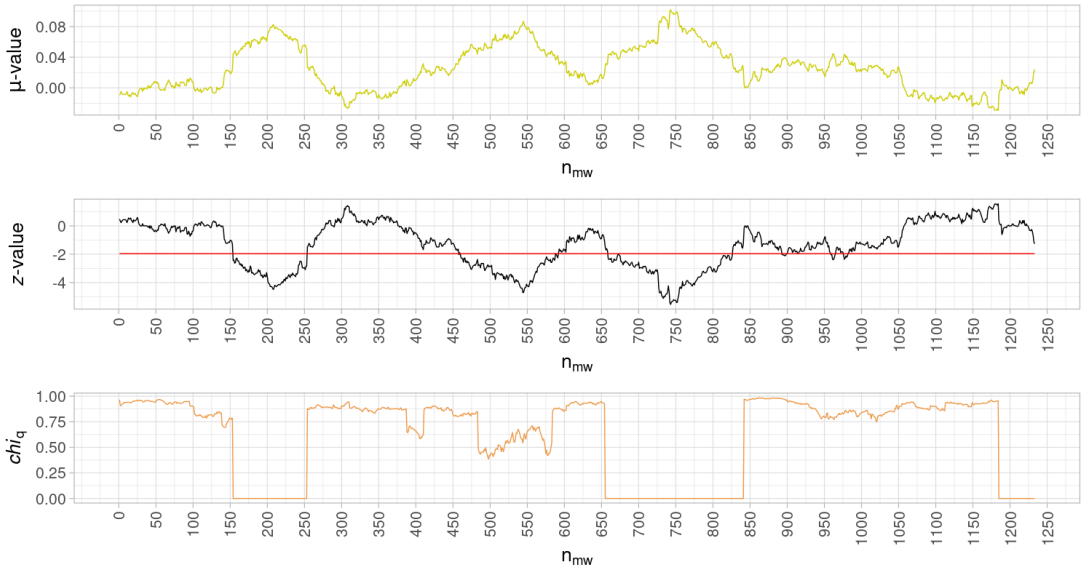



(b) Statistical test results during each moving window. The red line on the second plot ($z$-value during moving windows) is the rejection range, i.e., $1.96 \leq z \geq -1.96$.

Figure 4.14: Evaluating the effectiveness of link boosting and the response of metrics used to measure the statistical properties on the link $4 \rightarrow 8$. Evaluation uses a moving window configuration of $s_{Bs} = 30$ and $s_{mw} = 100$.




(a) The upper plot shows the raw PDR $pdr^e_{raw}$ (grey line), target reliability $r^e_{target}$ (red line), and observed boosted reliability $pdr^e_{proc}$ (green line) during each moving window. The dashed black line denotes the boundary where a drop in $q^e$ requires an extra transmission to meet the Target reliability $r^e_{target}$, i.e., a sensitive region. The lower plot shows the number of transmission required to meet the minimum target reliability $n^e_{max}$.




(b) Statistical test results during each moving window. The red line on the second plot ($z$-value during moving windows) is the rejection range, i.e., $1.96 \leq z \geq -1.96$.

Figure 4.15: Performing link boosting on the link $7 \rightarrow 10$. Evaluation uses a moving window configuration of $s_{Bs} = 30$ and $s_{mw} = 100$.

Fig. 4.14a, the lower plot plots the minimum number of transmissions $n^e_{max}$ required in a moving window to meet target reliability $r^e_{target}$; two spikes are spotted in the beginning and between moving windows 250 and 275. When we compare the

observed raw PDR $pdr^e_{raw}$ during those moving windows, the raw PDR dropped below 0.90. In those windows, at least three transmissions are required to meet target reliability. However, windows with observed raw PDR greater than 0.90 ($pdr^e_{raw} > 0.90$) require only two transmissions.

Next, we discuss the results in the sensitive and non-sensitive regions. Whether a link or window is in a sensitive region depends on the target reliability $r^e_{target}$ and the observed boosted reliability $pdr^e_{proc}$, i.e., $r^e_{target} - pdr^e_{proc}$, known as "margin of safety". A link is in the "*sensitive region*" if the link or moving window operates in a region where "margin of safety" is small (close to 0). Consider Fig. 4.15, which plots the results of link boosting on link $7 \rightarrow 10$. The target reliability is set to 0.98. It is represented with a red line in Fig. 4.15a. The line in grey and green denote the raw $pdr^e_{raw}$ and boosted $pdr^e_{proc}$ PDR, respectively. The dash blacked line denote the "sensitive region", where a window with raw PDR above 0.859 requires two transmissions ($n^e_{max}$), and below requires three. The $z$-value dropped below 1.96 (in small magnitude) between windows 150 and 275, 500 and 600, and 660 and 850, indicating non-random event outcomes (into the bursty region). Other metrics also reacted in those moving windows. The $\mu$-value increased (slightly), and the $chi_q$ indicating strong evidence against Bernoulli distribution with success probability equals to observed raw PDR.

In between windows 150 and 275, the observed boosted reliability $pdr^e_{proc}$ oscillates in the boundary between 2 and 3 transmissions to meet the minimum target reliability, i.e., the sensitive region. Between those windows where the observed raw PDR is above 0.859 (which implies two transmissions to meet the target), the link boosting failed by a very low margin ($\approx$0.006). Meanwhile, between windows 475 and 600, and 650 and 825, especially between the latter, the link boosting is successful for all windows. Even though the statistical properties of all those three periods are comparable, the raw PDR during the windows is not close to the borderline (i.e., to the sensitive region) in the last two periods (between windows 475 and 600, and 650 and 825). To conclude, links or windows with a very low "*margin of safety*", i.e., the difference between boosted link reliability and minimum target reliability, require a very strong statistical property compared to those with a high "*margin of safety*". For instance, Fig. 4.16 shows multiple links with statistical properties necessary for link boosting.

After evaluating multiple data sets from the testbed, the following are our findings.

- A large number of links in the testbed have the necessary statistical properties required for successful link boosting.

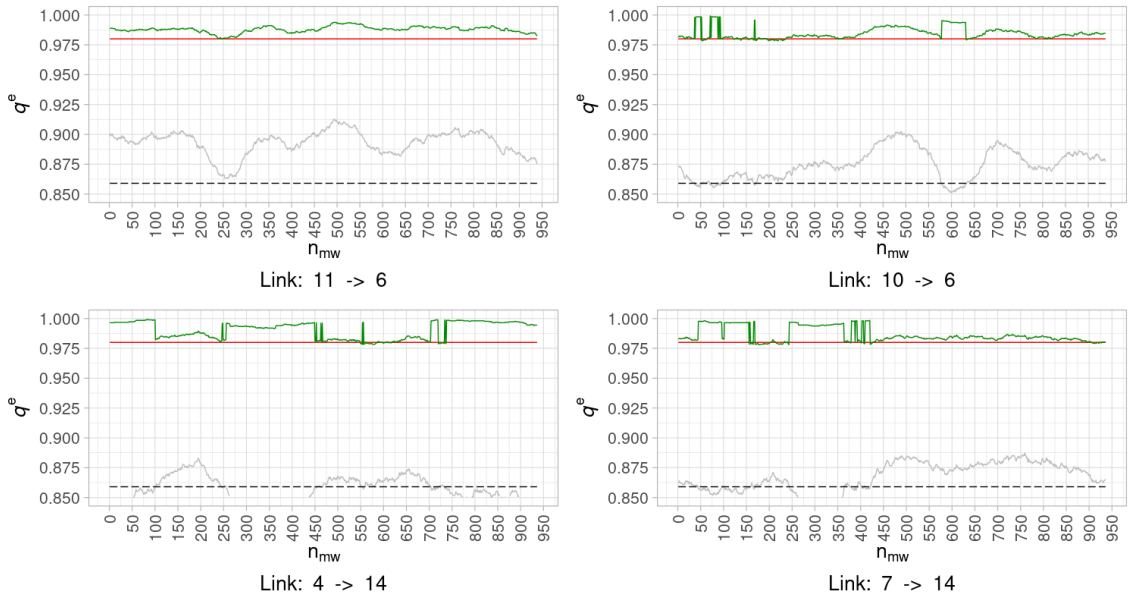



Figure 4.16: Performing link boosting on multiple links operating in a sensitive region

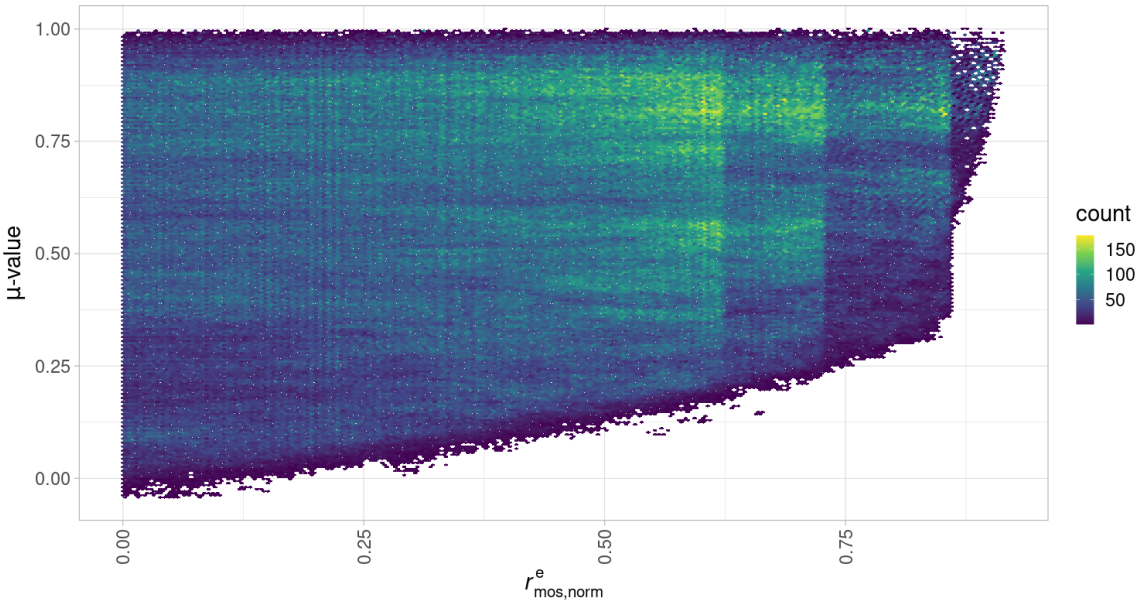
- To a great extent, using the metrics mentioned in this chapter, it is possible to measure certain statistical properties of transmission events.
- Links or moving windows with a low "margin of safety", i.e., the difference between boosted link reliability and minimum target reliability, require stronger statistical properties.

## 4.4 Measuring the "margin of safety" required to tolerate variation in statistical properties of a link

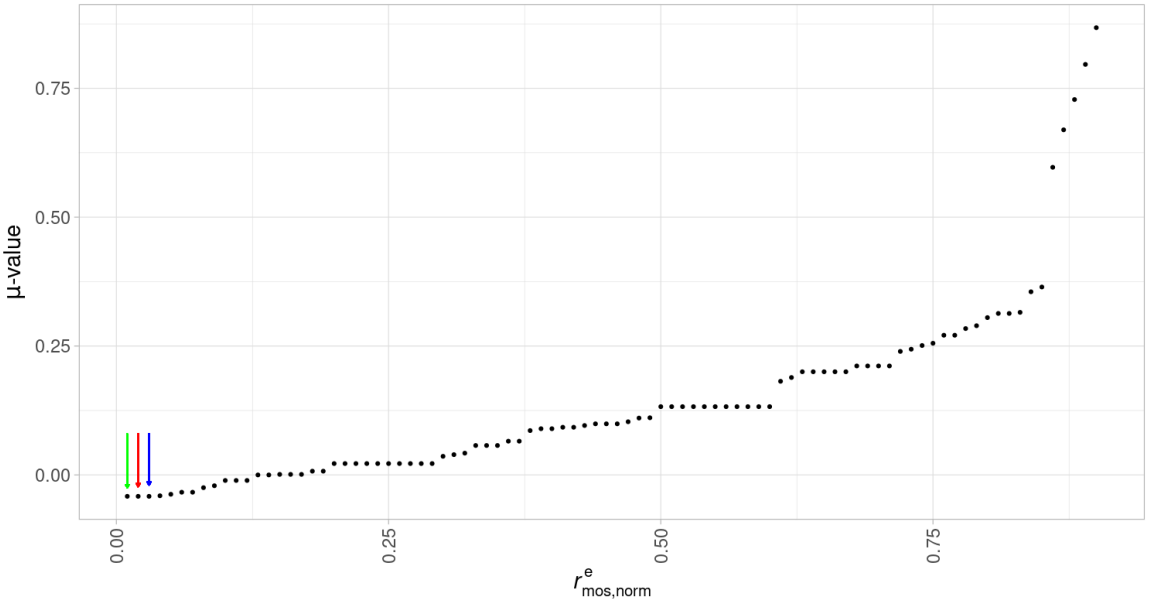
In the last section, we identified that link boosting was successful for certain moving windows or sequences with higher $\mu$-value depending on their "margin of safety" $r^e_{mos}$ (i.e., the tolerance in statistical independence property). Further, we identified the correlation between randomness and statistical independence. Therefore, we are only considering the $\mu$-value, not the $z$-value, since there is a strong correlation. In this section, we evaluate the minimum margin of safety required for a successful link boosting of sequences below a tolerable $\mu$-value. The *tolerable $\mu$-value* for a margin of safety $r^e_{mos}$ greater or equal to $n$ is the lowest $\mu$-value observed among the moving windows with the margin of safety $r^e_{mos}$ greater or equal to $n$ ($n \leq r^e_{mos}$), where the link boosting failed. To understand the variation in the tolerable $\mu$-value

depending on the link boosting requirements, we apply curve-fitting on the tolerable $\mu$-value depending on certain factors. The process of preparing the data for curve-fitting is later explained. We divide our evaluation into two parts. First, we evaluate the impact of the size of moving windows or sample sizes. Later, we evaluate the variation depending upon multiple target reliabilities.

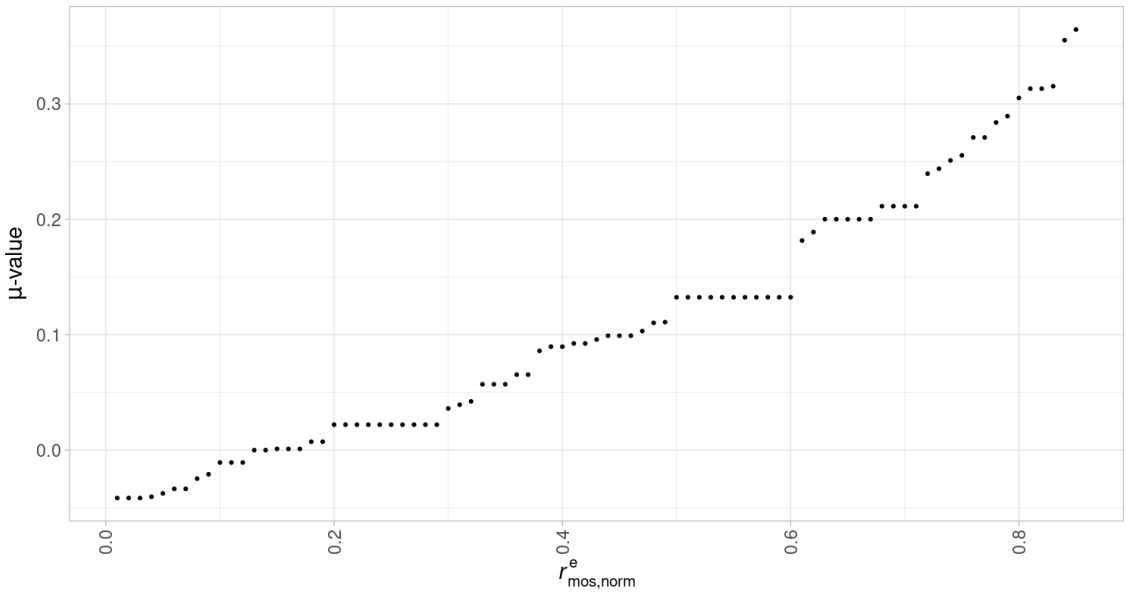
### 4.4.1 Foundations



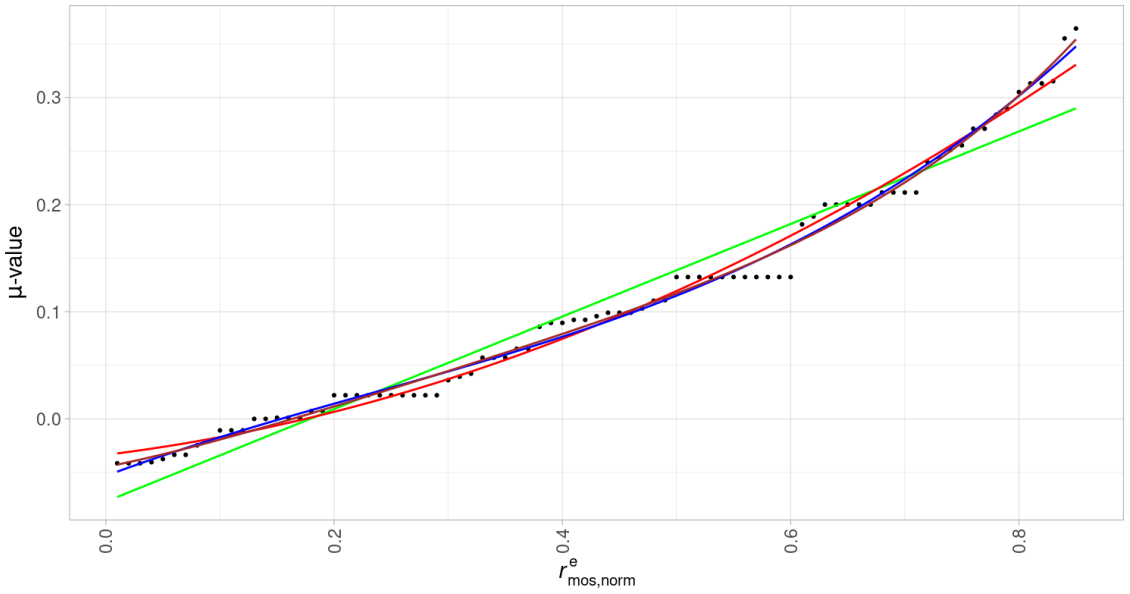

(a) Result after step 1




(b) Result after step 2




(c) Result after step 3




(d) Result after step 4

Figure 4.17: Plots showing the results of curve-fitting after each step.

We begin by explaining the steps to process the data for curve-fitting. Recall that, in Sec. 4.3.1.1, we discussed the relationship between Target missed $r^e_{\Delta,missed}$ depending on margin of safety $r^e_{mos}$ and $\mu$-value. The target reliability $r^e_{target}$ on a link changes depending on the route requirement. We normalize the margin of

safety for comparison depending on the maximum achievable margin of safety, i.e., normalized margin of safety

$$r^e_{mos,norm} = \frac{r^e_{mos}}{1 - r^e_{target}} \tag{4.4}$$

Fig. 4.17a shows the $\mu$-value and the normalized margin of safety $r^e_{mos,norm}$ during each moving window where link boosting failed, i.e., Target missed $r^e_{\Delta,missed} > 0$.

Next, we explain the steps to process the data for curve-fitting. Fig. 4.17 shows the results after each step. The figure shows the curve-fitting process for a moving window configuration $s_{Bs} = 30$ and $s_{mw} = 100$ (i.e., a sample size of 3000 events), and link boosting to a target reliability $r^e_{target}$ of 0.98.

***Step* 1:** Calculate the $\mu$-value and the margin of safety $r^e_{mos}$ using Eqn. 4.3.1.1 for each moving window where link boosting failed.

$$r^e_{mos}(n_{mw}) = q^e_{n^e_{max}}(n_{mw}) - r^e_{target} \qquad \text{(Eqn. 4.3.1.1 revisited)}$$

In this scenario, the upper bound for the margin of safety is 0.02, i.e., 1 - 0.98 (1 - $r^e_{target}$). From a margin of safety $r^e_{mos}$, calculate the normalized margin of safety $r^e_{mos,norm}$. Fig. 4.17a shows the results for the entire data set after step 1. It plots the $\mu$-value, and normalized margin of safety recorded during moving windows where link boosting failed.

***Step* 2:** Calculate the smallest observed $\mu$-value during the moving windows within a certain range of normalized margin of safety $r^e_{mos,norm}$, where link boosting failed. Let us use the Fig. 4.17b to explain step 2. In our evaluation, we used a range of 0.01. I.e., the first point (starting from the left and pointed out by the green arrow) in the plot denotes the smallest observed $\mu$-value among moving windows with a normalized margin of safety $r^e_{mos,norm}$ greater than or equal to 0. The second point (pointed out by the red arrow) denotes the smallest observed $\mu$-value among moving windows with a normalized margin of safety $r^e_{mos,norm}$ greater than or equal to 0.01. Similarly, the next point (pointed out by the blue arrow) for $\mu$-value among moving windows with a normalized margin of safety $r^e_{mos,norm}$ greater than or equal to 0.02. Continue this process for the subset of windows until the highest normalized margin of safety. Thus, we group the data into sub-groups based on their margin of safety. Fig. 4.17b shows the result after step 2.

***Step* 3:** Next, we reduce the data points by removing data points with a $\mu$-value higher than 0.5. Sequences with a $\mu$-value higher than 0.5 are bursty. Our focus is to analyze or study the tolerable $\mu$-value, i.e., tolerable burstiness or non-randomness. Fig. 4.17c shows the results after step 3.

***Step* 4:** Finally, we apply curve-fitting on the data points from the previous step. We apply multiple polynomial regression models, fitting curves with multiple degrees. Fig. 4.17d shows the results after curve-fitting. The green, red, blue, and brown coloured regression lines show the lines fitting models with degrees 1, 2, 3, and 4, respectively. Out of these models, we select the model with the highest "adjusted R-squared" value (of 0.99) [Mil14][14], which is with a degree of 4, for our future evaluation.

As mentioned earlier, the remaining study is divided into two parts,

- We study the influence of sample size, i.e., multiple moving window configurations, on tolerable $\mu$-value.
- Depending on the tolerable $\mu$-value, we calculate the minimum required margin of safety $r^e_{mos}$ for link boosting to multiple target reliabilities.

### 4.4.2 Influence of sample size

In this sub-section, we evaluate the influence of the sample size or the number of event outcomes in the moving window on the tolerable $\mu$-value. We apply link boosting to a target reliability $r^e_{target}$ of 0.98 for multiple window configurations on the synthesized data. The following window configurations are used in our evaluation,

*W.C.*1: $s_{Bs} = 30$ and $s_{mw} = 20$, resulting in 600 samples in a moving window,

*W.C.*2: $s_{Bs} = 30$ and $s_{mw} = 60$, resulting in 1800 samples in a moving window,

*W.C.*3: $s_{Bs} = 30$ and $s_{mw} = 100$, resulting in 3000 samples in a moving window, and

*W.C.*4: $s_{Bs} = 30$ and $s_{mw} = 140$, resulting in 4200 samples in a moving window.

[14]For a regression model, the R-squared values, ranging from 0 to 1, calculate the variance in the response and predictor variables. The adjusted R-square is the R-square value adjusted for the number of predictors in a regression model. It is used to measure the fitness of a model. The higher the R-squared value, the better the regression model fits to the data.

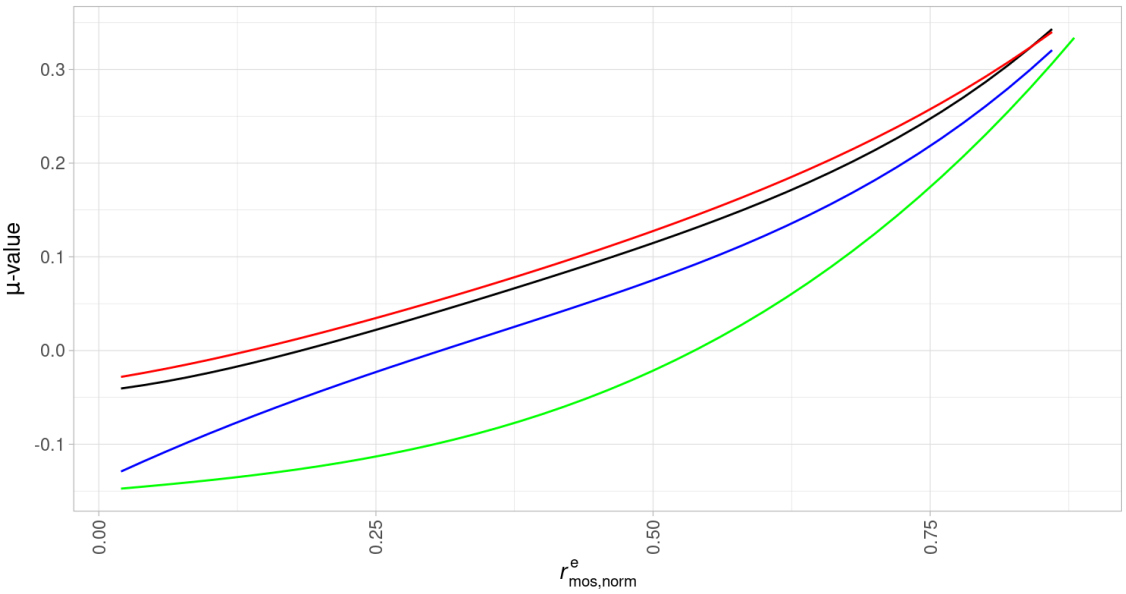



Figure 4.18: tolerable $\mu$-value while using multiple moving window configurations for a target reliability of $r^e_{target} = 0.98$. Among moving windows with failed link boosting, for a "normalized margin of safety" (the x-axis), the smallest $\mu$-value (the y-axis) observed during a window is plotted. The green, blue, black, and red lines denote the tolerable $\mu$-value for sample sizes of 600, 1800, 3000, and 4200 event outcomes.

Fig. 4.18 shows the results, the tolerable $\mu$-value using multiple moving window sizes. Each line in the plot denotes the curve-fitting on the smallest $\mu$-values observed for moving windows with normalized margin of safety $r^e_{mos,norm}$ where the link boosting failed. The green coloured line denotes the tolerable $\mu$-value using the window configuration *W.C.1* (sample size of 600), the blue coloured line using *W.C.2* (sample size of 1800), the black line using *W.C.3* (sample size of 3000), and the red coloured line using *W.C.4* (sample size of 4200).

When using the window configuration *W.C.1* (the green coloured line), the smallest sample size, the sequence requires a higher margin of safety to tolerate weakness in the statistical independence property. For instance, with a normalized margin of safety $r^e_{mos,norm}$ of 0.75, or margin of safety $r^e_{mos}$ of 0.015 (calculated using Eqn. 4.4), there exists a sequence with $\mu$-value 0.18 and failed link boosting to a target reliability $r^e_{target}$ of 0.98, where the boosted link reliability $q^e_{n^e_{max}}$ of the sequence is at least 0.995 ($r^e_{target} + r^e_{mos} = 0.98 + 0.015$). I.e., in other words, all sequences with normalized margin of safety $r^e_{mos,norm}$ and a $\mu$-value that falls below the curve had a successful link boosting to a target reliability $r^e_{target}$ of 0.98.

Smaller sized windows require stronger statistical properties even with a high margin of safety. For instance, there are windows with a $\mu$-value below 0 and link

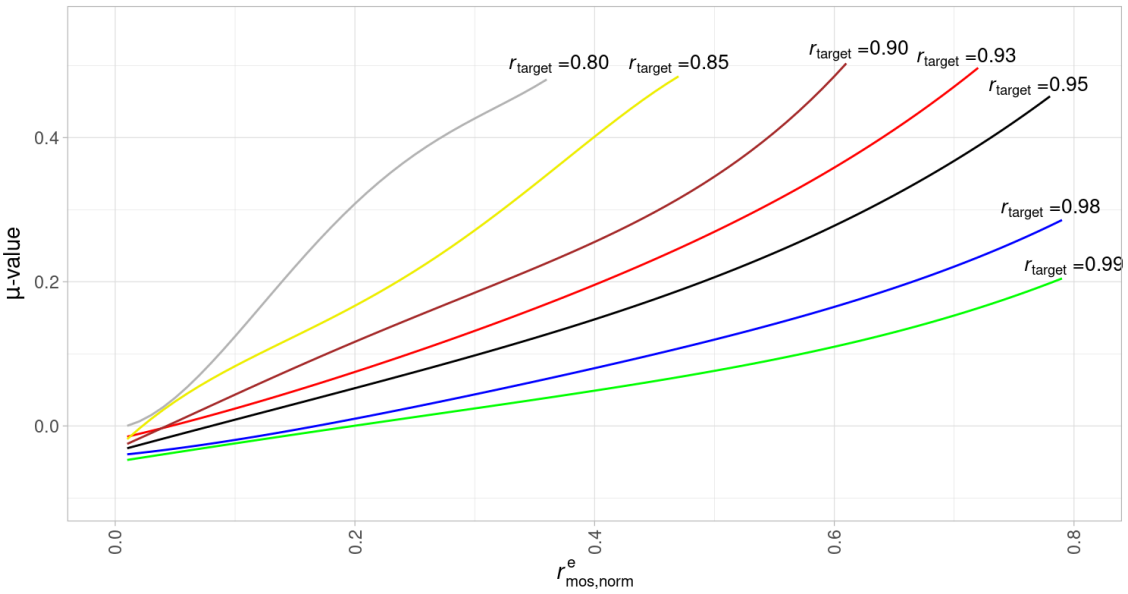
boosting failed where the normalized margin of safety is below 0.5 (see Fig. 4.18). Recall that the curve-fitting is applied to the smallest $\mu$-value observed among the windows above a certain margin of safety and failed link boosting. It does not mean that every sequence with the same $\mu$-value and margin of safety failed the link boosting. Moreover, it is less probable for a failed link boosting on those sequences (sequence with $\mu$-value of 0), and further, the Target missed $r^e_{\Delta,missed}$ for those sequences is below to 0.002.

As we increase the window size, the influence of window size or sample size, the tolerance ranges converge. For instance, consider the tolerable $\mu$-value using window configurations *W.C.3* and *W.C.4*, i.e., a sample size of 3000 (black coloured line) and 4200 (red coloured line) event outcomes. In practice, the link properties are not monitored for short sample sizes. Therefore, it is reasonable to use a window configuration or sample size containing 3000 event outcomes. Moreover, the tolerable $\mu$-value for a sample size of $l$ satisfies any sample size greater than $l$. For instance, consider the line for window configuration *W.C.2* (blue line) or sample size of 1800. All sequences of size 1800 with a $\mu$-value below 0.05 and a minimum normalized margin of safety of 0.50 had successful link boosting. It also applies to sequences with a size of 3000 and 4200. Therefore, based on our empirical analysis, a moving window configuration of at least 3000 event outcomes is considered (for instance, in the operation of $r_{min}$ routing).

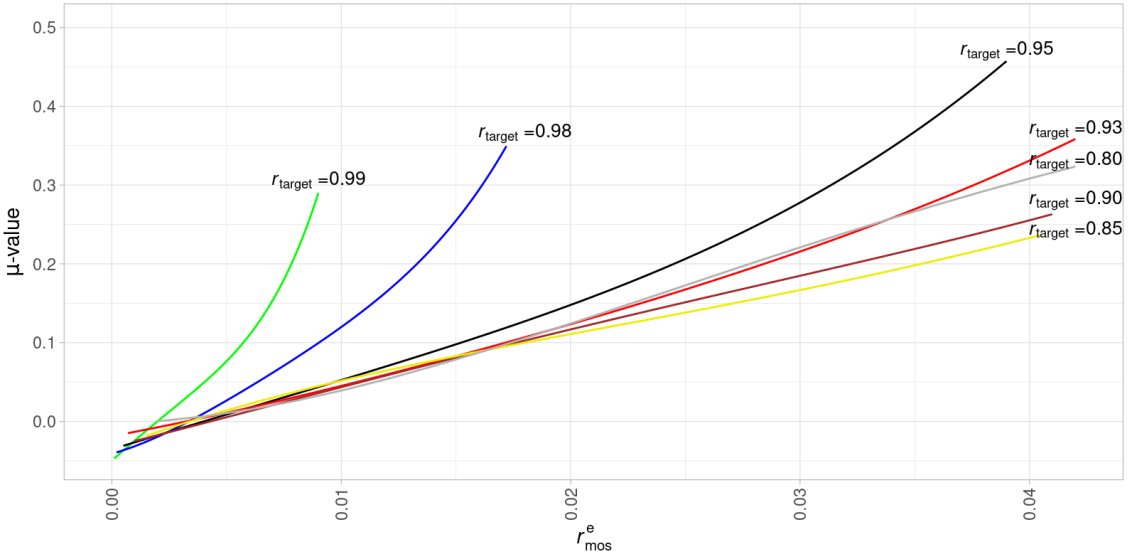
### 4.4.3 Influence of target reliabilities

In the previous section, we examined the influence of the sample size on the tolerable $\mu$-value. Next, we evaluate the influence of target reliabilities. I.e., we perform link boosting to multiple target reliabilities using the same moving window configuration. Our motivation is to find the margin of safety $r^e_{mos}$ required to tolerate a lack of statistical independence property in terms of $\mu$-value. In Sec. 4.3.2, the evaluation of link data collected from the testbed showed that the transmission event outcomes on most of the links have a $\mu$-value close to 0. Seldom, the $\mu$-values during the windows increase to a value of 0.1 (also decreases), i.e., minute variation from statistically independent outcomes. Therefore, we consider a $\mu$-value of 0.1 as the tolerable range and examine the margin of safety $r^e_{mos}$ required to tolerate a $\mu$-value up to 0.1.

A window configuration of $s_{Bs} = 30$ and $s_{mw} = 100$ (sample size of 3000) is used. Fig. 4.19 shows the tolerable $\mu$-value for multiple target reliabilities. Fig. 4.19a plots the tolerable $\mu$-value depending on the normalized margin of safety $r^e_{mos,norm}$ for successful link boosting to multiple target reliabilities (similar to Fig. 4.18). The area below the curves is the region of successful link boosting. For instance,




(a) Among moving windows with failed link boosting, for a normalized margin of safety $r^e_{mos,norm}$ (the x-axis), the window with the smallest $\mu$-value is plotted (the y-axis).




(b) Among moving windows with failed link boosting, for a margin of safety $r^e_{mos}$ (the x-axis), the window with the smallest $\mu$-value is plotted (the y-axis).

Figure 4.19: tolerable $\mu$-value while using multiple target reliabilities and same window configurations ($s_{Bs} = 30$ and $s_{mw} = 100$). The green, blue, black, red, brown, yellow, and grey lines denotes the tolerable $\mu$-value for target reliability $r^e_{target}$ 0.99, 0.98, 0.95, 0.93, 0.90, 0.85, and 0.80 event outcomes.

to boost the link reliability to a target reliability $r^e_{target}$ of 0.93 (the red line), all sequences with a $\mu$-value up to 0.2 and normalized margin of safety $r^e_{mos,norm}$ above 0.4 (or margin of safety $r^e_{mos}$ above 0.028 implies a minimum boosted link reliability

$q^e_{n^e_{max}}$ of 0.93 + 0.028, 0.958) had successful link boosting. Further, these criteria apply to link boosting below target reliability of 0.93. Moreover, what is the relation depending on the absolute value (margin of safety $r^e_{mos}$) rather than the normalized value (normalized margin of safety $r^e_{mos,norm}$)?

Fig. 4.19b shows the tolerable $\mu$-values depending on the margin of safety $r^e_{mos}$. For instance, in Fig. 4.19b, the green line denotes tolerable $\mu$-value depending on the margin of safety for successful link boosting to a target reliability $r^e_{target}$ of 0.99. The upper bound for margin of safety in this case is 0.01, i.e., $1 - 0.99$. Similarly, the blue line for successful link boosting to a target reliability $r^e_{target}$ of 0.98, and the maximum margin of safety $r^e_{mos}$ for this curve is 0.02 $(1 - 0.98)$.

Let us evaluate the minimum required margin of safety to tolerate a $\mu$-value up to 0.1. For instance, Fig. 4.19 shows that while link boosting to a minimum target reliability $r^e_{target}$ of 0.98, a sequence with a $\mu$-value 0.1 will have successful link boosting if we assign to a target reliability plus a normalized margin of safety $r^e_{mos,norm}$ of 0.41 or margin of safety $r^e_{mos}$ of 0.008, i.e., 0.988 ($r^e_{target} + r^e_{mos} = 0.98 + 0.008$). Tab. 4.5 tables the minimum required normalized margin of safety $r^e_{mos,norm}$ and margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1, i.e., the x-values of the plots (Fig. 4.19a and Fig. 4.19b) for a y-value equal to 0.1. Notably, below a target reliability $r^e_{target}$ of 0.90, the required margin of safety $r^e_{mos}$ converges between 0.016 and 0.018. It is visualized in Fig. 4.19b.

Recalling the motivation of this sub-section, what is the minimum required margin of safety to tolerate a lack of statistical independence property in terms of $\mu$-value? Based on our examination in Sec. 4.3.2, we considered a $\mu$-value of 0.1 as the tolerable range. We repeated the calculation of the required margin of safety to tolerate a $\mu$-value up to 0.1 for further target reliabilities. Fig. 4.20 shows the minimum margin of safety $r^e_{mos}$ required to tolerate a $\mu$-value up to 0.1 depending on multiple target reliabilities. As mentioned earlier, the required margin of safety $r^e_{mos}$

| $r^e_{target}$ | Curve colour | $\mu$-value | $r^e_{mos,norm}$ | $r^e_{mos}$ |
|---|---|---|---|---|
| 0.99 | green | 0.1 | 0.51 | 0.005 |
| 0.98 | blue | 0.1 | 0.41 | 0.008 |
| 0.95 | black | 0.1 | 0.29 | 0.015 |
| 0.93 | red | 0.1 | 0.23 | 0.016 |
| 0.90 | brown | 0.1 | 0.18 | 0.018 |
| 0.85 | yellow | 0.1 | 0.13 | 0.018 |
| 0.80 | grey | 0.1 | 0.08 | 0.016 |

Table 4.5: Required margin of safety to tolerate up to a $\mu$-value of 0.01.

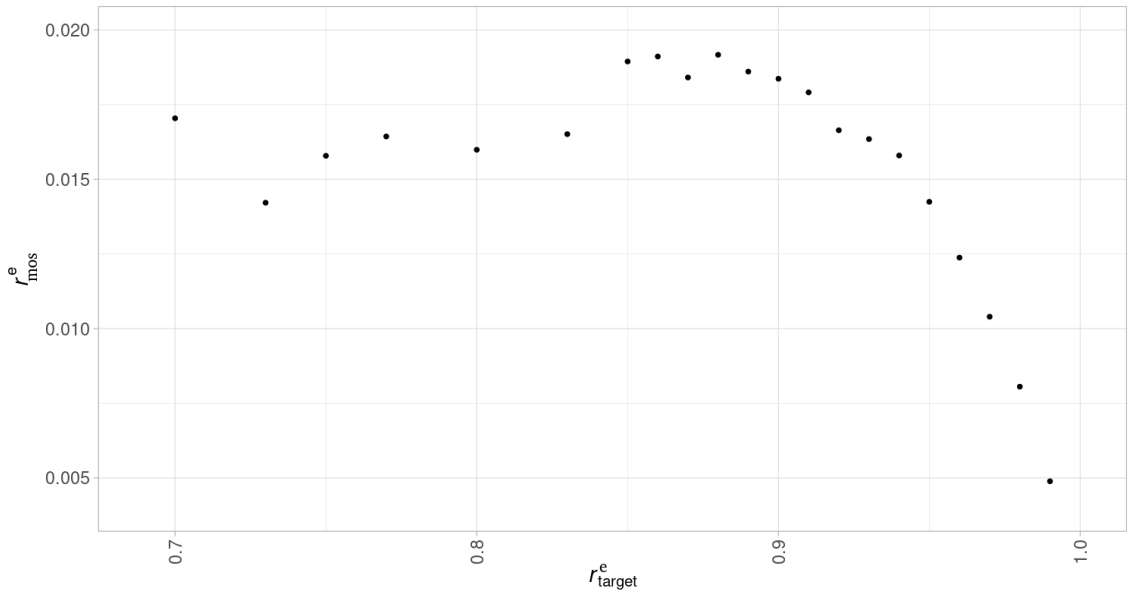



Figure 4.20: The minimum required margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1 depending on multiple target reliabilities.

converges between 0.016 and 0.018 for target reliabilities below 0.85. Therefore, a margin of safety $r^e_{mos}$ of 0.018 is recommendable for link boosting below a target reliability $r^e_{target}$ of 0.85. For instance, to boost the link reliability to a target reliability $r^e_{target}$ of 0.80, it is more reliable to boost to a reliability of 0.818 so that the link can tolerate up to a $\mu$-value of 0.1.

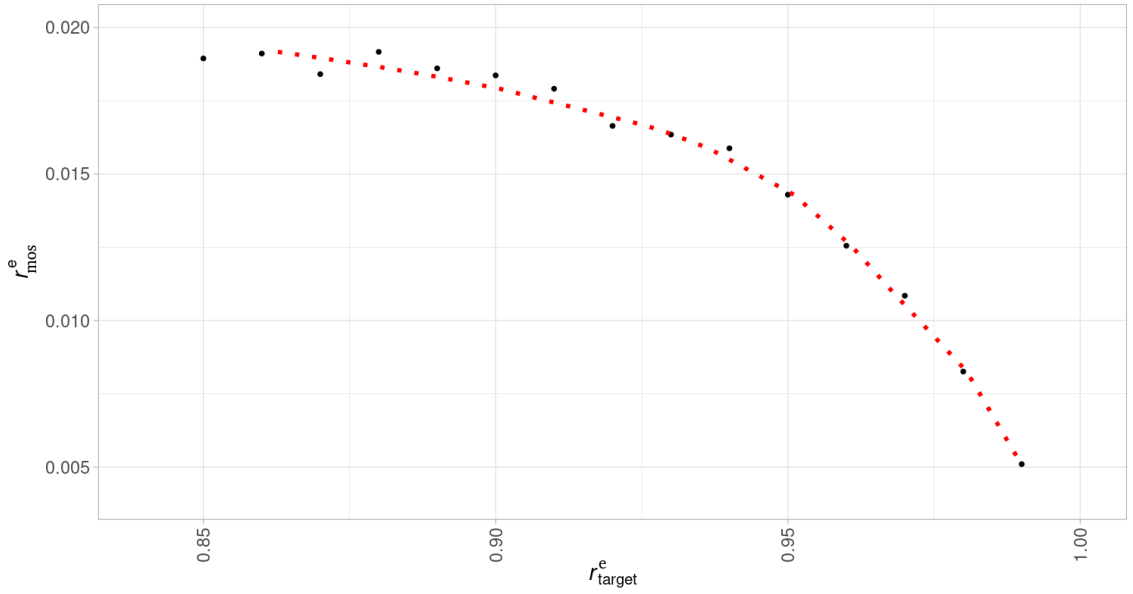



Figure 4.21: The minimum required margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1 depending on target reliabilities between 0.85 and 0.99. The red dotted line is the regression line.

For link boosting above 0.85, a dynamic margin of safety depending on the target reliability is required. Fig. 4.21 shows the required margin of safety $r^e_{mos}$ to tolerate a $\mu$-value up to 0.1 for target reliabilities from 0.85 to 0.99. The dotted red line shows the curve fitting on a polynomial regression model with a degree of 3. For boosting above 0.85, we use the curve's equation to calculate the required margin of safety. Hence, the minimum required margin of safety $r^e_{mosRequired,0.1}(r^e_{target})$ to tolerate up to a $\mu$-value of 0.1 for a target reliability $r^e_{target}$ is calculated as,

$$r^e_{mosRequired,0.1}(x) = \begin{cases} (-6.818944 \cdot x^3) + (17.77401 \cdot x^2) + & , \text{ if } x \geq 0.85 \\ (-15.45451 \cdot x) + 4.501372 & \\ 0.18 & , \text{ if } x < 0.85 \end{cases} \tag{4.5}$$

where $x = r^e_{target}$.

Concluding our findings, we started by evaluating the influence of sample sizes while calculating the margin of safety $r^e_{mos}$ required to tolerate variations in statistical independence property. We found that sample sizes do have an impact on our analysis. A sample size of 3000 is recommended as a minimum. Notably, a tolerable $\mu$-value that satisfies a sample size of 3000 satisfies any sample size greater than 3000. Following, we evaluated the minimum required margin of safety $r^e_{mosRequired,0.1}(r^e_{target})$ depending on the target reliability $r^e_{target}$ to tolerate a $\mu$-value up to 0.1.

## 4.5 Conclusion

Recalling the question stated at the beginning of this chapter, is it necessary that the outcomes of the transmission events must be statistically independent, random, and fit a Bernoulli distribution with a specific success probability for successful link boosting? To answer, we conducted link boosting (offline) on synthesized and real data collected from the testbed, and used the metrics ($\mu$-value, $z$-value, and $chi_q$) defined in the Chapter 3 to find the necessity of certain statistical properties.

To begin with, in Sec. 4.2, we explained the link boosting process on a link. Then, we introduced the metric Target missed $r^e_{\Delta,missed}$, to measure the effectiveness of link boosting depending on certain statistical properties. Following that, in Sec. 4.3.1, we looked at the applicability of metrics (used to assess different statistical properties) such as $\mu$-value, $z$-value, and $chi_q$, to find a possible relationship between the necessity of certain statistical properties for successful link boosting. Experimental results show that as transmission event outcomes become statistically dependent or bursty (non-random), the Target missed $r^e_{\Delta,missed}$ increases, implying failed link

boosting with a higher margin. Further, for sub-sequences or windows where the event outcomes are statistically independent or random and notably oscillatory, the link boosting was successful. Moreover, we observed a strong correlation between statistically independent and random sub-sequences in this context, i.e., between $\mu$-value and $z$-value. Pointing out, we observed that some moving windows with sub-sequences having weaker statistical independence property, i.e., higher $\mu$-values, had successful link boosting. Investigating those cases, we noticed that link boosting in those moving windows had a high margin of safety $r^e_{mos}$, i.e., the difference between boosted link reliability $q^e_{n^e_{max}}$ and minimum target reliability $r^e_{target}$. Findings from this analysis lead to further study on the influence of margin of safety $r^e_{mos}(n_{mw})$.

Before that, in Sec. 4.3.2, we performed offline link boosting on the data collected from the testbed under different traffic conditions. We further assessed our metrics (used to assess different statistical properties) and evaluated the reactiveness of metrics to changes in the event outcomes. Our evaluation results showed that a large number of links in the testbed have the necessary statistical properties required for successful link boosting. Moreover, to a great extent, the metrics are useful to measure the statistical properties of transmission events.

Finally, in Sec. 4.4, we assessed the minimum required margin of safety depending on the target reliability to tolerate up to a specific $\mu$-value. We considered a $\mu$-value of 0.1 tolerable depending on our findings from evaluating link data collected from the testbed. I.e., seldom, the $\mu$-values of links (which showed strong statistical independence property) increase up to 0.1. Based on our findings, we derived the minimum margin of safety $r^e_{mosRequired,0.1}(r^e_{target})$ to tolerate up to a $\mu$-value of 0.1 while boosting to a target reliability $r^e_{target}$.

As mentioned, a $\mu$-value of 0.1 was considered (as tolerable) based on the findings from evaluating data collected from the testbed. The testbed operates in the presence of external traffic, and a sufficient amount of competing nodes were present to create an independent *conditional collision probability* [Bia00] or random collision events. Is this applicable to the sensor networks where the applications running on top create periodic and aperiodic traffic? What if the traffic generated by the application is periodic? Is the random back-off channel access method of 802.11 necessary to create random collision events or independent *conditional collision probability*? We look further into this direction in the coming parts of the thesis.

# 5

# Time Token Bucket Algorithm

## Contents


**5.1 Introduction** . . . . . **122**

**5.2 Medium Management Layer** . . . . . **123**

**5.3 Time Token Bucket Algorithm (TTBA)** . . . . . **125**

5.3.1 State of the art . . . . . 126

5.3.2 Operation of the TTBA . . . . . 129

5.3.3 Dynamic token refilling to introduce randomness . . . . 138

**5.4 Experiments** . . . . . **141**

5.4.1 Evaluating the traffic monitoring metric, *unusable wasted assigned medium occupancy ratio* . . . . . 141

5.4.2 Evaluating the *dynamic refilling* mechanism . . . . . 150

**5.5 Algorithms and protocols using the functionalities of the MML** . . . . . **163**

5.5.1 TTBA for Cooperative Fair Medium Sharing . . . . . 163

5.5.2 Route selection based on the medium conditions . . . . 164

**5.6 Conclusion** . . . . . **164**

## 5.1 Introduction

Link reliabilities are significantly influenced by the traffic load in the channel and traffic patterns (if any) caused by the applications running on the nodes. For instance, an overloaded medium may increase the frame collision probability of a network and thus, affects the link reliabilities. Further, traffic patterns unintentionally created by applications running on the nodes may affect the statistical properties of a link. Thus, it is vital to monitor the medium load and shape the traffic generated by the nodes. In this chapter, we present a hardware-independent approach for traffic shaping that reduces the influence of applications on statistical link properties and a metric to monitor medium overload locally. The proposed mechanisms are utilized by the Medium Management Layer (MML) of the WiPS framework (see Fig. 5.1).

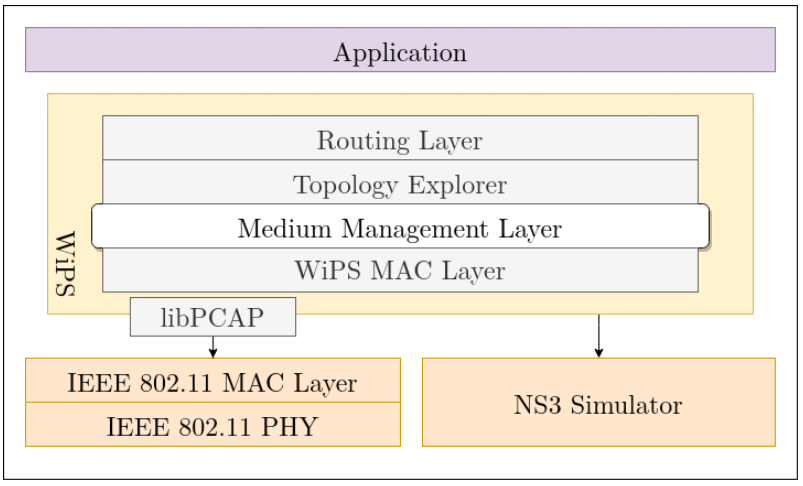



Figure 5.1: Medium Management Layer in the WiPS

This chapter is structured as follows. Sec. 5.2 discusses the MML of the WiPS, its modules, and its functionalities. In Sec. 5.3, the Time Token Bucket Algorithm (TTBA) for traffic shaping and monitoring is presented. This section also introduces the revised version of TTBA using *dynamic refilling*. Sec. 5.4 discusses the results of the experiments conducted to evaluate the applicability of our medium monitoring metric and compare the improvements using the revised version of TTBA over the original version. Then in Sec. 5.5, we briefly discuss how other layers of the WiPS use the MML. Finally, in Sec. 5.6, we summarize the chapter.

## 5.2 Medium Management Layer

The Medium Management Layer (MML) of the WiPS framework manages the medium occupancy time of a node. It calculates the required ratio of medium occupancy time by a node based on its requirements. Using the Time Token Bucket Algorithm (TTBA)[MKG17], the MML controls and monitors the flow of the frames and medium overload.

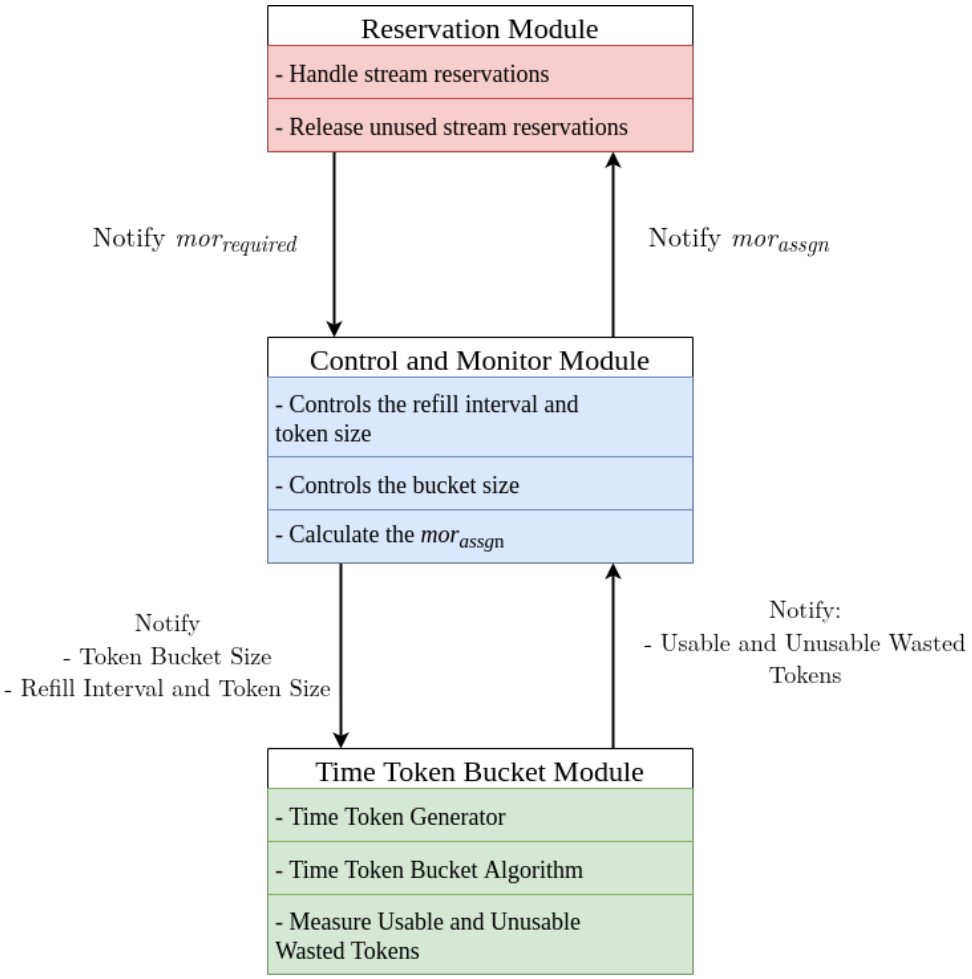



Figure 5.2: Medium Management Layer modules

To monitor the requirements and allocate the required ratio of medium occupancy time, the MML handles the data from each application (or from a stack) as a stream. Each stream has a stream identifier $s_{id}$ and priority $s_{prio}$. The frames from the higher layers arrive into the MML-arrival queue of the MML. Then, frames are processed

based on the corresponding stream priority. If there exist two or more frames with the same priority $s_{prio}$ (or no assigned priority), the frames are processed based on the arrival order (the FIFO rule applies). Next, we introduce and discuss the functionalities of different modules of the MML.

The Medium Management Layer (MML) modules and their interactions are depicted in Fig. 5.2. The MML consists of three modules:

1. Reservation Module,
2. Control and Monitor Module, and
3. Time Token Bucket Module.

The Reservation Module handles the medium reservations by each application and other protocol stacks. In this chapter, we use the terminology "medium occupancy ratio" to denote the transmission time ratio. Reservation decisions depend on the medium occupancy ratio assigned to node $v$, denoted as $mor^v_{assgn} \in [0, 1]$. It is calculated based on the requirements of the node and the available medium occupancy ratio $mor^v_{avail} \in [0, 1]$. The Control and Monitor Module handles the calculation of the assigned medium occupancy ratio $mor^v_{assgn}$. For instance, a $mor^v_{assgn} = 0.1$ configures 10% of the medium utilization time to node $v$.

When node $v$ starts, an assigned medium occupancy ratio $mor^v_{assgn}$ is configured by estimating its requirements $mor^v_{required} \in [0, 1]$. When the node requires to reserve additional data streams at run-time, the Reservation Module accepts or rejects the reservation depending on the assigned medium occupancy ratio $mor^v_{assgn}$. If the assigned medium occupancy ratio $mor^v_{assgn}$ is insufficient to support the new stream, i.e., $mor^v_{required} > mor^v_{assgn}$, the Reservation Module notifies the Control and Monitor Module. Then, the Control and Monitor Module processes the request and feedbacks the updated assigned medium occupancy ratio $mor^v_{assgn}$ to the Reservation Module.

Further, the Reservation Module monitors each stream and releases unused stream reservations. Depending on the settings of the MML, it releases streams in case of medium overload, i.e., when the assigned medium occupancy ratio $mor^v_{assgn}$ is updated, and $mor^v_{assgn} < mor^v_{required}$. For instance, if the medium becomes overloaded, the Control and Monitor Module recalculates the medium occupancy ratio $mor^v_{assgn}$ assigned to node $v$ and notifies the Reservation Module. The Reservation Module releases the stream based on stream priorities, $s_{prio}$, where the streams with the lowest priorities are released. An alternative is that the MML notifies or indicates the medium congestion ($mor^v_{assgn} < mor^v_{required}$) to service a stream, and then passes the responsibility (for instance, to release the stream or reduce the traffic) to the layer that created the stream.

As mentioned earlier, when node $v$ starts, a medium occupancy ratio $mor^{v}_{assgn}$ is assigned to the node. Then, the Control and Monitor module calculates the vital parameters for the operation of the Time Token Bucket Module depending on the assigned medium occupancy ratio $mor^{v}_{assgn}$. Further, it recalculates $mor^{v}_{assgn}$ depending on the requirements of the node at run-time. A medium occupancy ratio of $mor^{G}_{assgn} \in [0, 1]$ is reserved for the internal nodes [15] of the network $G$. For instance, in an environment where the network $G$ shares the channel with multiple foreign networks, to achieve a high reliability, a lower and conservative $mor^{G}_{assgn} = 0.2$ is propitious, i.e., limited by 20%. Contrary, if few or no foreign networks operate in the same channel, a higher value is possible, for instance, $mor^{G}_{assgn} = 0.7$, i.e., 70% of the transmission time for nodes in the network $G$. In [KMG19b], a method to dynamically determine the medium occupancy ratio $mor^{G}_{avail} \in [0, 1]$ available to the network $G$ is presented. Let us assume that the available medium occupancy ratio $mor^{G}_{avail}$ is configured to a suitable value. Then, while the Control and Monitor Modules recalculates the medium occupancy ratio $mor^{v}_{assgn}$ assigned to node $v$, it ensures that the sum of the assigned medium occupancy ratio to all internal nodes $v \in V^{int}$ is less than the assigned medium occupancy ratio to the network $G$, i.e., the following condition is satisfied,

$$\sum_{\forall v \in V^{int}} mor^{v}_{assgn} = mor^{G}_{assgn} \leq mor^{G}_{avail} \tag{5.1}$$

Furthermore, in [KMG19b], the medium occupancy ratio $mor^{G}_{assgn}$ assigned to the network $G$ is calculated by continuously monitoring the metrics derived from *usableWastedTokens* and *unusableWastedTokens* calculated by the Time Token Bucket Module. We discuss further while introducing the Time Token Bucket Algorithm (TTBA).

## 5.3 Time Token Bucket Algorithm (TTBA)

This section discusses the Time Token Bucket Algorithm (TTBA) used by the Time Token Bucket Module to control and monitor the traffic flow in nodes. The concept of the TTBA is discussed in [MKG17]. We begin by discussing the state of the art researches on traffic shaping and monitoring. Then, we discuss the operation of TTBA. Compared to the original version of TTBA, we introduce a revised token refilling mechanism to improve the link reliability and statistical link properties.

[15] Internal nodes are the nodes that use the Time Token Bucket algorithm whereas the external nodes belong to the foreign networks

In the revised version, instead of a constant refill interval, we use a dynamic refill interval.

## 5.3.1 State of the art

Traffic control in communication networks is performed by traffic monitoring and traffic shaping. In [Kra21], the author investigates the state of the art algorithms and development of the Time Token Bucket. We will revisit the research and discuss the important findings. Then, we will discuss the methods used to monitor medium traffic and overload.

### 5.3.1.1 Traffic shaping algorithms

Traffic shaping is the process of controlling the outgoing traffic from a node into the shared communication channel. It is used to achieve multiple objectives. For instance, to control medium overload, to share the medium depending on the requirements, or to control bursty transmission events. Depending on the objectives of the network and host computers, a traffic shaping algorithm forms the outgoing traffic from a node. The leaky bucket and token bucket algorithm [Tan03] are the early algorithms used for traffic shaping. The Time Token Bucket algorithm is a variant that developed from the Token Bucket algorithm for IEEE 802.11 network. Next, we discuss the traffic shaping algorithms such as the leaky bucket, token bucket, and time token bucket algorithms.

#### Leaky Bucket Algorithm

The leaky bucket Algorithm is used to limit or shape the flow of packets into the communication channel in packet-switched computer networks and telecommunication networks. In [Tur86], the author first proposed the leaky bucket algorithm. It uses the analogy of a fixed-size bucket holding water where the water leaks through a hole at the bottom. A fixed amount of water is added to the bucket periodically,, and in the meantime, a fixed volume of water leaks through the hole. If the water is poured at a rate higher than the water leaks through the hole, or pour water more than the remaining bucket capacity at once, the bucket overflows. Furthermore, if the bucket is empty, no water is leaking. This analogy is applied to shape the flow of packets from a node.

In the leaky bucket algorithm, the bucket holds transmission units of fixed size $k$ with a maximum bucket capacity of $s_b$ [Bytes]. The transmission units are leaked or transmitted at a constant time interval $\Delta t$ [s], thus, resulting in a transmission

rate of $k/\Delta t$ [Bytes/s]. Transmission units arrive at a constant or variable rate into the leaky bucket. If the bucket is full, then the bucket overflows, i.e. transmission units are discarded.

In practice, the transmission units in the bucket are represented using a counter $c_b$, where $0 \leq c_b \leq s_b$, which counts the number of pending packets of constant size $k$. When a new transmission unit arrives, it checks whether the counter is less than the bucket size $s_b$. If the counter value is higher than the bucket size, the transmission unit is discarded. Meanwhile, one transmission unit is transmitted at constant interval $\Delta t$ [s]. Further, when the counter is zero, i.e. the bucket is empty, no transmission units are sent.

#### Token Bucket Algorithm

The token bucket algorithm [Tan03] is developed to overcome the limitations of the leaky bucket algorithm. The leaky bucket algorithm limits the bandwidth usage. The Token Bucket algorithm permits bursts, restricts the burst size to the size of the bucket, and ensures that the host does not exceed the average permitted bandwidth. As discussed in [Kra21], in some literatures, authors use the terminology "Leaky Bucket" to explain the token bucket algorithm.

The token bucket algorithm uses the analogy of a fixed-sized bucket that holds up to $s_b$ tokens, i.e., the size of the bucket. The tokens are refilled at a constant interval $\Delta t$. A token represents the permission to send a packet. The packets from the host computer arrive in the queue. When a packet arrives in an empty queue, it waits if the bucket is empty. Else, it consumes a token, and it is sent. If multiple packets arrive in the queue, they are processed depending on the number of tokens in the bucket, i.e., a token per packet is consumed. For instance, if $x$ packets arrive in an empty queue at a time, and if there are $y$ tokens (where $y \leq s_b$) in the bucket, such that there are not enough tokens to service the packets, i.e., $x > y$, then $y$ packets are consecutively sent. The remaining $x - y$ packets wait until tokens are refilled. Thus, the maximum number of packets configured to send consecutively at a time is limited to $s_b$, i.e., the burst size. Moreover, on average, a transmission rate of $1/\Delta t$ [Packets/s] is achieved.

Another variant, known as the byte-count variant [Tan03], considers a token as bytes of size $k$ [Bytes] permitted to send instead of a packet. For example, a packet of size $s_p$ [Bytes] needs to consume at least a sum of tokens greater than or equal to $s_p$ [Bytes]. Thus, on average, a transmission rate of $k/\Delta t$ [Bytes/s] is achieved. Thus, the token bucket algorithm permits short bursts and limits (and allow) the

host computer to have an average (permitted) bandwidth by configuring the token size and refill interval.

#### Time Token Bucket

Time Token Bucket is developed for networks that support multiple transmission rates. For instance, IEEE 802.11 supports transmission in multiple transmission rates, ranging from 1 Mbps to 1.2 Gbps (with IEEE 802.11ax). As the technology evolves, faster and faster transmission rates are supported. Thus, frames with different transmission rates compete for the medium. A token bucket that controls packet transmission in bytes per second results in an unfair medium distribution among nodes. I.e., a node transmitting at low transmission rate occupies the medium for a higher amount of time than a node transmitting at a higher transmission rate. For these networks, it is reasonable to limit the amount of transmission time assigned to a node. We call this approach as Time Token Bucket algorithm. In [TG04], the authors described this approach under the name Time-based Regulator. The Time Token Bucket approach is similar to the conventional token bucket algorithm. Instead of bytes, the bucket represents the amount of transmission time (including other delays such as transmitting headers, inter-frame spacing, medium access delay) available to the node. The tokens are transmission time units, and together with refill interval, dictates the ratio of transmission time allocated to the node. A frame that arrives in the queue consumes the corresponding amount of transmission time required for transmitting. Thus, irrespective of their transmission rates, we can control the amount of medium occupancy by a node.

### 5.3.1.2 Traffic monitoring

Traffic monitoring is a vital function for traffic control in wireless networks. We discuss the state of the art traffic monitoring mechanisms for IEEE 802.11 networks. IEEE 802.11 nodes operate in ISM bands (2.4 or 5 GHz) where the medium is shared with external nodes, and monitoring the medium helps to estimate the available medium utilization time to a node. There are dedicated hardwares for medium monitoring of ISM bands, but we focus on available methods using off-the-shelf (commodity) WiFi adapters.

#### Channel Busy Time (CBT)

Channel Busy Time (CBT) is a hardware-dependent metric derived from the hardware register of a WiFi adapter. The CBT calculates the amount of time hardware detects energy above the noise threshold. The calculation method varies

from hardware to hardware. For instance, in TP-Link TL-WN772N v1.10 adapters with Atheros AR9002U chips using ath9k driver, the register holds the CBT from the beginning the node turned on. While in Logilink WL0084B adapter with Ralink chips using rt2800 driver, the register holds value from the last polling. I.e., in Ralink chips using rt2800 driver, the register value resets on each polling. Further, the noise threshold value for adapters varies. Therefore, hardware knowledge is necessary for metrics based on the CBT.

Earlier, access to the CBT register values was restricted and required modification to the driver [Ach+08; SAB08; Jar+05]. Now, several WiFi adapters provide access to the register values. Channel Busy Fraction (CBF) [DKS10] is a metric developed using the CBT. The CBF returns the fraction of time the medium was detected busy, i.e., CBT divided by the total sensing time. The CBF does not consider channel access delay and inter-frame spacing. Thus, a CBF value above 80% denotes a medium with close to 100% of the medium utilized.

#### Medium Utilization Time (MUT)

Another approach is to monitor the traffic in the medium and calculate the transmission time required for transmitting each frame. By listening to the medium (overhearing), a node can decode the header details of a captured IEEE 802.11 frame, which provides the necessary information to calculate the transmission time. Delays due to inter-frame spacing and medium access overhead are estimated. Aggregating these details from the overheard frames is used to calculate the fraction of the time medium occupied by 802.11 frames, and in [Ach+08], the authors presented it as the Medium Utilization Time.

Unlike the CBT, overhearing does not require access to the registers of a WiFi adapter. However, the WiFi adapter needs to support *monitor mode* to calculate the Medium Utilization Time (MUT). Further, the MUT only considers the fraction of the time medium utilized by IEEE 802.11 frames, and IEEE 802.11 operates in the ISM band, which is also used by other protocols (Bluetooth, IEEE 802.15.4) and devices (such as Microwave oven). Furthermore, the node requires to process each frame to calculate the MUT.

### 5.3.2 Operation of the TTBA

This section introduces the Time Token Bucket Algorithm (TTBA) used for traffic control. As discussed earlier in Sec. 5.2, when a node starts, a medium occupancy ratio $mor^{v}_{assgn}$ is assigned to the node $v$ and, the Control and Monitor Module calculates the time token bucket parameters. Based on the configuration, the TTBA

performs traffic shaping. Then, traffic monitoring is performed using the metrics derived by monitoring the operation of the time token bucket. The frames from the higher layers of the WiPS arrive in the MML-arrival queue and are processed based on their stream priority $s_{prio}$. When a frame is selected from the queue, it checks whether the transceiver queue is empty. For monitoring purposes, we restrict the number of frames in the transceiver queue to 1. Next, we discuss the traffic shaping using the time token bucket, followed by the metric developed from observing the operation of TTBA for traffic monitoring. Then, we discuss the improvements introduced to improve the initial version.

#### 5.3.2.1 Traffic Shaping using the time token bucket

Traffic shaping is performed by using the time token bucket discussed in Sec. 5.3.1.1 and controlling the operating parameters of the algorithm. Based on the requirements of each node, the operating parameters of the algorithm are configured. The following parameters control the functioning of the time token bucket.

- Token Size: A token represents the medium occupancy time permitted to the node. The token size or duration of tokens used in node $v$ is denoted as $d^v_{token} \in \mathbb{N}_0[\mu s]$.

- Token refill interval: It is the time interval between refilling tokens. The token refill interval of node $v$ is denoted as $d^v_{refill} \in \mathbb{N}_0[\mu s]$.

- Token bucket size: Unlike the conventional token bucket algorithm, where the bucket holds tokens that represent the permission to send a frame, the time token bucket holds transmission time. The token bucket size controls the maximum time or duration node $v$ may occupy the medium continuously, which is also the maximum burst length. It is denoted as $d^v_{txMax} \in \mathbb{N}_0[\mu s]$.

In the initial version [MKG17], the token size $d^v_{token}$ is the same for all nodes $v$. For instance,

$$\forall v \in V^{int} \quad d^v_{token} = 20\mu s \tag{5.2}$$

On each refill, a transmission time of 20 µs is added to the time token bucket. Next, the token refill interval $d^v_{refill}$ is calculated depending on the assigned medium occupancy ratio $mor^v_{assgn}$.

$$mor^v_{assgn} = \frac{d^v_{token}}{d^v_{refill}} \tag{5.3}$$

While using a fixed token size, reducing the refill interval increases the medium occupancy ratio, and increasing the refill interval decreases the medium occupancy ratio. Thus, by controlling the refilling, outgoing traffic from a node is controlled.

Further, to fill an empty bucket of size $d^v_{txMax}$, $n^v_{fillings}$ number of tokens are required.

$$n^v_{fillings} = \left\lceil \frac{d^v_{txMax}}{d^v_{token}} \right\rceil \text{[Tokens]} \tag{5.4}$$

Furthermore, the time required to refill an empty bucket $d^v_{fillTime}$ is calculated as,

$$d^v_{fillTime} = n^v_{fillings} \cdot d^v_{refill} \tag{5.5}$$

An alternative approach is to calculate the token size $d^v_{token}$ depending on the assigned medium occupancy ratio $mor^v_{assgn}$ by configuring the same refill interval $d^v_{refill}$ for all nodes. With this approach, all nodes in the network have the same periodic refill interval. While using the MML, token refill instances are potential time-points where a frame in the MML-arrival queue is pushed into the lower WiPS MAC Layer, and then, to the transceiver queue, i.e., the time points when a node attempts transmission. The bucket refilling configurations influence this timing. Therefore, if the nodes use the same refill interval $d^v_{refill}$, it affects the link reliability.

The former approach, calculating the refill interval $d^v_{refill}$ by assigning the same token size $d^v_{token}$ to all nodes, does not completely resolve the problem. If the medium occupancy ratio $mor^v_{assgn}$ assigned to two or more nodes are the same, those nodes have the same bucket configurations. Moreover, the calculated refill interval for a node, is constant, i.e., token refills are periodic. Furthermore, even if nodes use a different refill interval, nodes could still attempt transmission at a similar (or almost) point, i.e., the Least Common Multiple (LCM) of refill interval values used by each node. Thus, the frame collision probability increases, even though the underlying random-backoff mechanism of the IEEE 802.11 focuses on reducing it, and, particularly if the node acts as hidden stations. As discussed in Chapter 3, the random-backoff mechanism is only performed if the medium is not idle during a transmission attempt. Else, the transceiver transmits without a random-backoff. To overcome this limitation, we introduce a random interval between the token refills in Sec. 5.3.3. In this chapter, we refer to the token refilling mechanism in the original version of TTBA as *static refilling* and the revised mechanism (introduced in Sec. 5.3.3) as *dynamic refilling*.

Fig. 5.3 depicts the operation of TTBA with *static refilling*. Frames arriving in the MML-arrival queue are processed depending on the stream priorities. When a frame is processed, it checks whether the transceiver queue is empty. Once the node

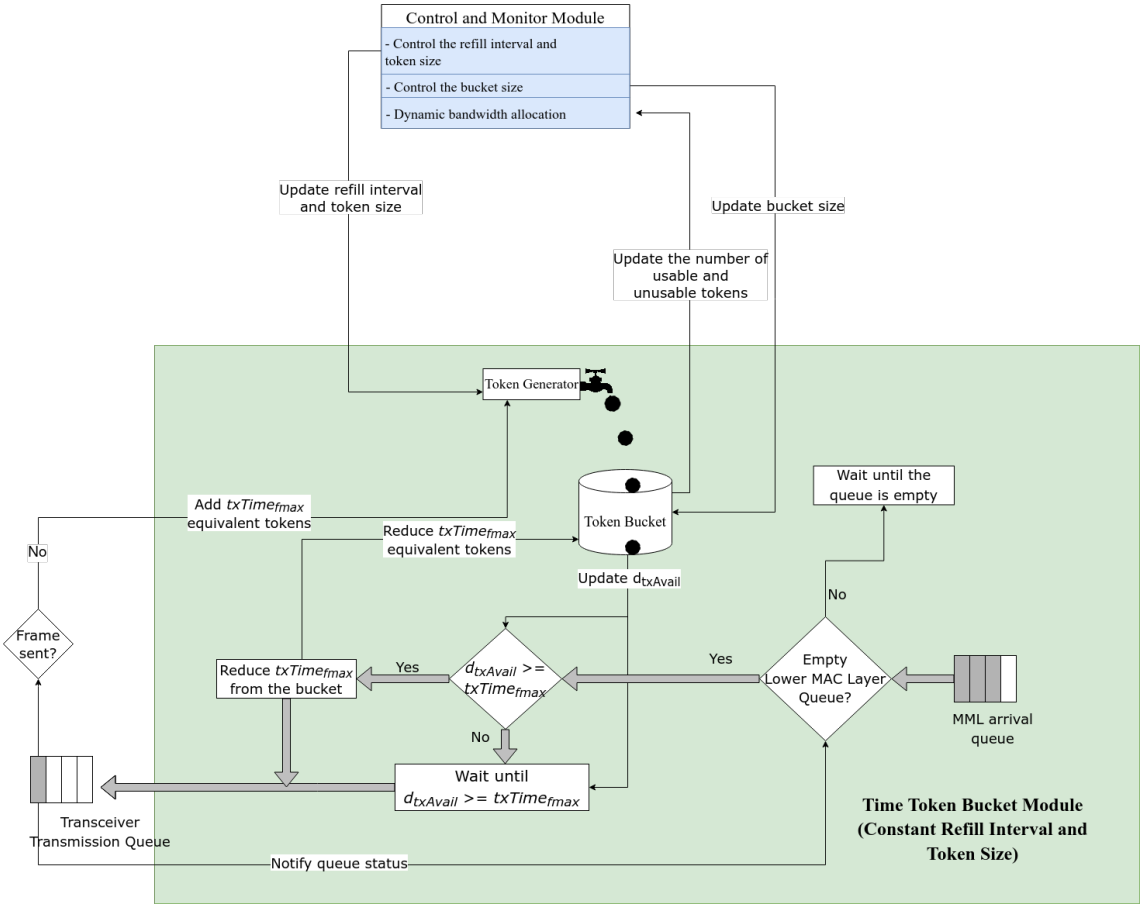



Figure 5.3: Time Token Bucket Module (Original version using *static refilling*)

satisfies that condition, the frame attempts to consume the estimated transmission time $txTime_{f,max}$ from the token bucket. The transmission time estimated for frame $f$ is denoted as $txTime_{f,max} \in \mathbb{N}_{>0}[\mu s]$, and it is calculated as:

$$txTime_{f,max} = \left\lceil \frac{s_{payload}}{r_{payload}} + \frac{MLO}{r_{header}} + PLO \right\rceil \mu s \tag{5.6}$$

where, $\frac{s_{payload}}{r_{payload}}$ calculates the time required to send the payload of size $s_{payload}$ at a transmission rate $r_{payload}$. In addition, the transmission time required to send IEEE 802.11 header is calculated. MAC-layer overhead $MLO$ [Bytes] is the size of the IEEE 802.11 header, and the header is transmitted at a rate $r_{header}$. The IEEE 802.11 header is transmitted at a different rate to support the legacy devices. Furthermore, the average Physical Layer Overhead $PLO$ [µs] is added. It consists of the time required to send the preamble, interframe spacing, i.e., DCF Interframe Space (DIFS), and the average Backoff-Interval.

The transmission time available in the token bucket of node $v$, i.e., the sum of tokens in the bucket, is denoted as $d^v_{txAvail} \in \mathbb{N}_0[\mu s]$. When the token bucket has enough transmission time to send frame $f$, i.e., $d^v_{txAvail} \geq txTime_{f,max}$, the frame is forwarded to the WiPS MAC layer. Else, frame $f$ waits until enough time tokens are refilled into the bucket. Frame $f$ only consumes the required amount of transmission time ($d^v_{txAvail}$) for sending the frame, i.e., it may consume the whole transmission time available in the bucket or a portion of it. Moreover, the transmission time required for frame $f$ must be less than the maximum burst or the bucket size, i.e.,

$$txTime_{f,max} \leq d^v_{txMax} \tag{5.7}$$

Else, depending on the MML settings, either the frame is discarded, or the bucket size is increased to support future arrivals of frames with the larger size.

Once frame $f$ is pushed into the lower layer by consuming the required transmission time, and if the frame is successfully transmitted, then, next frame is drawn from the MML-arrival queue. If the node fails to send the frame [16], the tokens consumed by the frame are issued back into the bucket since the node does not use the consumed transmission time.

As depicted in Fig. 5.3, we restrict the number of frames in the transceiver queue to 1. Otherwise, when a frame transmission is delayed due to medium overload, subsequent frames get accumulated in the transceiver queue. When the medium is available, the transceiver may consecutively transmit all the frames queued up in the transceiver queue. Thus, the burst size may exceed the bucket size, and the MML cannot control it. By restricting the number of frames in the transceiver queue to 1, if the frame forwarded into the transceiver queue (by consuming the required transmission time) is delayed, the tokens get accumulated in the token bucket, and eventually, result in bucket overflow. This principle is used in Sec. 5.3.2.2 to derive the metric used for medium monitoring.

Next, we explain the calculation of parameters used for traffic shaping using an example. Let us calculate the required ratio of medium occupancy for node $v$ to service an application $app_1$. Node $v$ periodically sends data with a payload size, $s_{payload} = 100$ Bytes. The payload is sent at a transmission rate, $r_{payload} = 1$ Mbps. The IEEE 802.11 MAC Layer overhead of size $MLO = 34$ Bytes is sent at 1 Mbps. The checksum and the Physical Layer Overheard approximate, $PLO = 274\,\mu s$, consisting of the DIFS delay of 50 µs, average random-backoff delay of 95 µs, and 144 µs

[16] We restrict the number of retires to 1, and therefore, the frame is discarded after maximum backoff time.

for preamble. Eqn. 5.6 is used to calculate the transmission time to send frame *f*, $txTime_{f,max}$:

$$txTime_{f,max} = \frac{100\,\text{Bytes}}{1\,\text{Mbps}} + \frac{34\,\text{Bytes}}{1\,\text{Mbps}} + 289\,\mu\text{s} = 1361\,\mu\text{s}$$

I.e., the token bucket requires a minimum bucket size $d^v_{txMax} = 1361\,\mu\text{s}$ to support the application $app_1$. The number of tokens required to refill an empty bucket, $n^v_{fillings}$, is calculated using Eqn. 5.4.

$$n^v_{fillings} = \left\lceil \frac{1361\,\mu\text{s}}{20\,\mu\text{s}} \right\rceil = 69\,[\text{Tokens}]$$

For instance, if application $app_1$ transmits 100 Bytes [17] of payload every 1500 ms, the node requires a medium occupancy ratio,

$$mor^v_{assgn} = \left\lceil \frac{1361\,\mu\text{s}}{1500\,\text{ms}} \right\rceil \approx 0.091\%$$

For a token size, $d^v_{token} = 20\,\mu\text{s}$ , the required refill interval $d^v_{refill}$ to support the application is calculated using Eqn. 5.3.

$$d^v_{refill} = \frac{20\,\mu\text{s}}{0.091\%} \approx 21978\,\mu\text{s} = 21.97\,\text{ms}$$

Thus, while refilling the time token bucket with tokens of size 20 µs every 21.97 ms, an empty bucket is refilled in approx 1500 ms. I.e., using Eqn. 5.5,

$$d^v_{fillTime} = n^v_{fillings} \cdot d^v_{refill} = 69 \cdot 21.97\,\text{ms} = 1516\,\text{ms} \tag{5.8}$$

Hence, using the above bucket configuration, node $v$ is permitted to use the medium approximately every 1500 ms to send a frame of size with payload $s_{payload} = 100$ Bytes at rate $r_{payload} = 1$ Mbps.

#### 5.3.2.2 Traffic Monitoring using the time token bucket

Traffic monitoring is performed by monitoring the ongoing traffic through the MML and the operations of the TTBA. Unlike the methods mentioned in Sec. 5.3.1.2, this approach is hardware-independent and has a low overhead. The following method and metrics were introduced in [MKG17]. An overflow of the token bucket occurs when a node does not utilize the assigned medium occupancy ratio $mor^v_{assgn}$. It

[17] Here, we are not considering requirements for ACK frames.

results in wasted tokens. By assessing the causality of wasted tokens, it is possible to evaluate the medium conditions. The tokens are wasted due to two reasons:

1. When the assigned medium occupancy ratio is higher than the required medium occupancy for a node, i.e., $mor^{v}_{assgn} > mor^{v}_{required}$. As mentioned in Sec. 5.2, based on the requirements of a node, a certain ratio of the medium occupancy is assigned to the node. When the node does not use the assigned medium occupancy ratio, the bucket keeps filling until it overflows, resulting in wasted tokens. Such tokens are called *usable wasted tokens*. These are usable tokens or transmission time units, and are wasted as the node does not require them.

2. When the network is overloaded, $mor^{v}_{assgn} > mor^{v}_{avail}$. When a frame is forwarded into the hardware queue, it is delayed depending on the medium condition. If the frame remains longer in the transceiver queue due to high contention, then the succeeding frames in the MML-arrival queue are delayed, and subsequently, the token bucket overflows. In this scenario, the tokens overflow, as the tokens in the bucket are unused due to medium overload. Such tokens are called *unusable wasted tokens*. It indicates that the medium is overloaded either due to the traffic from external or internal devices.

Two counters monitor the usable and unusable wasted tokens, $d^{v}_{WastedUsable}$ and $d^{v}_{WastedUnUsable}$, respectively. When a token is wasted, the respective counter is incremented by the size or duration of the wasted token. Thus, a counter denotes the wasted medium occupancy time assigned to the node. The wasted tokens are classified as usable or unusable depending on the transceiver queue and the MML-arrival queue status.

1. For node $v$, counter $d^{v}_{wastedUnUsable}(t)$ is the sum of *unusable wasted tokens* from the start until time $t$. It is incremented by the size of token wasted when the following conditions are satisfied.
   - The token bucket is full, and
   - the transceiver queue and the MML-arrival queue are not empty.

2. Similarly, for node $v$, counter $d^{v}_{wastedUsable}(t)$ is the sum of *usable wasted tokens* from the start until time $t$. It is incremented by the size of token wasted while
   - the token bucket is full, and
   - the transceiver queue or the MML-arrival queue is empty.

We derive additional metrics to monitor the medium using the parameters mentioned earlier. $mor^{v}_{wastedUnUsableRatio} \in [0,1]$ is the ratio of unusable wasted medium occupancy ratio assigned to node $v$. It is calculated as:

$$mor^{v}_{wastedUnUsableRatio}(t) = \frac{d^{v}_{wastedUnUsable}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})} \tag{5.9}$$

where, $t_{start}$ is the time at which the node starts.

Similarly, $mor^{v}_{wastedUsableRatio} \in [0,1]$ is the ratio of usable wasted medium occupancy ratio assigned to node $v$. It is calculated as:

$$mor^{v}_{wastedUsableRatio}(t) = \frac{d^{v}_{wastedUsable}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})} \tag{5.10}$$

For instance, $mor^{v}_{wastedUnUsableRatio} = 0.5$ means that the medium is overloaded, and 50% of the medium occupancy ratio $mor^{v}_{assgn}$ assigned to node $v$ is wasted. I.e., the medium occupancy ratio assigned to the node is higher than available. This knowledge is used in [KMG19b] to measure the available medium occupancy to the nodes. We use ratio of wasted unusable assigned medium occupancy ratio $mor^{v}_{wastedUnUsableRatio}$ as a metric to monitor medium overload.

Further, $mor^{v}_{wastedRatio}(t)$ is the ratio of wasted (or unused) assigned medium occupancy ratio by node $v$ until time $t$ from the start of medium monitoring. It is the sum of the usable and the unusable ratio of wasted assigned medium occupancy ratio. I.e.,

$$mor^{v}_{wastedRatio}(t) = mor^{v}_{wastedUnUsableRatio}(t) + mor^{v}_{wastedUsableRatio}(t) \tag{5.11}$$

Then, the ratio of used assigned medium occupancy ratio until time $t$ is

$$mor^{v}_{usedRatio}(t) = 1 - mor^{v}_{wastedRatio} \tag{5.12}$$

An alternative way to calculate the $mor^{v}_{usedRatio}$ is by calculating the ratio of time node $v$ used the medium for transmission. The sum of transmission time required to send each frame by node $v$ until time $t$ is calculated as,

$$d^{v}_{tx}(t) = \sum_{\forall f \in F^{v}(t)} \frac{s^{f}_{payload}}{r^{f}_{payload}} + \frac{MLO}{r_{header}} + PLO \tag{5.13}$$

where, $F^{v}(t)$ is the set of frames sent by node $v$ until time $t$. $s^{f}_{payload}$ is the payload size of frame $f$. Similarly, $r^{f}_{payload}$ is the transmission rate at which the payload of

frame $f$ is transmitted. $MLO$ is the medium layer overhead consisting of the IEEE 802.11 header, and $r_{header}$ is the rate at which it is sent. The remaining parameter, $PLO$, is the average time required for Physical Layer Overhead.

Then, the ratio of used assigned medium occupancy ratio by node $v$ until time $t$ from the time node starts $t_{start}$,

$$mor^{v}_{usedRatio}(t) = \frac{d^{v}_{tx}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})} \tag{5.14}$$

Thus, the sum of used and wasted ratio of assigned medium occupancy ratio until time $t$ from the time node starts $t_{start}$ equals 1.

$$\underbrace{\frac{d^{v}_{tx}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})}}_{mor^{v}_{usedRatio}(t)} + \underbrace{\frac{d^{v}_{wastedUnUsable}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})} + \frac{d^{v}_{wastedUsable}(t)}{mor^{v}_{assgn} \cdot (t - t_{start})}}_{mor^{v}_{wastedRatio}(t)} = 1 \tag{5.15}$$

The quality of the metric $mor^{v}_{wastedUnUsableRatio}$ is evaluated by comparing it to the metric *Channel Busy Fraction* (CBF) mentioned in Sec. 5.3.1.2. *Channel Busy Time* (CBT) is the total time transceiver sensed the medium occupied, including the medium usage time by the node [NET; CFG]. The parameter $d^{v}_{CBT}(t)$ denotes the CBT detected by node $v$ until time $t$. Using CBT, the *Channel Busy Fraction* (CBF), $tr^{v}_{CBF}$ is calculated. $tr^{v}_{CBF}(t)$ is the ratio of time the node $v$ sensed the medium as busy from beginning $t_{start}$ till time $t$.

$$tr^{v}_{CBF}(t) = \frac{d^{v}_{CBT}(t) - d^{v}_{CBT}(t_{start})}{t - t_{start}} \tag{5.16}$$

Further, *Medium Energy Sensed* $d^{v}_{MES}(t)$ is defined as the amount of time node $v$ sensed the medium busy, excluding the medium usage time by node $v$ from the beginning $t_{start}$ till time $t$.

$$d^{v}_{MES}(t) = ((d^{v}_{CBT}(t) - d^{v}_{CBT}(t_{start})) - d^{v}_{tx}(t) \tag{5.17}$$

Then, $tr^{v}_{MES}(t)$ is the ratio of the $d^{v}_{MES}(t)$ by node $v$ from beginning $t_{start}$ till time $t$. It is calculated as

$$tr^{v}_{MES}(t) = \frac{d^{v}_{MES}(t)}{t - t_{start}} \tag{5.18}$$

Later, in Sec. 5.4.1, we discuss the results of the experiments performed to evaluate the quality of the metric $mor^{v}_{wastedUnUsableRatio}$ by comparing to the metrics *Channel Busy Fraction*, $tr^{v}_{CBT}$ and ratio of *Medium Energy sensed*, $tr^{v}_{MES}(t)$.

### 5.3.3 Dynamic token refilling to introduce randomness

The dynamic token refilling mechanism is introduced to mitigate the limitations of *static refilling*. Before discussing the improvements, let us discuss the three scenarios when a frame created by an application transits to the lower WiPS MAC layer.

***Scenario* I:** Without *Medium Management Layer*.

Without traffic shaping, the frame created by applications moves toward the lower stacks and reaches the lower layer of the WiPS framework, the WiPS MAC Layer, and is finally appended to the transceiver queue. In this scenario, transmission trigger time points are completely dependent on the application, i.e., whether the frames are created at periodic or aperiodic intervals.

***Scenario* II:** With *Medium Management Layer* and ***insufficient*** filling of the time token bucket.

With traffic shaping, the frames created by applications move toward the lower stacks and pass through the MML before reaching the lowest layer of WiPS, the WiPS MAC Layer. Depending on the bucket filling, the frames that arrive at the MML-arrival queue either wait or move forward. Consider the scenario when the transmission time $d^{v}_{txAvail}$ available in the bucket is less than the transmission time required to send frame *f*, i.e., $d^{v}_{txAvail} < txTime_{f,max}$. In such a scenario, the frame waits until the required transmission time is filled into the bucket. Hence, the time the MML forwards the frame to the WiPS MAC Layer, and later into the transceiver queue depends on the token refilling interval and token size.

Constant and periodic refill intervals could potentially result in two or more nodes trying to transmit at (almost) the same point. Contrary to using the same refilling intervals $d^{v}_{refill}$ for refilling tokens, nodes could still attempt transmission at a similar point since refilling intervals are periodic. I.e., the LCM of refill interval values used by each node. Therefore, while using a constant refill interval, these issues need to be considered while configuring parameters used to control the TTBA.

***Scenario* III:** With *Medium Management Layer* and ***sufficient*** filling of the time token bucket.

As in scenario II, the frame created by an application passes toward the MML before reaching the lowest layer of WiPS. Unlike in the previous scenario, consider the scenario when the transmission time $d^{v}_{txAvail}$ available in the bucket is greater than the transmission time required to send frame *f*, i.e., $d^{v}_{txAvail} \geq txTime_{f,max}$.

When a frame arrives in the MML-arrival queue, as the time token bucket has enough transmission time, it immediately consumes the required transmission time, and the MML forwards the frame to the next layer and then to the transceiver queue. In such a scenario, the refilling interval or token size does not affect the transmission trigger points. Thus, the transmission timings depend on the timing characteristics of the particular application, i.e., whether the frames are generated periodically or aperiodically.

#### 5.3.3.1 Modified Workflow to TTBA

We introduce *dynamic refilling*, where the tokens are refilled at dynamic intervals. It mitigates the drawbacks of using constant token refill intervals and reduce the application's dependency on potential transmission trigger points. Fig. 5.4 shows the workflow of the Time Token Bucket using dynamic refill intervals. Similar to in Fig. 5.3, the thicker lines represent the flow of the frame.

The minimum and maximum refill interval duration in dynamic refilling of node $v$ are represented as $d^v_{minRefill}$ and $d^v_{maxRefill}$, respectively. The next refill interval, $d^v_{refill}$, of node $v$ is randomly drawn from a uniform distribution of values between $d^v_{minRefill}$ and $d^v_{maxRefill}$. The token size $d^v_{token}$ for the next refill is calculated accordingly, i.e.,

$$d^v_{token} = d^v_{refill} \cdot mor^v_{assgn} \tag{5.19}$$

where, $d^v_{refill}$ varies for each interval and $mor^v_{assgn}$ is the assigned medium occupancy ratio.

Using a dynamic refill interval mitigates the drawbacks of Scenario II. In this scenario, the frame that arrives at the MML-arrival queue waits for the required transmission time. When the token bucket collects the required number of tokens or transmission time to satisfy the medium occupancy requirement of the frame, it is pushed into the WiPS MAC Layer and then to the transceiver queue. As each refill interval varies, the time interval between any two consecutive frames forwarded by the MML into the lower layer also varies.

To mitigate the drawbacks of Scenario III, in such a scenario, frames are delayed (variable delay) until the arrival of the next refill token. The frames are delayed (until the next refill token arrives) to deal with the free flow of frames in the particular scenario. The delays depend on the configuration parameters $d^v_{minRefill}$ and $d^v_{maxRefill}$. For instance, we use a minimum $d^v_{minRefill}$ and maximum interval duration of $d^v_{maxRefill}$ of 50 µs and 1000 µs, respectively, in one of the experiments. In Sec. 5.4.2, we compare the two refilling mechanisms.

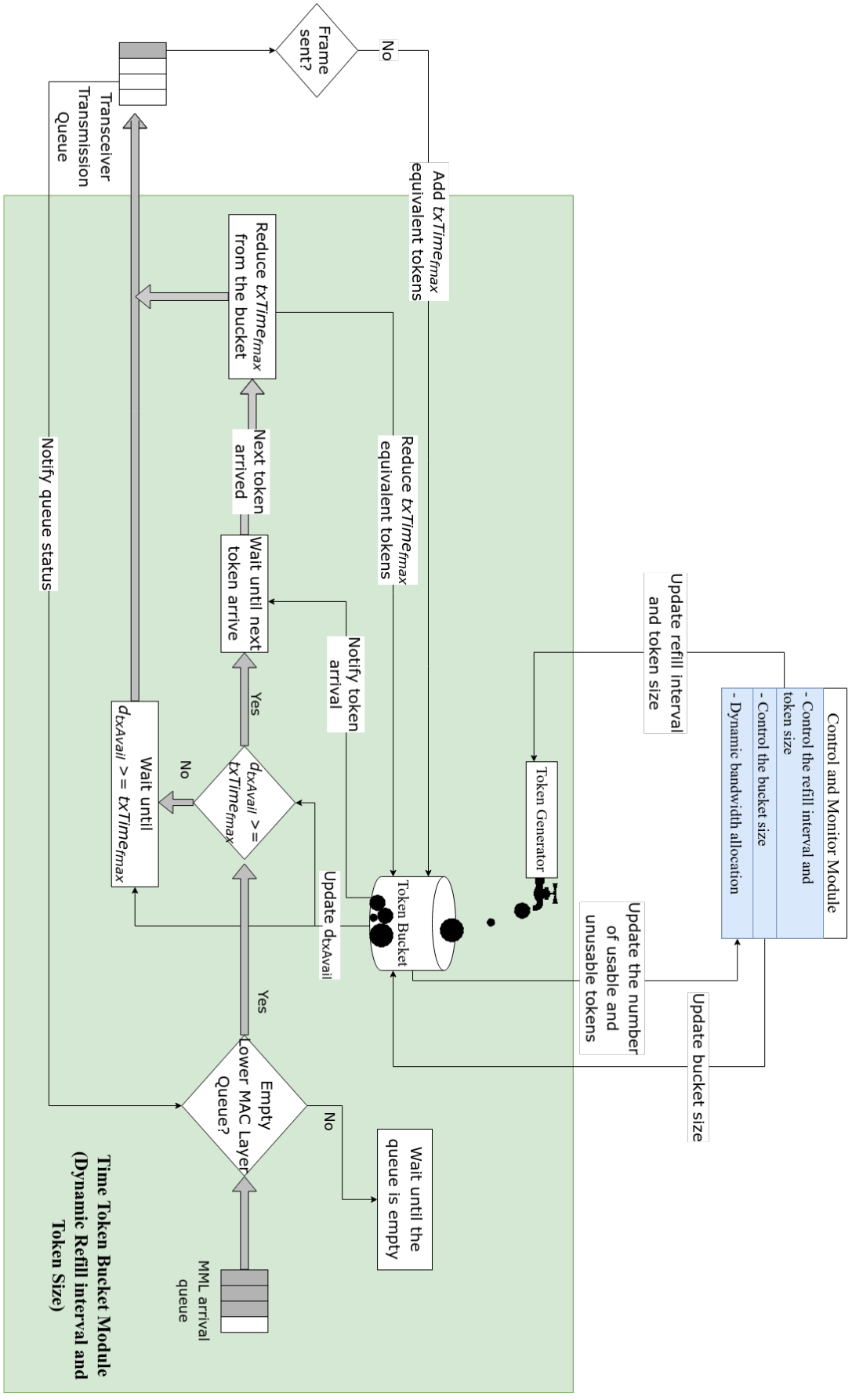



Figure 5.4: Time Token Bucket Module with dynamic token refilling interval.

Concluding this section, we begin by discussing the state of art methods for traffic shaping in communication networks and traffic monitoring in IEEE 802.11 networks. Then, we discussed the operation of the Time Token Bucket Algorithm (TTBA) for traffic shaping and monitoring in IEEE 802.11 contention-based networks. Further, we introduced the metric, ratio of unusable wasted assigned medium occupancy ratio $mor^{v}_{wastedUnUsableRatio}$, which is hardware-independent and has a low-overhead to monitor medium conditions. Finally, we presented the *dynamic refilling* to improve the original TTBA.

## 5.4 Experiments

In this section, we discuss the experiment setups and results to evaluate the applicability of the metric *ratio of unusable wasted medium occupancy ratio*, $mor^{v}_{wastedUnUsableRatio}$, and improvements by using *dynamic refilling* over *static refilling* for the Time Token Bucket Algorithm (TTBA). First, in Sec. 5.4.1, we discuss the findings from the experiments conducted to evaluate the applicability of metric $mor^{v}_{wastedUnUsableRatio}$ under multiple traffic conditions. Later, in Sec. 5.4.2, we discuss the experimental findings that show the improvements by using *dynamic refilling* for TTBA.

### 5.4.1 Evaluating the traffic monitoring metric, *unusable wasted assigned medium occupancy ratio*

Multiple experiments are conducted to evaluate the applicability of the metric, $mor^{v}_{wastedUnUsableRatio}$, to monitor medium overload. The experiment results presented in this section are first reported in [MKG17]. We conduct multiple experiments in the presence and absence of external traffic. A static and mobile testbed consisting of six Raspberry Pi [Gay14] nodes running Arch Linux with Linux kernel version 4.6.6-V7+ is used. It is possible to power the Single-Board Computer, Raspberry Pi, with direct power and a battery, creating the Raspberry Pi as a good choice for the static and, particularly, for the mobile testbed. A Logilink WL0084B USB WiFi adapter with Ralink Chipset using rt2800 driver is attached to each Raspberry Pi. The experiments in the presence of external traffic are conducted on the static testbed, which is located inside a university office building where external networks of multiple working groups are operating. The experiments in the absence of external traffic are conducted in the basement of the building. Regardless, in

both experiments, the testbed nodes are in single-hop distance, i.e., each node in the network can communicate with each other.

In the experiments, the applications running on each node transmit a frame with a constant payload of size $s_{payload} = 480B$ at a transmission rate $r_{payload} = 1Mbps$. The IEEE headers are also transmitted at a rate $r_{header} = 1Mbps$. The remaining parameters $MLO$ and $PLO$ are 34 Bytes and 289 µs, respectively. Applying Eqn. 5.6, the transmission time required is

$$txTime_{f,max} = \frac{480B}{1Mbps} + \frac{34B}{1Mbps} + 289\,\mu s \approx 4211\,\mu s$$

In the following experiments, the MML-arrival queue is always filled. Thus, whenever there is enough transmission time available in the bucket to transmit a frame, a frame waiting in the MML-arrival queue consumes it. Therefore, if available, the assigned medium occupancy ratio $mor^v_{assgn}$ is used, and this experiment setup results in zero usable wasted tokens. We conduct multiple experiments with different assigned medium occupancy ratios $mor^v_{assgn}$ to the nodes (resulting in different network loads), and compare our metric with other metrics. All experiments are conducted in 2.4 GHz. The first group of experiments (in Sec. 5.4.1.1) is conducted to assess the sensitivity of our metric where the network load generated by the testbed is low. The second group of experiments (in Sec. 5.4.1.2) is conducted with higher network loads.

#### 5.4.1.1 Experiments to assess the sensitivity of $mor^v_{wastedUnUsableRatio}$

Experiments A.I, A.II, and A.III are conducted to assess the sensitivity of the metric. We compare the results of the metric $mor^v_{wastedUnUsableRatio}$ to other metrics *Channel Busy Fraction* $tr^v_{CBF}$ and ratio of the *Medium Energy Sensed* $tr^v_{MES}(t)$ in different network loads. The testbed nodes are operated in channel 6, i.e., 2.437 GHz as the central frequency during the experiments.

**Experiment A.I**

This experiment is conducted in the absence of external traffic. Tab. 5.1 shows settings for experiments A.I and A.II. As stated in the table, each node generates traffic that requires a medium occupancy ratio of 1%. Thus, the testbed network uses 6% of the total medium occupancy ratio. Since there is no external traffic, the medium is available to satisfy the required medium occupancy ratio ($mor^G_{required} = 6\%$) by the testbed network.

| **Parameter** | **Value** |
|---|---|
| $d^v_{txMax}(\forall v \in V^{int})$ | 4545 µs |
| $d^v_{token}(\forall v \in V^{int})$ | 1 µs |
| $mor^v_{assgn}(\forall v \in V^{int})$ | 1% |
| $mor^G_{assgn}$ | 6% |
| $d^v_{refill}(\forall v \in V^{int})$ | 100 µs |
| Experiment duration | 55 min |

Table 5.1: Parameters for the experiments A.I and A.II

| Node ID | $d^v_{wastedUnUsable}(t)$ | $mor^v_{wastedUnUsableRatio}(t)$ | $tr^v_{CBF}(t)$ | $tr^v_{MES}(t)$ |
|---|---|---|---|---|
| 38 | 0 | 0% | 5.70% | 4.79% |
| 39 | 0 | 0% | 7.80% | 6.88% |
| 40 | 0 | 0% | 8.14% | 7.21% |
| 41 | 0 | 0% | 5.80% | 4.87% |
| 42 | 0 | 0% | 5.74% | 4.81% |
| 43 | 0 | 0% | 6.36% | 5.44% |

Table 5.2: Metric values from the experiment A.I where $t = 3\,300\,000\,000$ µs or 55 min

Tab. 5.2 shows the results of the experiment. The table shows the reported metric values until the end of the experiment, i.e., $t = 55$ min. No tokens (*unusable wasted tokens*) were wasted in any of the nodes. Thus, our metric $mor^v_{wastedUnUsableRatio}$ is zero at all nodes, indicating no medium overload. I.e., all nodes have enough medium occupancy time to support the traffic generated by them, i.e., $mor^v_{assgn} < mor^v_{avail}$. Further, the table shows the metrics $tr^v_{CBF}$ and $tr^v_{MES}(t)$ reported during the experiment.

**Experiment A.II**

Tab. 5.1 shows the settings for the experiment. Similar to Experiment A.I, there are six nodes where each node generates traffic that requires 1% of medium occupancy ratio. Thus, the network in total requires 6% of the medium occupancy time. However, different from the previous experiment, this experiment is conducted in the presence of external traffic.

Tab. 5.3 shows the metric values at the end of the experiment. As testbed nodes share the medium with external nodes, tokens were wasted on multiple nodes in

| Node ID | $d^v_{wastedUnUsable}(t)$ | $mor^v_{wastedUnUsableRatio}(t)$ | $tr^v_{CBF}(t)$ | $tr^v_{MES}(t)$ |
|---|---|---|---|---|
| 7 | 293,358 µs | 0.89% | 24.80% | 23.83% |
| 8 | 232,685 µs | 0.71% | 19.87% | 18.89% |
| 22 | 13,686 µs | 0.04% | 16.90% | 15.91% |
| 23 | 249,952 µs | 0.76% | 24.57% | 23.59% |
| 28 | 42,648 µs | 0.13% | 17.90% | 16.92% |
| 29 | 11,152 µs | 0.03% | 14.32% | 13.32% |

Table 5.3: Metric values from the experiment A.II where $t = 3\,300\,000\,000$ µs or 55 min

varying amounts. I.e., seldom there were points during the experiment where the medium was overloaded. For instance, on node 7, 293 356 unusable tokens were wasted, resulting in $d^v_{wastedUnUsable} = 293\,356$ µs (293 356 tokens of size $d^v_{token} = 1$ µs). The experiment was conducted for 55 min, or 3 300 000 000 µs and 1% of the medium occupancy time was assigned to node 7, i.e., 1% of the 3 300 000 000 µs. Then, the *ratio of unusable wasted medium occupancy ratio*,

$$mor^7_{wastedUnUsable}(55\text{min}) = \frac{293\,356\,\mu\text{s}}{3\,300\,000\,000\,\mu\text{s} \cdot 1\%} = 0.89\% \quad \text{applying Eqn. 5.9}$$

Similarly, on nodes 8 and 23, a relatively higher number of tokens are lost than on other nodes. Nodes 8 and 23 reported an *ratio of unusable wasted medium occupancy ratio* of 0.71% and 0.76%, respectively. Furthermore, on the same nodes 7, 8, and 23, the WiFi hardware sensed medium busy for a longer period than other nodes. The $tr^v_{CBF}(t)$ and $tr^v_{MES}(t)$ metric values reflect it. On the other hand, node 29, which wasted the lowest percentage of assigned medium occupancy ratio, sensed medium busy for a shorter period than other nodes. This experiment results show that the nodes that sensed a comparatively longer duration of medium energy (higher contention to access the medium) lost more unusable tokens or could not use the assigned share of the medium, thus, a higher value for the metric $mor^v_{wastedUnUsableRatio}$.

**Experiment A.III**

In this experiment, we reduce the traffic generated by the network by half compared to the previous experiment. Both experiments (A.II and A.III) were conducted during office time. Tab. 5.4 shows the parameters for the experiment. Each node generates traffic that requires 0.5% of the medium occupancy time. Thus, the six nodes in total require 3% of the medium occupancy time.

| **Parameter** | **Value** |
|---|---|
| $d^v_{txMax}(\forall v \in V^{int})$ | 4545 µs |
| $d^v_{token}(\forall v \in V^{int})$ | 1 µs |
| $mor^v_{assgn}(\forall v \in V^{int})$ | 0.5% |
| $mor^G_{assgn}$ | 3% |
| $d^v_{refill}(\forall v \in V^{int})$ | 200 µs |
| Experiment duration | 55 min |

Table 5.4: Parameters for the experiments A.III

| Node ID | $d^v_{wastedUnUsable}(t)$ | $mor^v_{wastedUnUsableRatio}(t)$ | $tr^v_{CBF}(t)$ | $tr^v_{MES}(t)$ |
|---|---|---|---|---|
| 7 | 8,164 µs | 0.02% | 14.80% | 14.31% |
| 8 | 4,726 µs | 0.01% | 10.25% | 9.76% |
| 22 | 0 | 0% | 11.86% | 11.37% |
| 23 | 8,527 µs | 0.03% | 13.73% | 13.24% |
| 28 | 0 | 0% | 12.74% | 12.24% |
| 29 | 0 | 0% | 8.39% | 7.89% |

Table 5.5: Metric values from the experiment A.III where $t = 3,300,000,000$ µs or 55 min

Tab. 5.5 shows the results of the experiment. Compared to the previous experiment, nodes 22, 28, and 29 reported or lost zero unusable wasted tokens, i.e., transmissions were never delayed than the estimated due to medium overload. However, nodes 7, 8, and 23, which also reported (a higher amount of) wasted tokens in the previous experiment, lost a relatively lower number of tokens in this experiment. Similar to the previous experiment, the nodes that lost most tokens sensed a comparatively longer period of medium energy. It is reflected by the metrics $tr^v_{CBF}$ and $tr^v_{MES}$. The results show that the proposed metric $mor^v_{wastedUnUsableRatio}$ is sensitive to changes in the medium condition and could detect medium overload locally.

Another notable observation from experiments A.II and A.III is the difference in the duration of medium sensed as busy. Comparing the values of the metrics $tr^v_{CBF}$ and $tr^v_{MES}$ from Tab. 5.3 and 5.5, an increase in 3% of the network traffic created by the testbed increased the value of metrics by (approx) 10%. We repeated these experiments and observed a similar effect. One possible explanation for this higher traffic, the increase in the traffic generated by the testbed might have affected the external APs operating in the same channel, and, for instance, it increased the

collision probability. This leads to further collisions and further retransmissions by external nodes, increasing the overall network load, thus, increasing the duration of medium sensed as busy.

#### 5.4.1.2 Experiments using higher network loads

In this group of experiments, we evaluate the metric $mor^{v}_{wastedUnUsableRatio}$ under higher network loads. In these experiments, the testbed nodes attempt to use 30% to 96% of the medium occupancy time. The medium occupancy ratio assigned to the internal network, $mor^{G}_{assgn}$, is equally split among the six nodes. For instance, in the experiment run where the testbed network generates a load of $mor^{G}_{assgn} = 30\%$, each node generates traffic that requires 5% of the medium occupancy time, i.e., $\forall v \in V^{int}\, mor^{v}_{assgn} = 5\%$. An experiment consists of six runs, wherein each experiment run, the nodes have distinct medium requirements. For instance, in an experiment run, the testbed network generates traffic that requires 30% of the medium occupancy time, while in another experiment run, 54%. In each experiment run, we calculate the metric value at each node at the end of the run. Further, each experiment run is conducted for 20 min.

The experiments are conducted in the presence and absence of external traffic. The first two experiments, experiments A.IV and A.V, are conducted in the presence of external traffic. Both experiments used the same topology. However, experiments were conducted on different channels of the 2.4 GHz ISM band where multiple APs operates on each channel and are located at different locations. Furthermore, both experiments are conducted during office hours. The last experiment, experiment A.VI, is conducted in the absence of external traffic.

**Experiment A.IV**

| **Parameter** | **Value** |
|---|---|
| $d^{v}_{txMax}(\forall v \in V^{int})$ | 4545 µs |
| $mor^{v}_{assgn}(\forall v \in V^{int})$ | $\{5\%; 9\%; 11\%; 13\%; 14\%; 15\%\}$ |
| $mor^{G}_{assgn}$ | $\{30\%; 54\%; 66\%; 78\%; 84\%; 90\%\}$ |
| Experiment duration | $6 \times 20$ min |

Table 5.6: Parameters for the experiments A.IV, and A.V

This experiment is conducted in the presence of external traffic and on channel 1, i.e. 2.412 GHz as the central frequency. Six experiment runs are conducted where

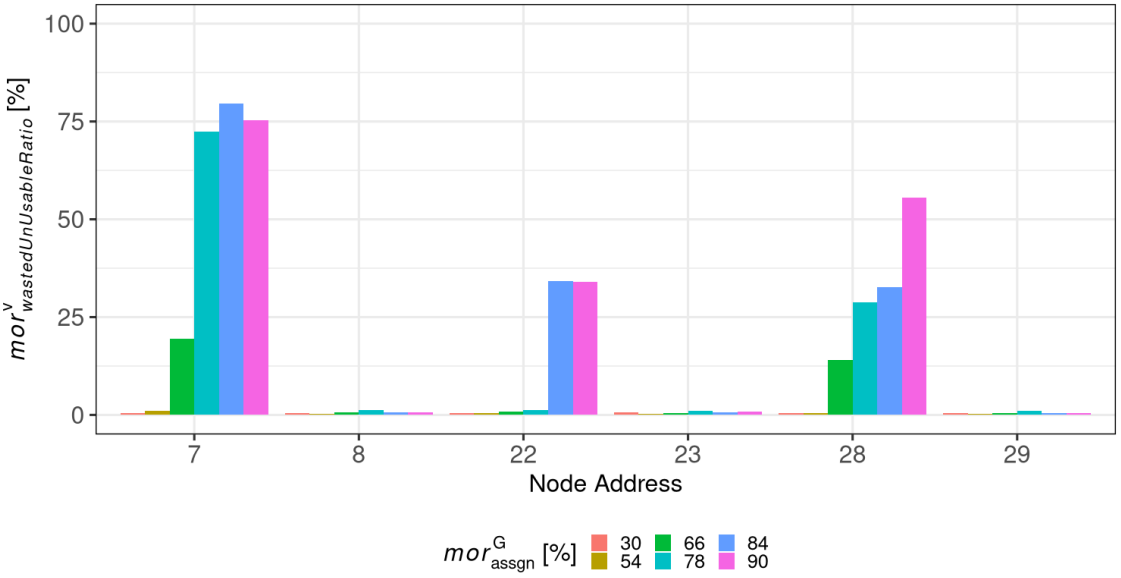



Figure 5.5: $mor^{v}_{wastedUnUsableRatio}(t)$ for nodes from the experiments on channel 1 with distinct medium requirements, where $t = 20\,\text{min}$.

the testbed nodes generate a load between 30% and 90%. Tab. 5.6 shows the settings for each node during each experiment run.

Fig. 5.5 shows the results of each experiment run. The figure shows the $mor^{v}_{wastedUnUsableRatio}$ at each node for the entire duration ($t = 20\,\text{min}$) of an experiment run. As we increased the network load, it initially affected nodes 7 and 28 (with $mor^{G}_{assgn} \geq 66\%$). With further increases, it affected node 22 too (with $mor^{G}_{assgn} \geq 84\%$). As mentioned earlier, the experiments in the presence of external traffic were conducted during office hours, where APs from multiple working groups were located in the same building. An AP of our working group is operating on the same channel (channel 1), and it is located close to node 7. Thus, it is not unexpected that node 7 lost most of the tokens or detected higher network overload than other nodes. The experiment was repeated, a similar pattern was observed.

### Experiment A.V

Next, we conducted the experiment runs on channel 6, i.e., 2.437 GHz. Apart from experimenting on a different channel, this experiment uses the same settings as the previous one. Fig. 5.6 shows the results of the experiment. Unlike the previous experiment, node 22 and 8 were initially affected as we increased the network load. I.e., as we increased the network loads, nodes 22 and 8 observed higher network overload than other nodes.

Further, node 7, which was affected while operating in channel 1 (in the previous experiment), detected relatively low network overload locally in this experiment. The

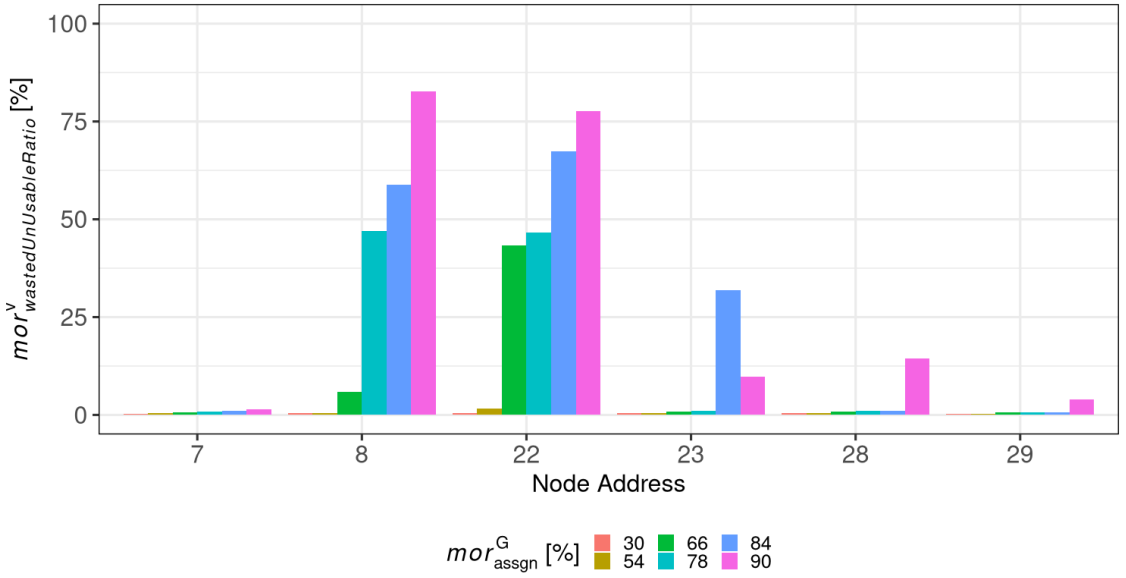



Figure 5.6: $mor^{v}_{wastedUnUsableRatio}(t)$ for nodes from the experiments on channel 6 with distinct medium requirements, where $t = 20$ min.

experiments were repeated, and we observed the same pattern. These experimental results (from experiment A.IV, and A.V ) show that depending on the channel on which the testbed network operates, the nodes that are affected by traffic overload varies. I.e., even though the testbed nodes are in single-hop distance, internal nodes may compete with a varying number of nodes (incl. internal and external) to access the medium. For instance, an external node (or an AP) is in the communication range to some of the internal nodes and a hidden station to the others. Notably, our metric detected traffic overload at each node that depends on the channel and the external networks operating in the channel.

**Experiment A.VI**

Next, we discuss the results of the experiments conducted in the absence of external traffic. This experiment is conducted in a cellar where there are no other functioning wireless devices during the experiment. The nodes are placed close to each other. Tab. 5.7 shows the medium occupancy ratio assigned to the nodes and the testbed during each experiment run. Compared to the assigned medium occupancy ratio to testbed nodes in the previous experiment runs in the presence of external traffic, in this experiment, the highest assigned during an experiment run is 96%. Since there is no external traffic, generating a network load above the capacity enables us to evaluate the impact on nodes.

Fig. 5.7 shows the results of the experiment. The results show that increasing the network load affects the nodes randomly depending on their contention window and

| Parameter | Value |
|---|---|
| $d^{v}_{txMax}(\forall v \in V^{int})$ | 4545 µs |
| $mor^{v}_{assgn}(\forall v \in V^{int})$ | $\{5\%; 9\%; 11\%; 13\%; 15\%; 16\%\}$ |
| $mor^{G}_{assgn}$ | $\{30\%; 54\%; 66\%; 78\%; 90\%; 96\%\}$ |
| Experiment duration | $6 \times 20$ min |

Table 5.7: Parameters for the experiments A.VI

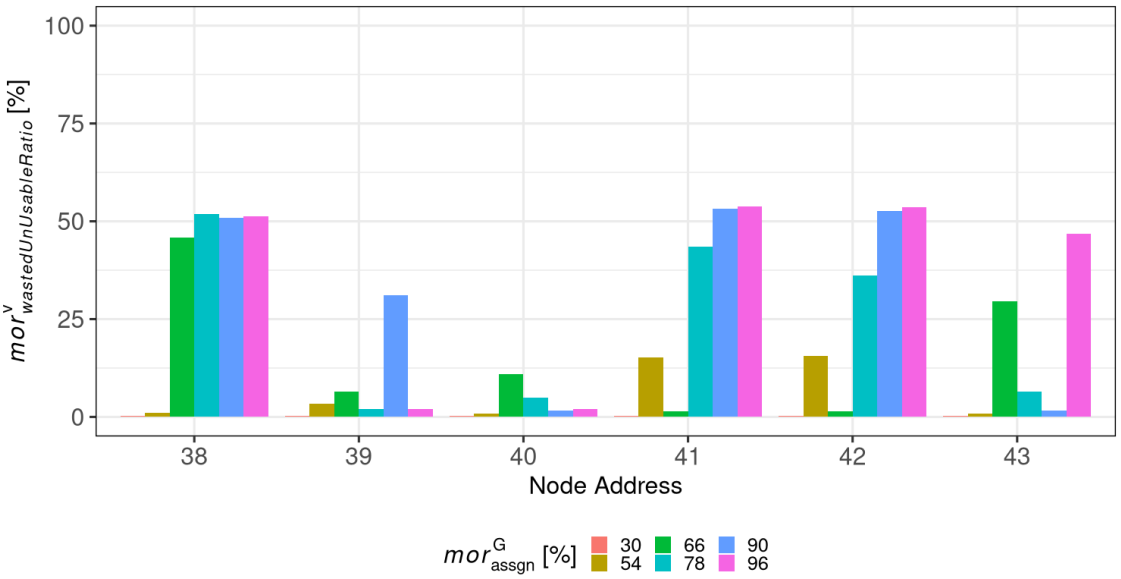



Figure 5.7: $mor^{v}_{wastedUnUsableRatio}(t)$ for nodes from the experiments in the absence of external traffic with distinct medium requirements, where $t = 20$ min.

random backoff. For instance, with a network load of 54% (i.e., $\forall v \in V^{int}\, mor^{v}_{assgn} = 9\%$), the network began to overload, and node 41 and 42 lost most of the tokens or could not use the complete assigned share of medium occupancy time. As we further increased the network load to 66% (i.e., $\forall v \in V^{int}\, mor^{v}_{assgn} = 11\%$), it mostly affected nodes 38 and 43, and nodes 41 and 42 were relatively unaffected. We conducted multiple experiments using the same settings and observed that random nodes were affected when the network started to overload, i.e., with a network load or assigned medium occupancy ratio to the testbed, $mor^{G}_{assgn}$, of 54% and 66%. This randomness is expected as there is no external traffic to influence. The contention for the medium is only between the nodes of the testbed, and thus at each experiment, it affected different nodes.

As we further increased the network load, we observed a similar outcome. For instance, in the experiment run with a network load of 90%, node 39 did not use the complete share of assigned medium occupancy time. However, in the experiment

run with a network load of 96%, node 39 used most of its assigned medium share. Further, in the experiment run where the testbed network generated traffic that requires 96% of the medium occupancy time, nodes 38, 41, 42, and 43 shared impact almost equally. I.e., those nodes could not use nearly half of the medium occupancy ratio assigned to them. However, during the experiments in the presence of external traffic (see the results of experiments A.IV, and A.V), a higher network load affected distinct nodes depending on the topology and channel they operate.

#### 5.4.1.3 Conclusion

The above experiment results show that the metric *ratio of unusable wasted medium occupancy ratio*, $mor^{v}_{wastedUnUsableRatio}$ calculated by monitoring the unusable wasted tokens can detect network overload locally. Experiment results presented in Sec. 5.4.1.1 illustrate the metric's applicability to detect network load locally, similar to the metric CBF. Unlike CBF which is hardware dependent and often restricted in practice as only some drivers permit to poll the required register values to calculate it, our metric is hardware-independent. Moreover, our metric depends on the contention faced by a node while attempting transmissions. In Sec. 5.4.1.2, experiments under multiple network loads and medium conditions were conducted. The results show that the external traffic affected the testbed nodes differently, even in a single-hop network. While in the absence of external traffic, an overloaded network affected the nodes depending on their contention window and random backoff.

### 5.4.2 Evaluating the *dynamic refilling* mechanism

This section discusses the experimental setup and results of experiments performed to illustrate the improvements by using *dynamic refilling* for TTBA. We conducted the experiments using the NS3 network simulator [RH10] to replicate the scenarios mentioned in Sec. 5.3.3. To evaluate a link, we borrowed the concept of moving windows from OB-EWMA [MG21] to restrict the transmission history. For instance, a PDR slot size $d_{ps} = 10s$ and a moving window size $s_{mw} = 5$ results in a window with transmission history on the link for the last 50 s. In the experiments, we evaluate the packet delivery ratio (PDR) and the statistical properties of the link using metrics discussed in Chapter 4, $\mu$-value and $z$-value from the Runs-test, during each moving window.

In our evaluation, we use the network topology depicted in Fig. 5.8. In this setup, it is assumed that the medium is only used by the nodes 1, 2, 3, 4, 5, and 6, and all links are 100% reliable unless frames collide. Further, the nodes are synchronised,

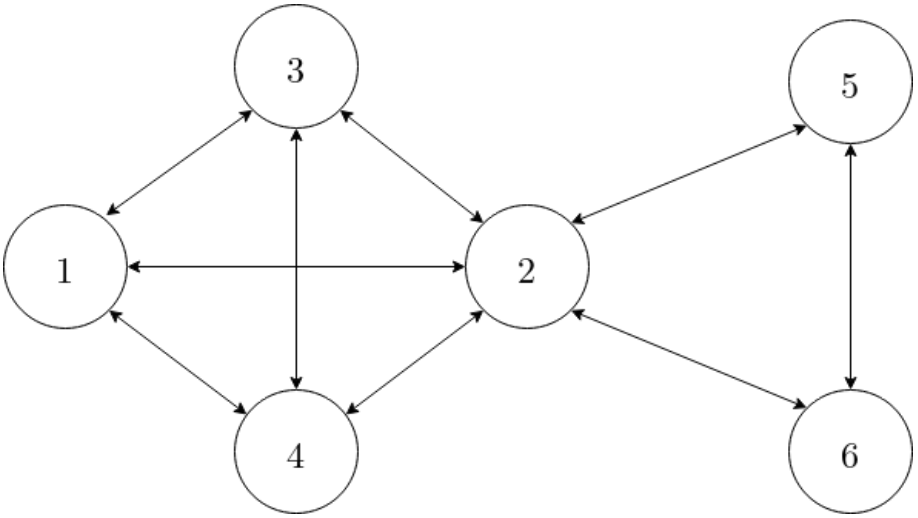



Figure 5.8: Topology used in the experiments

i.e., they start simultaneously and use the same virtual clock of NS3. In our analysis, we evaluate link $1 \rightarrow 2$. In this topology, nodes 5 and 6 are hidden stations to node 1. The applications on the nodes transmit broadcast frames, and thus, there are no mechanisms to avoid the influence of hidden stations.

An aperiodic or periodic application (or applications) runs on each node during an experiment. However, the frames generated by applications on the nodes have the same payload size and are transmitted at the same rate. A frame contains a payload of size $s_{payload} = 52B$ and is transmitted at a rate $r_{payload} = 11Mbps$. The IEEE headers are also transmitted at a rate $r_{header} = 11Mbps$ since all nodes in the network can decode it. The remaining parameters ($MLO$ and $PLO$) are 34 Bytes and 289 µs, respectively. Thus, by applying Eqn. 5.6, the transmission time required is

$$txTime_{f,max} = \frac{52B}{11Mbps} + \frac{34B}{11Mbps} + 289\,\mu s \approx 351\,\mu s$$

#### 5.4.2.1 Nodes with the same assigned medium occupancy ratio

In this set of experiments, we evaluate the scenario where two or more nodes have the same medium occupancy requirements, or the requirements result in a MML configuration that increases the collision probability of the network. We set up the applications on the nodes to create a scenario where the requirements result in a MML configuration to show the advantages of *dynamic refilling*.

Tab. 5.8 shows the application characteristics of an application running on each node. For instance, node 1 has a periodic application that generates new information every 14 ms. As mentioned, every application sends a frame with the same payload

| **Application Settings** | | | |
|---|---|---|---|
| Node ID | Application start offset | Application characteristics | $mor^{v}_{required}$ |
| 1 | 0 | Periodic - 14 ms | 2.507% |
| 3 | 0 | Aperiodic - random interval between 25 - 30 ms | 1.276% |
| 4 | 10 ms | Periodic - 46 ms | 0.76% |
| 5 | 0 | Aperiodic - random interval between 25 - 30 ms | 1.276% |
| 6 | 10 ms | Periodic - 70 ms | 0.501% |

Table 5.8: Application settings for experiments B.I and B.II

size and requires a transmission time $txTime_{f,max} = 351\,\mu s$. I.e., node 1 transmits every 14 ms a frame that requires a transmission time of 351 µs. So the required medium occupancy ratio to support this application is

$$mor^{1}_{requied} = \frac{351\,\mu s}{14\,ms} = 2.507\%$$

Similarly, nodes 4 and 6 have periodic applications that generate a frame every 46 ms and 70 ms, respectively. However, the applications start 10 ms from the start of the node (column *Application start offset*). For instance, at node 4, the application generates the first frame after 56 ms from the start of the node, and then periodically generates the frame. The required medium occupancy ratio for nodes 4 and 6 are stated in the last column.

Aperiodic applications run on nodes 3 and 5, where frames are created with a minimum and maximum interval of 25 ms and 30 ms, respectively. The transmission intervals are randomly selected from a uniform distribution (between 25 and 30 ms). Thus, on average, a frame is created every 27.5 ms, which requires a transmission time $txTime_{f,max} = 351\,\mu s$. Therefore, the required medium occupancy ratio to support this application on node 3 is

$$mor^{3}_{requied} = \frac{351\,\mu s}{27.5\,ms} = 1.276\%$$

Further, no application is running on node 2. In this set of experiments, the simulations are run for 30 minutes. For analysis, we used a PDR slot size $d_{ps} = 10s$ and a moving window size $s_{mw} = 5$ . It results in a window with transmission history for the last 50s. Thus, in each moving window, at least 3500 transmission events are recorded on link 1→2 (one transmission every 14 ms).

**Experiment B.I:** Replicating Scenario II.

| MML Settings | | | | | | |
|---|---|---|---|---|---|---|
| Node ID | $mor^{v}_{required}$ | $mor^{v}_{assgn}$ | $d^{v}_{token}$ (Static) | $d^{v}_{txMax}$ | $d^{v}_{minRefill}$ (Dynamic) | $d^{v}_{maxRefill}$ (Dynamic) |
| 1 | 2.507% | 2.5% | 50µs | 360µs | 50µs | 2500µs |
| 3 | 1.276% | 1.25% | | | | |
| 4 | 0.76% | 0.75% | | | | |
| 5 | 1.276% | 1.25% | | | | |
| 6 | 0.501% | 0.50% | | | | |

Table 5.9: MML settings for experiment B.I

In this set of simulations, we assigned a medium occupancy ratio slightly below the node requirement to replicate Scenario II. Thus, when a frame arrives in the MML-arrival queue, there is insufficient transmission time available in the token bucket. Tab. 5.9 shows the MML settings for the experiment. Further, these settings replicate a scenario that increases the collision probability of the network due to the relationship between the calculated refill intervals on each node.

Node 1 requires a medium occupancy ratio $mor^{1}_{required} = 2.507\%$, and is assigned a medium occupancy ratio $mor^{1}_{assgn} = 2.5\%$. A token size $d^{v}_{token} = 50\,\mu s$ is used for the simulation using *static refilling*. The token refill interval $d^{1}_{refill}$ to provide a medium occupancy ratio $mor^{1}_{assgn} = 2.5\%$ to node 1 is

$$d^{1}_{refill} = \frac{50\,\mu s}{2.5\%} = 2000\,\mu s \qquad \text{applying Eqn. 5.3}$$

Similarly, applying Eqn. 5.3, the token refill intervals to meet the assigned medium occupancy ratio for nodes 3, 4, 5, and 6 are,

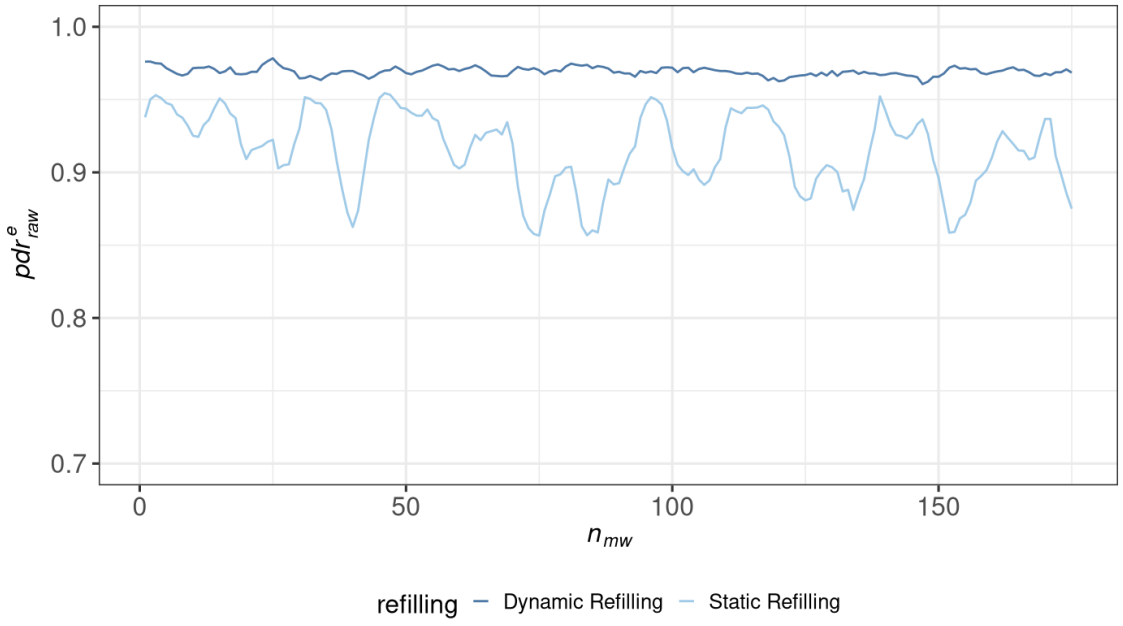
$$d^{3}_{refill} = \frac{50\,\mu s}{1.25\%} = 4000\,\mu s$$

$$d^{4}_{refill} = \frac{50\,\mu s}{0.75\%} = 6667\,\mu s$$

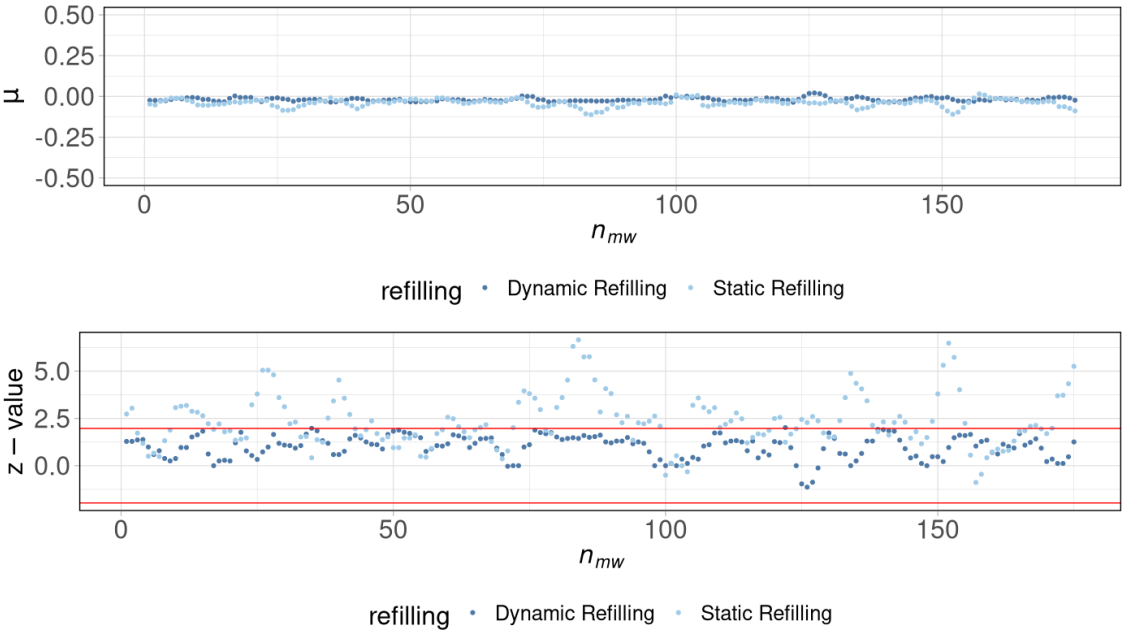
$$d^{5}_{refill} = \frac{50\,\mu s}{1.25\%} = 4000\,\mu s$$

$$d^{6}_{refill} = \frac{50\,\mu s}{0.5\%} = 10000\,\mu s$$

The Highest Common Factor (HCF) of the token refill intervals among nodes 1, 3, 5, and 6 is 2000 µs, i.e., the token refill interval of node 1. It could potentially




(a) Observed raw PDR during the windows




(b) Statistical properties of the link. The red line denotes $z$-value boundary to differentiate between random and non-random Bernoulli event outcomes.

Figure 5.9: Simulation results of link $1 \rightarrow 2$ from experiment B.I

lead to a situation where node 1 and node 3, 5, or 6 refill a token at (almost) the same point. Whereas while using *dynamic refilling*, the next token refill interval varies. The minimum and maximum refill intervals are set as $d^{v}_{minRefill} = 50\,\mu s$ and $d^{v}_{maxRefill} = 2500\,\mu s$. As mentioned earlier, we evaluate link 1→2, where nodes 5 and 6 are hidden stations to node 1. Further, there are no mechanisms to mitigate the influence of hidden stations.

Fig. 5.9 shows the results of the simulations. Fig. 5.9a shows the observed PDR during each moving window. Compared to *static refilling*, using *dynamic refilling* reduces the collisions on the link and improves the link reliability. Fig. 5.9b shows the $\mu$-value and $z$-value from the runs test. The results show that using *dynamic refilling* improves the statistical properties of the link required for link boosting.

In contrast, using *static refilling*, often the $z$-values cross to the oscillatory region, but in a tolerable margin. Recalling the conclusion in Chapter 4, a $\mid z \mid > 1.97$ denotes that the evaluated Bernoulli sequence is non-random. A $z$-value higher than 1.97 shows evidence for an oscillatory sequence and below for a bursty sequence (see Chapter 4). Further, the $\mu$-values during the moving windows also imply the same conclusion. In comparison with *static refilling*, using *dynamic refilling* shows better results.

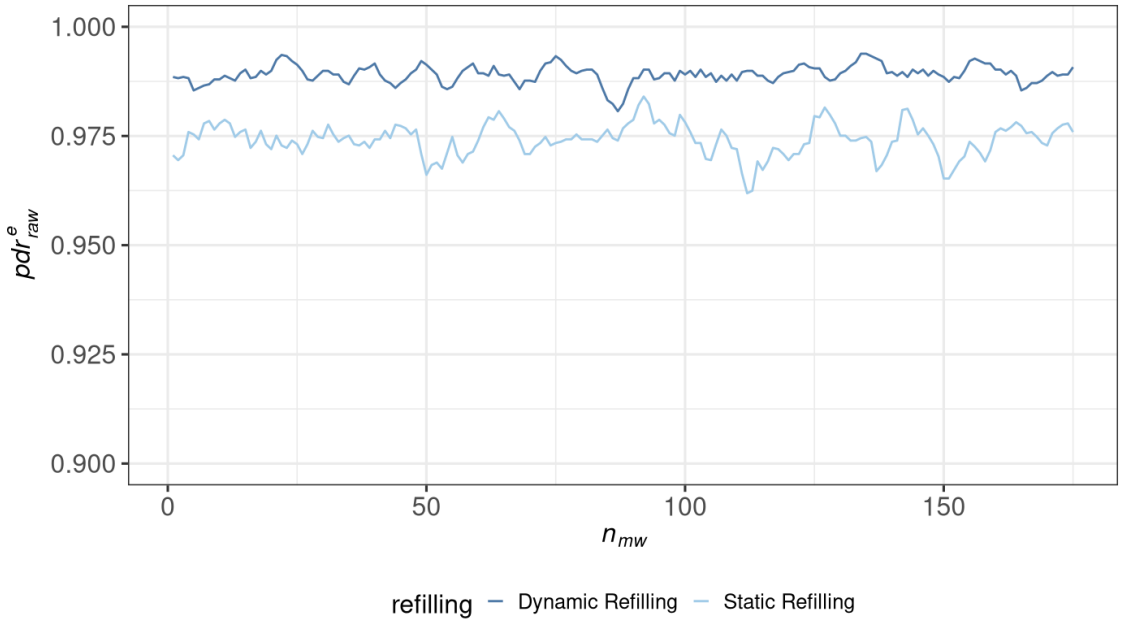
**Experiment B.II:** Assigning the required amount of medium occupancy ratio to the nodes, i.e., $mor^{v}_{assgn} = mor^{v}_{required}$

Next, we simulate the scenario when the exact amount of required medium occupancy ratio is assigned to the nodes. Thus, when a frame arrives in the MML-arrival queue, either there is enough transmission time available in the bucket or not (Scenario II, or III). The application settings (see Tab. 5.8) are the same as in the previous experiment. Apart from the assigned medium occupancy ratio, the MML settings (see Tab. 5.9) are also the same.

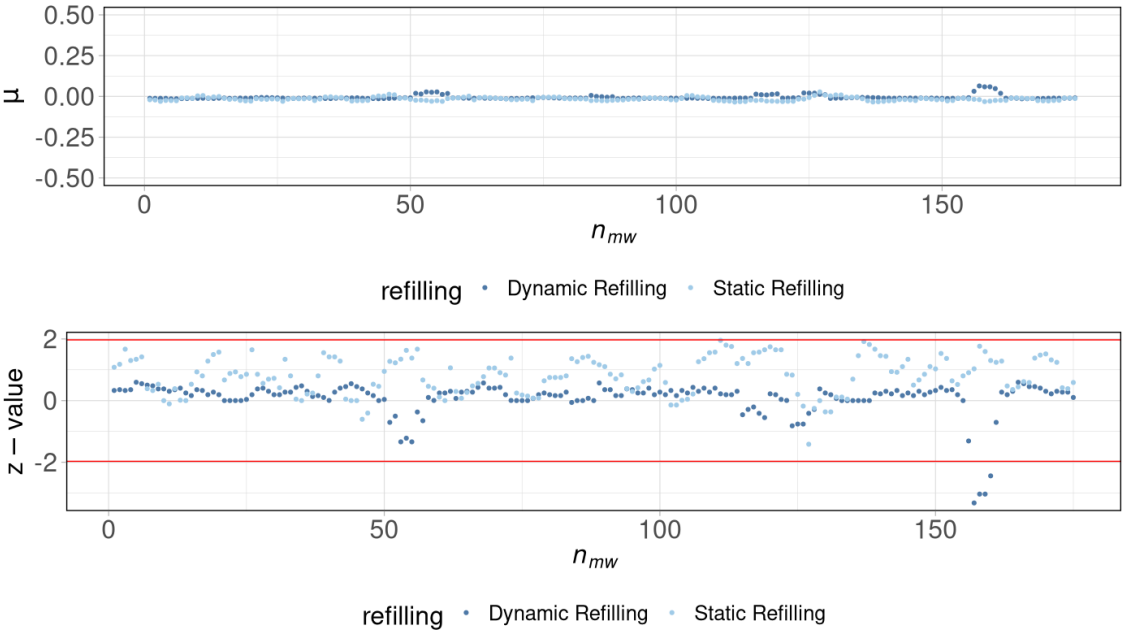
Fig. 5.10 shows the results of the simulations. Fig. 5.10a and Fig. 5.10b show the observed PDR and statistical properties of the link during each moving window, respectively. Compared to the previous experiment, the link reliabilities improved for *static refilling* and *dynamic refilling*, especially using the former. The refilling interval dependency for *static refilling* is weaker in this experiment, since the HCF of refill interval used at nodes 1, 3, 4, 5, and 6 is not the refill interval of node 1. Applying Eqn. 5.3

$$d^{1}_{refill} = \frac{50\,\mu s}{2.507\%} = 1994\,\mu s$$

$$d^{3}_{refill} = \frac{50\,\mu s}{1.276\%} = 3909\,\mu s$$




(a) Observed raw PDR during the windows




(b) Statistical properties of the link. The red line denotes $z$-value boundary to differentiate between random and non-random Bernoulli event outcomes.

Figure 5.10: Simulation results of link $1 \rightarrow 2$ from experiment B.II

$$d^4_{refill} = \frac{50\,\mu s}{0.76\%} = 6579\,\mu s$$

$$d^5_{refill} = \frac{50\,\mu s}{1.276\%} = 3909\,\mu s$$

$$d^6_{refill} = \frac{50\,\mu s}{0.501\%} = 9980\,\mu s$$

The HCF of the refill intervals of nodes 1, 3, 4, 5, and 6 is 1. Thus, the link reliability (of the link 1→2) is less affected by the refill intervals of the nodes. It is still possible that all nodes refill a token at (almost) the same point (i.e., LCM of those refill intervals). Further, in both cases, the link holds the statistical properties required for successful link boosting, i.e., observed a $\mu$-value close to 0 and a $\mid z \mid \leq 1.96$ during the moving windows.

#### 5.4.2.2 A network with weak statistical properties for link boosting

These experiments evaluate the role of traffic shaping using the TTBA to remove the influences of application characteristics on the statistical link properties, and further improvements with revised version of TTBA using *dynamic refilling*. To begin with, we illustrate the drawback of static refilling during Scenario III, i.e., when a frame arrives in the frame-arrival queue, there is enough transmission time available in the time token bucket. Thus, in this scenario, the transmission trigger points depend on the applications running on the node. Then, we illustrate improvements in statistical link properties using traffic shaping, especially with *dynamic refilling*.

| **Application Settings** | | | |
|---|---|---|---|
| Node ID | Application start offset | Application characteristics | $mor^v_{required}$ |
| 1 | 0 | Periodic - 10 ms | 3.51% |
| 3 | 0 | Aperiodic - random interval between 100 - 140 ms | 0.585% |
| | 10 ms | Periodic - 130 ms | |
| 4 | 10 ms | Periodic - 120 ms | 0.293 % |
| 5 | 50 ms | Periodic - 130 ms | 0.27% |
| 6 | 10 ms | Aperiodic - random interval between 90 - 140 ms | 0.235% |
| | 20 ms | Periodic - 150 ms | |

Table 5.10: Application settings for experiments B.III and B.IV

Tab. 5.10 shows the applications running on each node during both experiments. The applications are configured to cause statistically dependent frame losses on link 1→2, if no traffic shaping is applied. The application running on node 1 transmits a frame periodically every 10 ms. On node 3, two applications are running, an aperiodic application and a periodic application. The former transmits a frame with an interval randomly selected from a uniform distribution between 100 ms and 140 ms. The latter transmits a frame every 130 ms. The periodic application starts 10 ms from the start of the node. Thus, the first frame (from the periodic application) is transmitted at 140 ms (*application start offset* of 10 ms) from the start of the node.

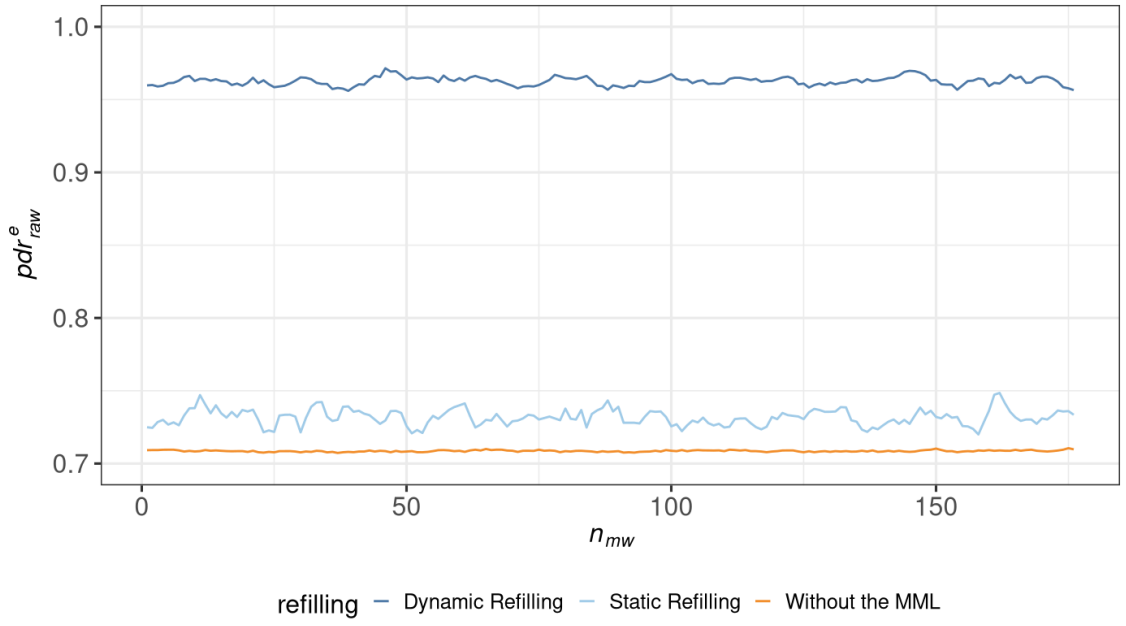
Nodes 4 and 5 have a periodic application that transmits a frame periodically every 120 ms and 130 ms, respectively. Node 4 sends the first frame at 130 ms (*application start offset* of 10 ms), and similarly, node 5 sends the first frame 180 ms (*application start offset* of 50 ms) from the start of the node. These frames generated by the periodic applications running on nodes 3, 4, 5, and 6 transmit at the same (or almost) time as the frame sent from node 1 (the HCF of their periods is 10 ms). Hence, the transmission event outcomes on link 1→2 depends on these applications. In our evaluation, we used a PDR slot size $d_{ps} = 10s$ and a moving window size $s_{mw} = 4$. It results in a window with transmission history for the last 40s. Further, in each moving window, at least 4000 transmission events are recorded on link 1→2 (one transmission every 10 ms).

We conduct two different experiments. Experiment B.III replicate Scenario III, and experiment B.IV evaluate the functioning of TTBA when assigning the exact amount of required medium occupancy ratio to the nodes. If necessary traffic shaping is not performed in both experiments, the application characteristics influence the statistical link properties.

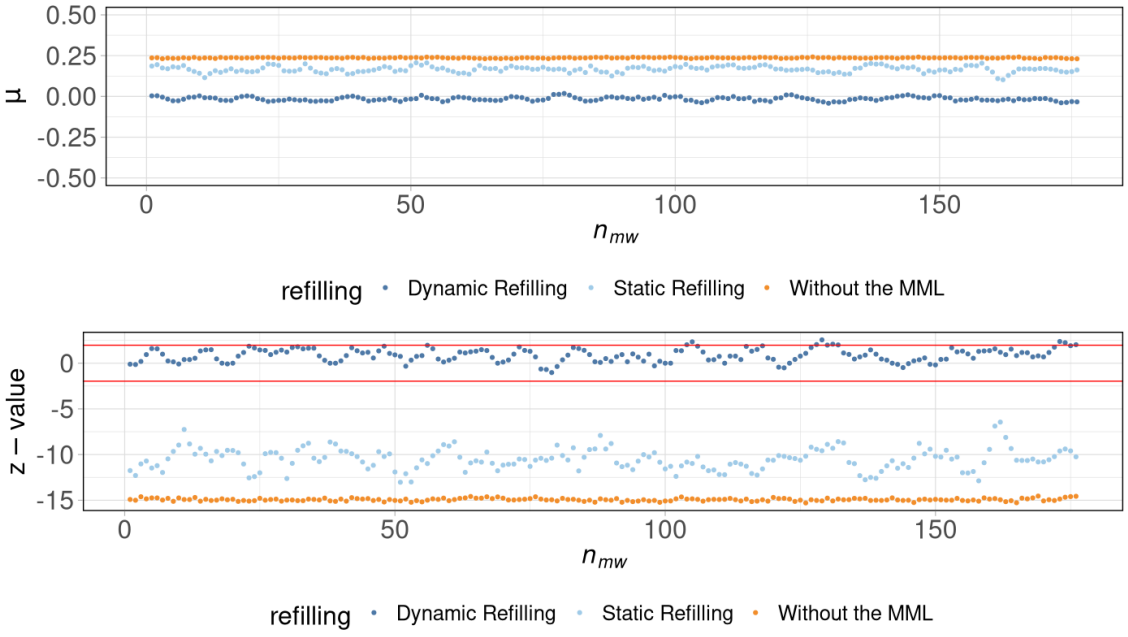
**Experiment B.III:** Replicating Scenario III

In this set of simulations, we illustrate the application's dependency on the transmission outcomes on link 1→2, and the improvements while using TTBA, notably with dynamic refilling during Scenario III. We conduct three simulations,

- without the MML to show the outcomes without traffic shaping,
- with *static refilling*, and
- with *dynamic refilling*.




(a) Observed raw PDR during the windows




(b) Statistical properties of the link. The red line denotes $z$-value boundary to differentiate between random and non-random Bernoulli event outcomes.

Figure 5.11: Simulation results of link $1 \rightarrow 2$ from experiment B.III

| **MML Settings** | | | | | | |
|---|---|---|---|---|---|---|
| Node ID | $mor^{v}_{required}$ | $mor^{v}_{assgn}$ | $d^{v}_{token}$ (Static) | $d^{v}_{txMax}$ | $d^{v}_{minRefill}$ (Dynamic) | $d^{v}_{maxRefill}$ (Dynamic) |
| 1 | 3.51% | 3.7% | 50µs | 360µs | 50µs | 1000µs |
| 3 | 0.59% | 0.6% | | | | |
| 4 | 0.293% | 0.31% | | | | |
| 5 | 0.27% | 0.28% | | | | |
| 6 | 0.24% | 0.25% | | | | |

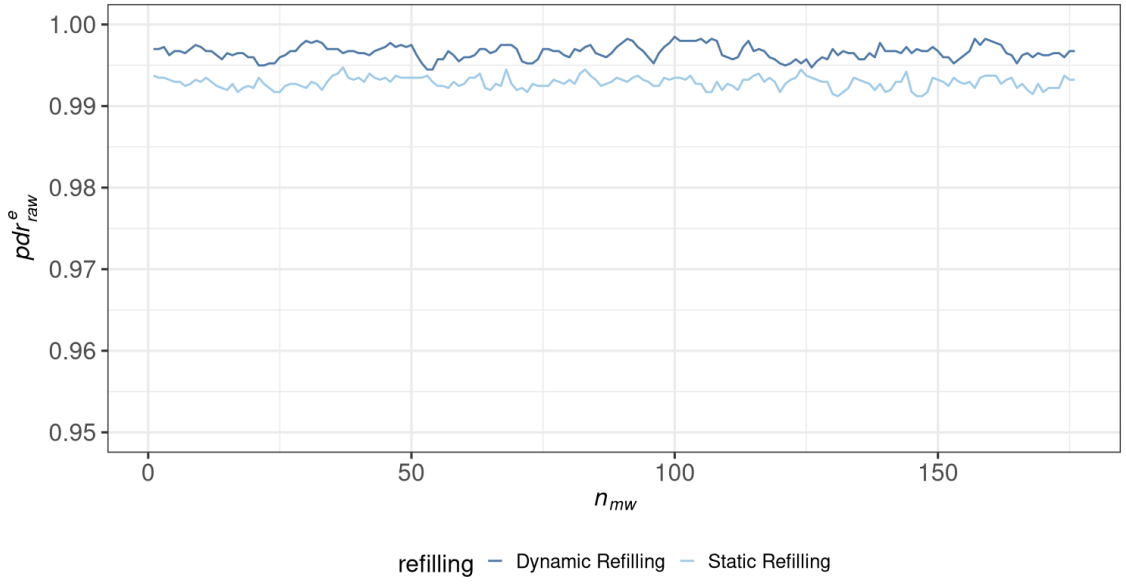
Table 5.11: MML settings for experiments B.III and B.IV

Tab. 5.11 shows the MML settings for the simulations. To imitate Scenario III, we assigned to the nodes a medium occupancy ratio slightly higher than the required. Thus, when a frame arrives in the MML-arrival queue, enough transmission time $d^{v}_{txAvail}$ is available to transmit the frame [18]. For instance, on node 3, the required transmission ratio $mor^{3}_{required}$ is 0.59%, we assigned a medium occupancy ratio $mor^{3}_{assgn}$ of 0.6%. Further, *static refilling* uses a token size of 50 µs, and for *dynamic refilling*, a minimum and maximum interval $d^{v}_{minRefill} = 50\,\mu s$ and $d^{v}_{maxRefill} = 1000\,\mu s$ are used. The remaining MML settings are provided in the table.

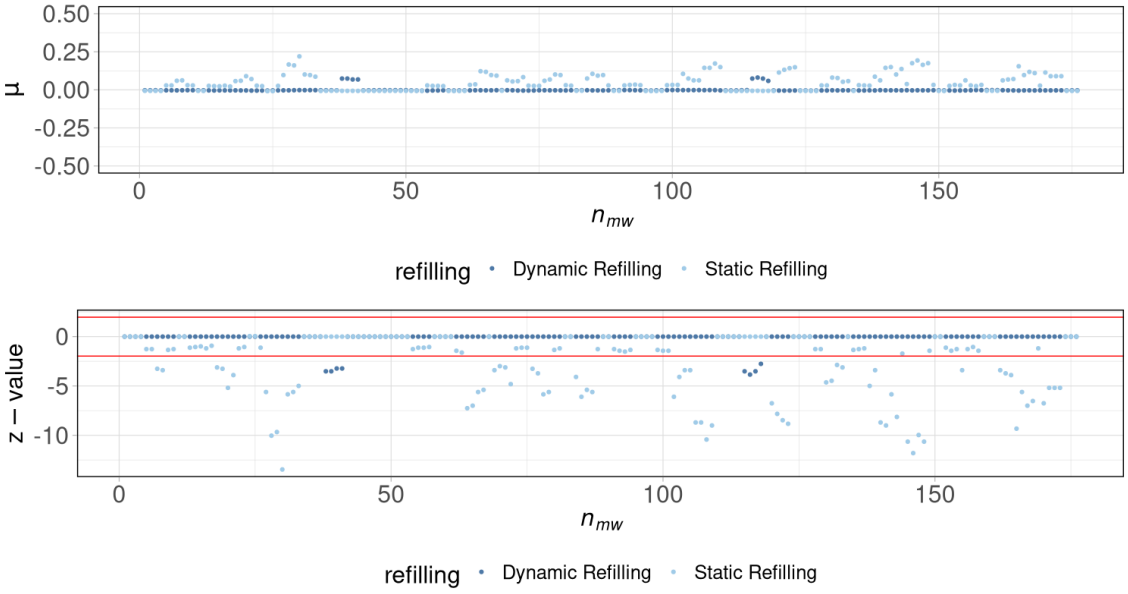
Fig. 5.11 shows the results of the simulations. Without MML, the transmission outcomes or frame losses on link 1→2 are bursty or statistically dependent. Fig. 5.11a shows the link reliability during each simulation. Fig. 5.11b shows the $\mu$-value and $z$-value from the runs test. Without the MML, the link reported a $z$-value higher than -15, which shows that the event outcomes on each moving window are non-random. Similarly, a $\mu$-value of 0.25 also concludes the statistically dependent or bursty event outcomes on the link.

Using the MML and with *static refilling*, the link reliability and the statistical link properties of link 1→2 did not improve substantially. The link reported a $z$-value between -7.5 and -12.5, showing evidence of non-random event outcomes. Similarly, the $\mu$-value indicating dependent event outcomes. However, with *dynamic refilling*, when a frame arrives in the MML-arrival queue, it waits until the arrival of next refill token. This helps to mitigate the dependency of applications (running on other nodes) on the transmission outcomes on link 1→2. In contrast, dynamic refilling improved the link reliability significantly (see Fig. 5.11a) and the statistical link properties required for successful link boosting (see Fig. 5.11b).

[18] A token contains the transmission time assigned to the node. When a frame arrives, it consumes the transmission time, not a token or tokens. Thus, the frame may consume the whole transmission time available in the bucket or a portion.




(a) Observed raw PDR during the windows




(b) Statistical properties of the link. The red line denotes $z$-value boundary to differentiate between random and non-random Bernoulli event outcomes.

Figure 5.12: Simulation results of link $1 \rightarrow 2$ from experiment B.IV

**Experiment B.IV:** Assigning the required amount of medium occupancy ratio to the nodes, i.e., $mor^{v}_{assgn} = mor^{v}_{required}$

Next, we evaluate link $1 \rightarrow 2$ by assigning the required medium occupancy ratio $mor^{v}_{required}$ to the nodes to support the applications stated in Tab. 5.10. Apart from the assigned medium occupancy ratio to the nodes, the remaining MML settings (see Tab. 5.11) are the same. Thus, when a frame arrives in the MML-arrival queue, either there is enough transmission time available in the bucket, or it waits until enough tokens are refilled.

Fig. 5.12 shows the results of the simulation. Fig. 5.12a and Fig. 5.12b show the link reliability and statistical properties of the link during each moving window. Results show that the *static refilling* and the *dynamic refilling* mitigate the dependency on the transmission outcomes due to the properties of the applications running nodes. In contrast to *static refilling*, using *dynamic refilling* improves the link reliability and statistical properties of the link.

Compared to experiment B.II, in this experiment, the nodes have different assigned medium occupancy ratios. Hence, in this experiment, when the node uses *static refilling* for TTBA, the refill intervals at each node differs. This influence is depicted in the link reliability of link $1 \rightarrow 2$, i.e., when comparing the observed raw PDR in Fig. 5.10a and Fig. 5.12a. It reduces the probability of two or more nodes transmitting at the same (or almost) time.

#### 5.4.2.3 Conclusion

Experiment B.I shows that when two or more nodes in the network have the same medium occupancy requirement, *static refilling* might compute a similar refill interval that increases the collision probability of the network. Experiment results show that *dynamic refilling* mitigates such a scenario, where the subsequent token refill intervals vary with *dynamic refilling*.

The simulation results from experiment B.IV show that the traffic shaping using the static and dynamic refilling mitigate the application's dependency on transmission outcomes, unless for Scenario III. I.e., with *static refilling*, the MML cannot influence the transmission trigger points. It completely depends on the applications. However, with the revised version of TTBA using *dynamic refilling*, frames wait until the arrival of the next token, which to helps mitigate the dependency. To summarize, with *dynamic refilling*, the link showed a higher PDR and stronger statistical link properties than *static refilling* for evaluated scenarios.

## 5.5 Algorithms and protocols using the functionalities of the MML

Next, we briefly show how certain algorithms and protocols use the functionalities of the MML for their operation.

### 5.5.1 TTBA for Cooperative Fair Medium Sharing

In [KMG19b], an algorithm for cooperative medium occupancy scaling among nodes using TTBA is introduced. It dynamically reacts to the changes of the cooperative nodes and uses the metric $mor^{v}_{wastedUnUsableRatio}$ to detect medium overload. For instance, node mobility, changes in the traffic pattern of external traffic, or the introduction of new nodes may affect the cooperative nodes.

For cooperative scaling, the algorithm defines the requirements (for the medium share) as the *base* and *preferred* requirements. The base requirement of a node is the required amount of medium occupancy ratio to satisfy the base requirements of applications running on it. Similarly, preferred to satisfy the preferred requirements of applications. The scaling happens between the base and preferred requirements, i.e., the assigned medium occupancy ratio $mor^{v}_{assgn}$ to a node $v$ varies depending on the scaling algorithm. Scaling is conducted depending on specific rules and the fairness index, *QoSFairnessIndex*, and rules depend on the recent medium conditions. The algorithm uses the metric $mor^{v}_{wastedUnUsableRatio}$ to monitor the medium. For instance, scaling up or increasing the assigned medium occupancy ratio could lead to medium overload, and the algorithm uses the metric $mor^{v}_{wastedUnUsableRatio}$ to monitor such situations and react accordingly. Or, a new cooperative node or an increase in the traffic generated by the external networks could lead to medium overload, that may lead to a situation where cooperative nodes are instructed to scale down.

Notably, when scaling up, the algorithm allocates an additional varying share of the medium than the base requirement. This additional medium share allocation or scaling may lead to Scenario III mentioned earlier (see Sec. 5.3.3), i.e., when a frame arrives in MML-arrival queue, there is always enough transmission time accumulated in the token bucket to transmit the frame. As discussed earlier (see experiments B.III), using *static refilling* may affect the network and link quality. As shown earlier, the revised version of TTBA using *dynamic refilling* introduced in this chapter helps to mitigate this issue.

### 5.5.2 Route selection based on the medium conditions

In [KG22], the authors introduced $dR_{min}$-routing, a decentralized routing protocol that uses $r_{min}$-routing, [KMG19a] for initial route candidate selection. It considers the information by the Reservation Module of the MML for route selection. While calculating the route between two nodes, $dR_{min}$-routing considers the medium requirements to the intermediate nodes in a potential route. If the MML cannot reserve the necessary medium occupancy ratio to service a new route through the node, $dR_{min}$-routing considers a different route.

$dR_{min}$-routing uses the MML to monitor unused reservations. When a new route is calculated, it requests MML to reserve a stream and the required medium occupancy ratio to service the route. Further, $dR_{min}$-routing instructs the Reservation Module to release a stream reservation if it is unused for the last $t_{release}$ period. For instance, with a configuration $t_{release} = 10$s, if a stream is unused for the last 10s, the MML releases the reservation and notifies the routing protocol.

## 5.6 Conclusion

This chapter discussed the Time Token Bucket Algorithm (TTBA) used by the Medium Management Layer (MML) of the WiPS for its operation. TTBA is a hardware-independent approach for traffic shaping and monitoring. The traffic shaping is conducted by controlling the time token bucket parameters such as the token refilling interval $d^v_{refill}$, token size $d^v_{token}$, and bucket size $d^v_{txMax}$. The metric $mor^v_{wastedUnUsableRatio}$ calculated from unusable wasted tokens is proposed to monitor medium overload. Following that, a new approach for refilling tokens to the token bucket that may potentially improve the link reliabilities, and reduce the influence of applications on statistical link properties is introduced.

Real-world and simulation experiments are conducted to evaluate the applicability of the TTBA. In Sec. 5.4.1, the efficacy of the $mor^v_{wastedUnUsableRatio}$ to detect medium overload locally is studied for multiple traffic scenarios. The results show that our hardware-independent metric $mor^v_{wastedUnUsableRatio}$ can identify medium overload. In the experiments, the assigned medium occupancy ratio to the internal network $mor^G_{assgn}$ was equally split among the internal nodes. The experiment results in the presence of external traffic show that increasing the network load affected certain nodes (a higher value of $mor^v_{wastedUnUsableRatio}$) depending on the channel and the topology. It shows that even though the testbed nodes are in single-hop distance, the number of nodes an internal node competes to access the medium may vary depending on the external networks. Our metric $mor^v_{wastedUnUsableRatio}$ monitors the

network overload observed by each node locally and helps to identify the amount of traffic a node can handle without overloading the medium. Further, in the absence of external traffic, in an overloaded medium, the nodes were affected depending on their random backoff and contention window.

Next, improvements with *dynamic refilling*, the revised token refilling mechanism for TTBA is evaluated. In our experiments (see Sec. 5.4.2), we considered specific scenarios that could potentially affect the link reliabilities and statistical link properties, and depicted the improvements using *dynamic refilling*. The experiments show that the characteristics of applications running on the nodes can cause dependent event outcomes. Without necessary measures, it may affect the operation of protocols (for instance $r_{min}$-routing) relying on those properties. Even with adequate measures, it is still possible that statistical link properties are affected by factors beyond the control of the internal network, for instance, due to the influence of external devices or networks. Finally, we briefly discussed how certain algorithms and protocols use the functionalities of MML for their operations.

# 6

# OB-EWMA: A Link Metric for Reliability-constrained Routing in Wireless Networks

## Contents


**6.1 Introduction** . . . . . **168**
**6.2 State of the art** . . . . . **168**
**6.3 Outlier Bounded Exponential Weighted Moving Average (OB-EWMA)** . . . . . **170**
6.3.1 Motivation . . . . . 170
6.3.2 Definition of OB-EWMA . . . . . 171
**6.4 Link metrics used in evaluation** . . . . . **175**
6.4.1 Window Mean with EWMA (WMEWMA) . . . . . 175
6.4.2 FaLQE . . . . . 176
6.4.3 F-LQE . . . . . 177
6.4.4 *min*-EWMA . . . . . 179
**6.5 Experimental Evaluation** . . . . . **179**
6.5.1 Properties of PDR distribution . . . . . 180
6.5.2 Metric Evaluation . . . . . 182
6.5.3 Routing Experiments . . . . . 189
**6.6 Conclusion** . . . . . **193**

## 6.1 Introduction

Wireless technologies are now utilized for applications such as Wireless Networked Control Systems (WNCS) that require end-to-end communication with a specified Quality-of-Service (QoS). QoS requirements may be specified concerning different quality metrics, for example, latency, bandwidth, energy, or reliability, and suitable QoS routes are discovered to implement the specified QoS requirements. For the discovery and successful operation of QoS routes that guarantee minimum end-to-end reliability, the network status information must be sufficiently accurate and resilient. In practice, link conditions of a wireless network may change due to varying channel quality, node movements, interference from external nodes, and dynamic traffic profiles. Thus, it is necessary to have a link metric that is adaptive to reflect these changes in the network while also being conservative to avoid frequent violations of QoS requirements.

This chapter presents a link metric developed by considering the requirements for reliability-constrained routing in contention-based wireless networks. It is structured as follows: Sec. 6.2 discusses the state-of-the-art link metrics developed to calculate link quality. Following that, in Sec. 6.3, the proposed link metric for reliability-constrained routing in contention-based wireless networks is introduced. Sec. 6.4 discusses the link metrics used in the evaluation to compare the performance of the proposed metric. In Sec. 6.5, we discuss the experimental results, and finally, in Sec. 6.6, we conclude the findings.

## 6.2 State of the art

There are multiple state-of-the-art link metrics to estimate the quality of links in a wireless network. In [Bil13], the author categorizes and compares different link metrics based on their design objectives, data gathering approaches, and analysis methods. Here, we primarily focus on link quality metrics addressing link reliability. i.e., the probability for a successful transmission on the link. In [Vla+08], the authors compare various raw link quality metrics such as Received Signal Strength Indication (RSSI), Signal-to-Interference-plus-Noise Ratio (SINR), Packet Delivery Ratio PDR, and Bit-Error Rate (BER). Their study shows that each metric only shows specific aspects of the link. Further, retrieving the SINR and BER of a received IEEE 802.11 frame by off-the-shelf commodity hardware requires modifications to the driver module.

Most link metrics are derived from the observed PDR of a link since calculating PDR of a link from historical data is straightforward. A class of link quality metrics is developed by applying the Exponential Weighted Moving Average (EWMA) statistical technique to the observed PDR values. Small memory footprint, exponentially decreasing weighting factors for historical values, flexibility, and simplicity make EWMA attractive for several applications. The authors compare the metrics Moving average and the metrics evolved from EWMA such as WMEWMA (Window Mean with Exponential Moving Average), TWMA (Time Weighted Moving Average), and Flip-Flop EWMA in [WC03]. Their study exemplifies that WMEWMA outperforms other metrics in terms of reactivity (reacting to the changes in observed PDR) and stability (low volatility). However, the estimation error (i.e., overestimating the link quality) by the metrics mentioned depends on the variance of the observed PDR. In [LXL19], the authors introduced FaLQE, a link quality metric that uses a dynamic history weight for EWMA based on the degree of fluctuation (variance) to the observed PDR values. However, FaLQE reacts immediately to any changes in the observed PDR, even though it is an outlier (we will discuss further on outliers in later sections).

In [Bac+10], the authors present F-LQE, a link quality metric, using fuzzy logic to estimate link quality by combining several raw metrics. I.e., F-LQE considers PDR, link asymmetry, stability, and channel quality information provided by the hardware for its calculation. Depending on the raw values and mapping functions defined, F-LQE focuses on accurately estimating the link quality and classifies the links into different quality levels. In [Bac+10], the performance of F-LQE is compared with other link quality metrics such as RNP[Cer+05], ETX [De +03], 4-Bit [Fon+07], and WMEWMA [WC03]. Results show that F-LQE outperforms others regarding temporal behaviour and estimation accuracy. However, the mapping functions defined by the authors for the calculation of F-LQE may lead to overestimation in certain cases. For instance, a 'stable' 'symmetric' link with observed PDR greater than 0.95 is mapped or estimated to link quality of 1. Further, by design F-LQE considers different aspects of the link, which may lead to frequent recalculation of the link quality. Moreover, F-LQE, RNP, and 4-Bit measure bi-directional link quality, whereas we are interested in uni-directional link quality.

## 6.3 Outlier Bounded Exponential Weighted Moving Average (OB-EWMA)

In the section, we present the link metric OB-EWMA [MG21]. We begin by presenting the motivation behind the development of the metric in Sec. 6.3.1. Then, in Sec. 6.3.2, we present the link metric calculation.

### 6.3.1 Motivation

Chapter 2 introduced $r_{min}$-routing, which discovers and operates bandwidth-efficient $r_{min}$-routes with a specified minimal statistical success ratio for packet delivery. For instance, for a given value $r_{min} = 0.9$, a route with a minimal statistical success ratio for end-to-end packet delivery of at least 90% is discovered. $r_{min}$-routing builds on a network model $G$, where $V$ is the set of nodes and $q : V \times V \rightarrow \mathbb{R}_{[0,1]}$ is a statistical link quality function for single-hop communication. From $q$, the set of asymmetrical links $E =_{def} \{(v, v') \in V \times V \mid q(v, v') > 0\}$ can be derived. To achieve $r_{min}$, the statistical reliability of individual links is improved by well-directed retransmissions applied during the operation of routes. For this, the routing algorithm needs the information regarding the existence of links $e = (v, v')$, and their quality, $q(v, v')$ which may denote the probability for successful single-hop delivery of a frame from node $v$ to $v'$. This directs to the question of defining the link quality $q^e$ for a potential link $e$.

A direct approach would be to define $q^e$ as the Packet Delivery Ratio (PDR) $pdr^e_{raw}$ of link $e$, i.e., the number of successfully received packets divided by the number of transmitted packets. However, in the context of wireless networks with contention-based medium access, this link metric is neither sufficiently accurate nor resilient, which would lead to frequent violations of target end-to-end route reliability $r_{min}$ throughout route operation. We developed OB-EWMA (Outlier Bounded Exponential Weighted Moving Average) [MG21], a link metric devised for reliability-constrained routing in wireless networks, to overcome this problem. OB-EWMA is designed to satisfy the following properties:

1. **Based on Packet Delivery Ratio** (PDR): The link metric should incorporate PDR since the route reliability requirements are expressed as a minimum success ratio.

2. **Adaptive**: The link metric should reflect the dynamic changes of PDR to account for varying link quality. I.e., the metric should adapt the changes to PDR over the link during the network operation.

3. **Stability**: Over time, the link quality variation should be small to keep existing routes stable.

4. **Conservative but bounded**: The link metric should tolerate only a small ratio of downward PDR outliers to avoid frequent route failures.

As discussed in Sec. 6.2, there are various link metrics, and each of them is designed for specific use cases. OB-EWMA returns an adaptive, stable, and conservative link quality by tolerating a small configurable ratio of downward PDR outliers. I.e., OB-EWMA estimates a conservative value, which reduces the likelihood of overestimating the link quality rather than the actual, and at the same time, being resistant to outliers.

An estimated link quality by a link metric is considered as *overestimated* if the estimated link quality is higher than the actual. Contrary, an estimate is considered as *underestimated* if the estimate is smaller than the actual. If the link metric often overestimates the link quality, then it may affect the higher protocols that depend on the estimates for its operation (for instance, routing). In the case of $r_{min}$-routing, $r_{min}$-routes may fail to meet the specified target reliability if the link metric overestimates the link quality. For instance, if the estimated link quality on link $e$ is 0.90, and the observed is 0.85. However, the opposite, underestimating, does not affect $r_{min}$-routes, i.e., the observed is 0.90, and the estimated is 0.85. Thus, it is beneficial to be conservative though resistant to outliers. We discuss further the properties of OB-EWMA and its applicability in reliability-constrained routing in coming sections.

### 6.3.2 Definition of OB-EWMA

OB-EWMA (Outlier Bounded Exponential Weighted Moving Average)[MG21] is an EWMA based metric measured on the outlier boundary of empirical PDR distributions for a specific history duration. It is devised for reliability-based routing protocols on contention-based wireless networks. Similar as in [WTC03], time is structured into a sequence of time slots of constant duration $d_{ps}$ called PDR slots. PDR slots are consecutively numbered, starting with 1. For each PDR slot $n_{ps} \in \mathbb{N}_+$ and each link $e \in V \times V$, a packet delivery ratio $pdr^e_{raw}(n_{ps})$ is calculated based on data collected at run time. The duration $d_{ps}$ of PDR slots has to be chosen such that a sufficient number of frames is sent on each link to obtain meaningful statistical PDR values. We enforce that a minimum number of frames $n_{f,min}$ is sent on each link in each PDR slot to obtain meaningful statistical PDR values. If the number of payload frames is insufficient, additional probing frames are sent. However, the

duration $d_{ps}$ of PDR slots has to be chosen suitably to limit the bandwidth usage ratio caused by this traffic.

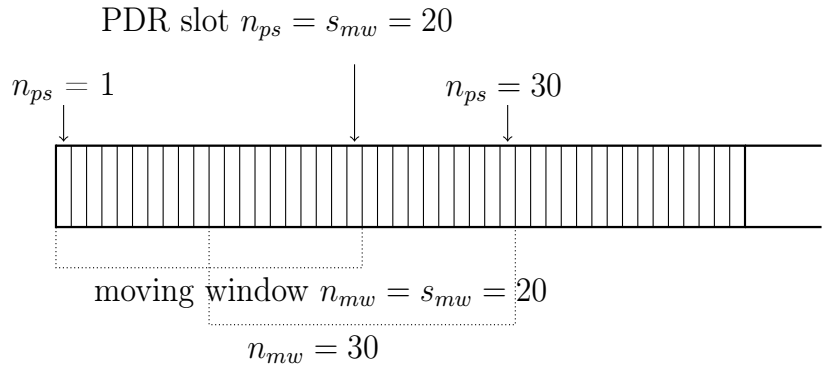



Figure 6.1: Moving window with size $s_{mw} = 20$ PDR slots [MG21].

Next, moving windows of size $s_{mw} \geq 1$, consisting of a fixed number $s_{mw}$ of consecutive PDR slots, are built. Moving windows are consecutively numbered, starting with $n_{mw}$, where moving windows $n_{mw}$ comprise PDR slots $n_{ps}$, i.e. $n_{mw} - s_{mw} + 1 \leq n_{ps} \leq n_{mw}$. In such a way, for $n_{mw} \geq s_{mw}$, the numbering of PDR slots and moving windows are aligned. Fig. 6.1 depicts the concept of moving windows over PDR slots.

In Chapter 3, the original concept of moving windows was reused for the offline evaluation of statistical link properties. Further, we extended the network model defined in Chapter 2, and notations were derived by extending the model defined in Chapter 3. Corresponding to Bernoulli slots defined in Chapter 3, and in the original concept, the difference is that the length of PDR slots is not in the number of Bernoulli trials but is measured in time.

Recalling the frame format of WiPS frames at a receiver, it comprised the WiPS node address *snd* and *rcv* of the sender and receiver, respectively, transmission counter value *txCtr*, and the local timestamp value *rxTs* at which the frame arrived in the user stack of the receiver node. Let $\bar{F}^e_{rx}(n_{ps}) \subseteq F_{rx}$ and $n^e_{rx}(n_{ps})$ denote the set of frames received on link $e = (v, v')$ and the number of received frames during PDR slot $n_{ps}$, respectively. These are derived as:

$$\begin{aligned}\bar{F}^e_{rx}(n_{ps}) =& \{f \in F_{rx} \mid f.snd = addr(v) \wedge f.rcv = addr(v') \\ & \wedge\ t_0 + d_{ps} \cdot (n_{ps} - 1) \leq f.rxTs < d_{ps} \cdot n_{ps}\}\end{aligned}$$

where, $t_0$ is the starting point of the experiment

$$n^e_{rx}(n_{ps}) = \mid \bar{F}^e_{rx}(n_{ps}) \mid \tag{6.1}$$

From $\bar{F}^e_{rx}(n_{ps})$, node $v'$ derives the number of frames $n^e_{tx}(n_{ps})$ that have been sent on link $e$ in PDR slot $n_{ps}$ as:

$$n^e_{tx}(n_{ps}) = \begin{cases} \max(n_{f,min}, \max(f.txCtr \mid f \in F^e_{rx}(n_{ps})) - \\ \qquad \max(f.txCtr \mid f \in F^e_{rx}(n_{ps}-1))) & \text{if } n^e_{rx}(n_{ps}) > 0 \\ n_{f,min} & \text{if } n^e_{rx}(n_{ps}) = 0 \end{cases} \tag{6.2}$$

From this, the packet delivery ratio $pdr^e_{raw}(n_{ps})$ during the PDR slot $n_{ps}$ on link $e$ required to calculate OB-EWMA is determined as:

$$pdr^e_{raw}(n_{ps}) = \min\left(1\,,\, \frac{n^e_{rx}(n_{ps})}{n^e_{tx}(n_{ps})}\right) \tag{6.3}$$

The PDR $pdr^e_{raw}(n_{ps})$ may not always be accurate as frames received in slot $n_{ps}$ may have been already transmitted in slot $n_{ps}-1$ (for $n_{ps} > 1$). In addition, the minimum and maximum values of *txCtr* may deviate from accurate values, as frames at the beginning and/or the end of a time slot may have been lost. Nevertheless, this is statistically compensated by having PDR slots with sufficiently many transmissions on each link and by aggregating PDR slots into sufficiently large moving windows.

Next, we define the metric OB-EWMA. It is defined in two steps.

- First, we introduce a function OB (Outlier Boundary):

$$\begin{aligned} &\mathrm{OB}^e_p(n_{mw}, s_{mw}) = \text{empirical } p\text{-percentile of the set of values} \\ &\{pdr^e_{raw}(k) \mid n_{mw} - s_{mw} + 1 \le k \le n_{mw}\} \text{ for } n_{mw} \ge s_{mw}, \text{ else } \textit{undefined} \end{aligned} \tag{6.4}$$

For link $e \in E$, $p \in \mathbb{R}_{0,1}$, and moving window $n_{mw}$ of size $s_{mw}$, the function $\mathrm{OB}^e_p(n_{mw}, s_{mw})$ yields the $p$-percentile of the PDR values observed in this moving window. The $p$-percentile is the $pdr^e_{raw}$ value defining the boundary between regular PDR values and downward outliers. It is determined such that 100 times $p$ (for instance $p$=0.1, then 10%) percentage of the PDR values of link $e$ observed in the

specified moving window are smaller than or equal to $pdr^e_{raw}$. The conservativeness of the estimation depends on the chosen $p$ value. The smaller the $p$, the more conservative the estimations of link quality become. Further, $\text{OB}^e_p(n_{mw}, s_{mw})$ is undefined for $n_{mw} < s_{mw}$, because this would not yield statistically meaningful results.

- Second, the link quality metric OB-EWMA (Outlier Bounded Exponential Weighted Moving Average) is introduced:

$$\text{OB-EWMA}^e_{\alpha,p}(n_{mw}, s_{mw}) = \begin{cases} \textit{undefined}, & \text{if } n_{mw} < s_{mw} \\ \text{OB}^e_p(n_{mw}, s_{mw}), & \text{if } n_{mw} = s_{mw} \\ \alpha \cdot \text{OB-EWMA}^e_{\alpha,p}(n_{mw} - 1, s_{mw}) + & \\ \qquad (1-\alpha) \cdot \text{OB}^e_p(n_{mw}, s_{mw}), & \text{if } n_{mw} > s_{mw} \end{cases} \tag{6.5}$$

For link $e$ and $\alpha$, $p \in \mathbb{R}_{0,1}$, $\text{OB-EWMA}^e_{\alpha,p}(n_{mw}, s_{mw})$ determines the link quality value of moving window $n_{mw}$ of size $s_{mw}$, taking the history of link qualities and observed PDR values during the PDR slots $pdr^e_{raw}(n_{mw})$ in the moving window $n_{mw}$ into account. The value $\alpha$ determines the weight given to the history (i.e., the previous link estimates). The smaller the $\alpha$, the more adaptive the metric becomes, whereas a larger value for $\alpha$ increases the stability. Note that $\text{OB-EWMA}^e_{\alpha,p}(n_{mw}, s_{mw})$ is defined for complete moving windows of size $s_{mw}$, i.e., only for $n_{mw} \geq s_{mw}$, else it would not yield a meaningful estimation.

The design of OB-EWMA is motivated by analyzing the PDR distribution of different WiFi links. It is discussed further in later sections (see Sec. 6.5.1). Further, it is important to note that sufficient statistical data needs to be gathered before providing link quality estimates. It is achieved by sending a minimum number of frames $n_{f,min}$ on each link in each PDR slot and by waiting for $s_{mw}$ PDR slots before calculating $\text{OB-EWMA}^e_{\alpha,p}$.

Concluding, by selecting suitable values for parameters $d_{ps}$, $s_{mw}$, $p$, and $\alpha$, OB-EWMA can be parametrized to satisfy the properties mentioned in Sec. 6.3.1.

1. **Based on PDR**: The link metric OB-EWMA applies the function OB (Outlier Boundary, see Eqn. 6.3.2), which determines the empirical $p$-percentile of the set of recent PDR values. The parameters $n_{f,min}$ and $d_{ps}$ are set to obtain meaningful statistical PDR values.

2. **Adaptivity**: The parameters $s_{mw}$ and $\alpha$ of OB-EWMA determine the degree of adaptivity. For instance, the parameter $s_{mw}$ defines the size of the history (of the raw data) to be considered. Smaller values for $s_{mw}$ increase adaptivity,

i.e., adapt to recent changes. An exponential weight is given to the history of link estimates with $\alpha$, and using smaller values increase adaptivity.

3. **Stability**: The same parameters, $s_{mw}$ and $\alpha$ determine the degree of stability of the link estimates. Larger values for the parameters increase the stability, which opposes adaptivity.

4. **Conservative estimates**: OB-EWMA uses the function OB, parameterized with $p$ to determine the empirical $p$-percentile. The estimates become more conservative for smaller values of $p$. However, a very small value for $p$ is prone to outliers, and it is necessary to configure a "meaningful" value for $p$.

We elaborate on the effect of these parameters in the coming sections.

## 6.4 Link metrics used in evaluation

To demonstrate the applicability of OB-EWMA and compare it with state-of-the-art link metrics, we selected the following metrics in our evaluation.

### 6.4.1 Window Mean with EWMA (WMEWMA)

One of the earliest and most commonly used EWMA based link reliability metrics is Window Mean with EWMA (WMEWMA). As stated in Sec. 6.2, in [WC03], the authors show that WMEWMA outperforms the metrics TWMA, Flip-Flop EWMA, and moving average. It is designed to estimate the Packet Delivery Ratio (PDR) by inputting the recently observed PDR values on the link. Here, for fair comparison with OB-EWMA, the observed PDR value during a moving window $n_{mw}$ is provided as the input to calculate WMEWMA.

Let $n^e_{rx}(n_{mw}, s_{mw})$ and $n^e_{tx}(n_{mw}, s_{mw})$ be the number of received and transmitted frames during the moving window $n_{mw}$ with size $s_{mw}$, respectively. By applying Eqn. 6.3.2, for $n_{mw} \geq s_{mw}$,

$$n^e_{rx}(n_{mw}, s_{mw}) = \sum_{n_{ps}=n_{mw}-s_{mw}+1}^{n_{mw}} n^e_{rx}(n_{ps}) \tag{6.6}$$

Similarly, by applying Eqn. 6.3.2, for $n_{mw} \geq s_{mw}$,

$$n^e_{tx}(n_{mw}, s_{mw}) = \sum_{n_{ps}=n_{mw}-s_{mw}+1}^{n_{mw}} n^e_{tx}(n_{ps}) \tag{6.7}$$

From this, PDR $pdr^e_{raw}(n_{nw}, s_{mw})$ during the moving window $n_{mw}$ with size $s_{mw}$ for link $e$ is determined as:

$$pdr^e_{raw}(n_{nw}, s_{mw}) = min\left(1, \frac{n^e_{rx}(n_{mw}, s_{mw})}{n^e_{tx}(n_{mw}, s_{mw})}\right) \tag{6.8}$$

The estimated link quality using WMEWMA after moving window $n_{mw}$ with size $s_{mw}$ is

$$\text{WMEWMA}_\alpha(n_{mw}, s_{mw}) = \begin{cases} pdr^e_{raw}(n_{mw}, s_{mw}), & \text{if } n_{mw} = s_{mw} \\ \alpha \cdot \text{WMEWMA}_\alpha(n_{mw}, s_{mw}) + \\ \quad (1-\alpha) \cdot pdr^e_{raw}(n_{mw}, s_{mw}), & \text{if } n_{mw} > s_{mw} \\ \text{undefined}, & \text{otherwise} \end{cases} \tag{6.9}$$

## 6.4.2 FaLQE

The link metric FaLQE is similar to WMEWMA, where the history weight parameter $\alpha$ is dynamic. Here, $\alpha$ depends on the difference between the observed PDR value in the current and previous window, denoted as $\delta_{pdr}$.

$$\delta_{pdr}(n_{nw}, s_{mw}) = |pdr^e_{raw}(n_{mw}, s_{mw}) - pdr^e_{raw}(n_{mw} - 1, s_{mw})| \tag{6.10}$$

Then, $\alpha$ for the moving window $n_{mw}$ is calculated as,

$$\alpha(n_{nw}, s_{mw}) = \begin{cases} 0.6 & \delta_{pdr}(n_{nw}, s_{mw}) < 0.2 \\ 0.4 & 0.2 \leq \delta_{pdr}(n_{nw}, s_{mw}) \leq 0.5 \\ 0.1 & \delta_{pdr}(n_{nw}, s_{mw}) > 0.5 \end{cases} \tag{6.11}$$

FaLQE estimated during the moving window $n_{mw}$ with size $s_{mw}$ is

$$\text{FaLQE}(n_{mw}, s_{mw}) = \begin{cases} pdr^{e}_{raw}(n_{mw}, s_{mw}), & \text{if } n_{mw} = s_{mw} \\ \alpha(n_{nw}, s_{mw}) \cdot \text{FaLQE}(n_{mw} - 1, s_{mw}) + \\ \quad (1 - \alpha(n_{nw}, s_{mw})) \cdot pdr^{e}_{raw}(n_{mw}, s_{mw}), & \text{if } n_{mw} > s_{mw} \\ \text{undefined}, & \text{otherwise} \end{cases} \tag{6.12}$$

### 6.4.3 F-LQE

F-LQE is a fuzzy logic-based link quality estimator that uses multiple raw link information for its calculation [Bac+10]. It uses the following parameters:

1. Link reliability, denoted as "Smoothed PRR" (SPRR) [19], which is the same as WMEWMA.

2. Link stability, represented as SF. It is calculated as

$$\frac{\sigma_{PDR}}{\mu_{PDR}}$$

   where, $\sigma_{PDR}$ and $\mu_{PDR}$ are the Standard Deviation (SD) and mean of $n$ observed PDR values, respectively. In our evaluation, "*n measured PDR values*" are the PDR values of the PDR slots in a moving window.

3. Asymmetry, i.e., difference between the uplink and downlink qualities. It is calculated as

$$\left| pdr^{(v,v')}_{raw}(n_{mw}, s_{mw}) - pdr^{(v',v)}_{raw}(n_{mw}, s_{mw}) \right|$$

   As mentioned in Sec. 6.2, F-LQE considers the link quality as bidirectional.

4. Average Signal-to-Noise Ratio (SNR).

F-LQE is not designed for a WiFi-based network (designed for CC2420 hardware). Modifications are required to implement it for other platforms. For instance, the authors permit to exclude parameters from the calculation. We calculate F-LQE separately for both directions since we focus on the uni-directional link quality. Further, access to the WiFi hardware is required to calculate SNR, and thus it is not considered.

[19] PDR referred to as PRR in the original paper [Bac+10]

The calculation of F-LQE involves multiple steps. We calculate F-LQE for each moving window. The process starts by calculating the *SPRR* and SF. As mentioned, *SPRR* is same as WMEWMA

$$SPRR_{\alpha}(n_{mw}, s_{mw}) = \text{WMEWMA}_{\alpha}(n_{mw}, s_{mw}) \tag{6.13}$$

Link stability parameter SF during moving window $n_{mw}$ with size $s_{mw}$

$$SF(n_{mw}, s_{mw}) = \frac{\sigma_{PDR}(n_{mw}, s_{mw})}{\mu_{PDR}(n_{mw}, s_{mw})} \tag{6.14}$$

where, $\sigma_{PDR}(n_{mw}, s_{mw})$ is the SD of raw PDR values of PDR slots $pdr^{e}_{raw}(n_{ps})$ in the moving window $n_{mw}$, whereas $\mu_{PDR}(n_{mw}, s_{mw})$ is the mean. Next, the calculated values of the metrics, *SPRR* and *SF*, are inputted to membership functions, having piecewise linear forms $u_{sprr}$ and $u_{sf}$, respectively.

$$u_{sprr}(n_{mw}, s_{mw}, \alpha) = \begin{cases} 1, & \text{if } SPRR_{\alpha}(n_{mw}, s_{mw}) > 0.95 \\ ((10/7) * SPRR_{\alpha}(n_{mw}, s_{mw})) - 0.357, & \text{if } 0.25 > SPRR_{\alpha}(n_{mw}, s_{mw}) \leq 0.95 \\ 0 & \text{otherwise} \end{cases} \tag{6.15}$$

$$u_{sf}(n_{mw}, s_{mw}) = \begin{cases} \left(-(10/7) * \dfrac{\sigma_{PDR}(s_{mw}, n_{mw})}{\mu_{PDR}(s_{mw}, n_{mw})}\right) + 1, & \text{if } SF(n_{mw}, s_{mw}) < 0.7 \\ 0 & \text{otherwise} \end{cases} \tag{6.16}$$

Next, the link quality for a subset of PDR values (in our evaluation, it is the PDR values in a moving window), $\mu_{\text{F-LQE}}$, is calculated as

$$\begin{aligned} \mu_{\text{F-LQE}_{\alpha}}(n_{mw}, s_{mw}) = \beta \cdot min(u_{sprr}(n_{mw}, s_{mw}, \alpha), u_{sf}(n_{mw}, s_{mw})) \\ + (1 - \beta) \cdot mean(u_{sprr}(n_{mw}, s_{mw}, \alpha), u_{sf}(n_{mw}, s_{mw})) \end{aligned} \tag{6.17}$$

The recommended values for the $\beta$ range between 0.5 to 0.8, where the authors suggest a $\beta$ value of 0.6 for "better results". Next, F-LQE or the link metric is calculated by applying EWMA over the $\mu_{\text{F-LQE}}$ for an alpha of 0.6.

$$\text{F-LQE}_{\alpha}(n_{mw}, s_{mw}) = \begin{cases} \mu_{\text{F-LQE}_{\alpha}}(n_{mw}, s_{mw}), & \text{if } n_{mw} = s_{mw} \\ \alpha \cdot \text{F-LQE}_{\alpha}(n_{mw}, s_{mw}) + & \\ \quad (1-\alpha) \cdot \mu_{\text{F-LQE}_{\alpha}}(n_{mw}, s_{mw}), & \text{if } n_{mw} > s_{mw} \\ \text{undefined}, & \text{otherwise} \end{cases} \tag{6.18}$$

### 6.4.4 *min*-EWMA

We introduce the link metric *min*-EWMA to capture the lower bound for conservative estimates, defined as OB-EWMA with percentile $p = 0$. It is introduced to illustrate the influence of outliers being too conservative in link quality estimation. *min*-EWMA is calculated as

$$min\text{-EWMA}^{e}_{\alpha}(n_{mw}, s_{mw}) = \text{OB-EWMA}^{e}_{\alpha,0}(n_{mw}, s_{mw}) \tag{6.19}$$

## 6.5 Experimental Evaluation

In this section, we further discuss the motivation behind the design of OB-EWMA using real-world experiment and synthetic data, followed by the experimental evaluation of the proposed metric. The experiment results presented in this section are first reported in [MG21]. Real-world experiments are conducted to evaluate the link metric and compare the performance to other metrics. Experiments are performed in a WiFi testbed (the same testbed used for other experiments in this thesis), consisting of 11 nodes deployed in a university environment. All testbed nodes are built using PC Engine APU2 SBCs running Arch Linux (see Sec. 2.4.2). Fig. 6.2 shows the placement of nodes used in the experiments.

Mainly, the experiments are divided into two groups. First, multiple experiments are conducted to collect raw link reliability information using different transmission settings, i.e., different transmission rates. These experiments are conducted for 3 hours, where nodes transmit frames in a random interval between 100 - 300ms between each frame transmission on channel 6 (2.437 GHz). In the experiments, we configured $n_{f,min} = 30$, and different PDR slot size duration $d_{ps} \in \{10, 20, 30\}$ (seconds). Thus, the shortest PDR slot used in the experiments ($d_{ps} = 10$s) has at

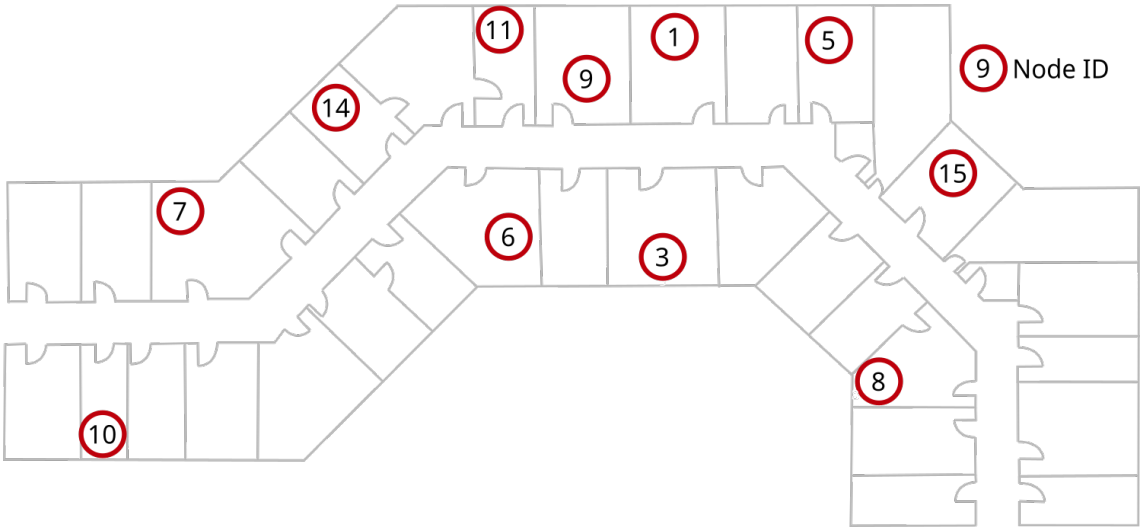



Figure 6.2: Node placement in the testbed

least 33 transmission events. The continuous frame losses on a link are identified and handled by the Topology Explorer layer of the WiPS framework (see Sec. 2.4). The data collected during the experiments are then aggregated (offline analysis) to evaluate or compare the performance of link metrics.

Following that, we conduct real-world experiments to depict the influence of the proposed metric in $r_{min}$-routing. We explain further while discussing the results.

### 6.5.1 Properties of PDR distribution

Sec. 6.3.1 discussed the motivation behind developing the link metric OB-EWMA. As discussed in Sec. 6.2, most link metrics are designed to accurately estimate the link reliability or success probability of upcoming transmission events. The performance of those link metrics is measured based on the estimation accuracy, i.e., how close the estimates are to the observed. However, when utilizing link metrics for routing, overestimating the link quality is more disruptive than underestimating. For instance, in the case of $r_{min}$-routing, if the estimated link quality is higher than the actual, it may affect its operations. I.e., if the estimated link quality is higher than observed, $r_{min}$-routing may assign a smaller number of transmissions on a link than the required number to meet the specified target reliability. On the contrary, if the link quality is underestimated, this problem does not exist. However, being too conservative may increase bandwidth usage. Therefore, it is important to reduce the likelihood of overestimation and also, at the same time, limit the conservativeness in the estimation.

Further, to better understand the properties of WiFi links, we examine the distribution of PDR ($pdr^e_{raw}(n_{ps})$) values of each link $e$, for different slot durations $d_{ps} \in \{10, 20, 30\}$. In the experiments, 47 links show an average PDR of above 40%. Observing the raw PDR values $pdr^e_{raw}(n_{ps})$, a higher percentage of observed PDR values of links oscillates. Examining the PDR value distributions of links, the values are left-skewed, with skewness ranging from 0 to -5 (deviation from a normal distribution). Fig. 6.3 shows the density plot of observed PDR values of a subset of links where the duration of PDR slots are $d_{ps} = 20$s. It shows that the downward outliers are less frequent than PDR values above the average.

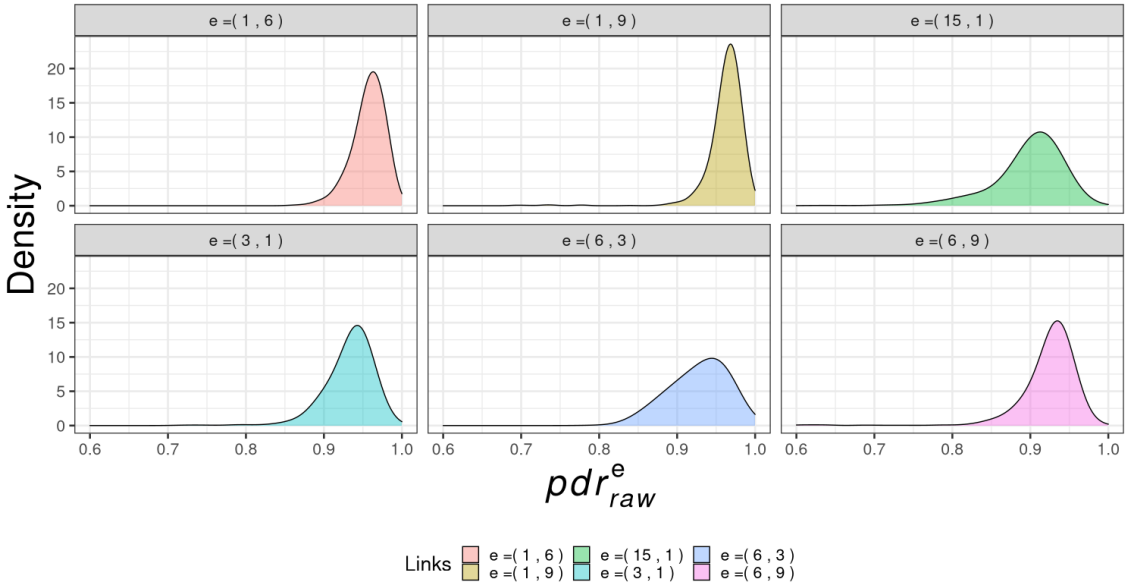



Figure 6.3: Density plot of observed PDR values ($pdr^e_{raw}(n_{ps})$, where $d_{ps} = 20s$) of a subset of links

There may have several explanations for this left-skewness. For instance, it may occur if there are high downward outliers, i.e., a significant drop in link quality from the average, or if values have an upper bound (the PDR values with a maximum of 1). Both cases fit our observation in the experiments for links with high reliability. Furthermore, the shapes of observed PDR value distributions suggest that there is only a small number of extreme downward outliers. Thus, for a conservative estimate of link quality, it is feasible to consider a lower $p$-percentile with a sufficiently small $p > 0$. It also helps to avoid rare outliers in the calculation. We illustrate this advantage (being resistant to outliers) while we discuss the performance of OB-EWMA to other metrics in the coming section. Moreover, compared to symmetric or right-skewed distributions of PDR values, the $p$-percentile is far from the minimum, increasing usable link reliability.

### 6.5.2 Metric Evaluation

Next, we examine the link data collected from multiple experiments. We exemplify our grounds by applying different link metrics on synthesized data and data collected from the testbed. Fig. 6.4 shows the observed PDR values for link $e_1$ and $e_2$, $pdr^e_{obs}(n_{mw}) = pdr^e_{raw}(n_{mw})$, where the duration of the PDR slots $d_{ps} = 10$s for a sequence of 500 PDR slots. Further, the figure depicts the estimated link quality $pdr^e_{est}(n_{mw}, s_{mw}) = \text{OB-EWMA}^e_{\alpha,p}(n_{mw} - 1, s_{mw})$, with $\alpha = 0.9$ and $p = 0.1$. In our experiments, we have used a $p = 0.1$, which yields a conservative estimate, and at the same time, resistant outliers. The percentile value ($p = 1$) is arbitrarily selected by observing the distribution of observed PDR values.

As expected, the results show that the link quality estimates are adaptive, conservative, and sufficiently stable, with only a few overestimated link qualities. I.e., the estimates adapted to the changes in the link quality, and they are conservative, resulting only in few overestimated windows. For the metric $\text{WMEWMA}_{0.9}$, the authors recommend a history weight $\alpha$ of 0.6. However, this yields a high fluctuation, and therefore, we use a history weight $\alpha$ of 0.9 in the following experiments for higher stability. Fig. 6.5 shows the estimations by each link metric on link $e_1 = (15, 8)$. Metrics $\text{WMEWMA}_{0.9}$ and F-LQE estimate a link quality close to the average of the observed PDR. It led to too many overestimated estimates. As depicted in the figure, estimations by FaLQE are unstable. The variance in the observed PDR affected the stability of the estimates by FaLQE. Our metric OB-EWMA, as expected and mentioned, returns an adaptive, conservative, and sufficiently stable estimate.

Another notable characteristic of OB-EWMA is that it implicitly captures the volatility in its calculation. I.e., the Standard Deviation (SD) of observed PDR values for a link is implicitly considered by selecting a low value for $p$. To illustrate the effect, we use synthetic data. Consider three links with observed PDR values having the same average but with a different SD. Fig. 6.6 shows three links, $e_3$, $e_4$, and $e_5$, with synthesized PDR values (randomly generated) with a SD of 0.014, 0.029, and 0.057, respectively, and a mean PDR value of 0.85. The link metrics are applied to the synthesized data with a moving window of size $s_{mw} = 30$. The grey dotted line denotes the synthesized PDR values.

Comparing the results, the link metrics $\text{WMEWMA}_{0.9}$ (with $\alpha = 0.9$) and FaLQE return a link quality estimate close to the mean, 0.85 (see Figures 6.6a, 6.6b, and 6.6c). As the SD increases, the stability of estimated link quality by $\text{WMEWMA}_{0.9}$ and FaLQE decreases. F-LQE, the fuzzy logic-based link metric, slightly responds to the change in SD. I.e., the estimated link quality for the links relatively decreased as the SD increased. However, there is a large number of overestimated windows by

F-LQE. We discuss overestimation by the metrics later in our analysis. Further, the proposed metric OB-EWMA implicitly considered the volatility in its calculation.

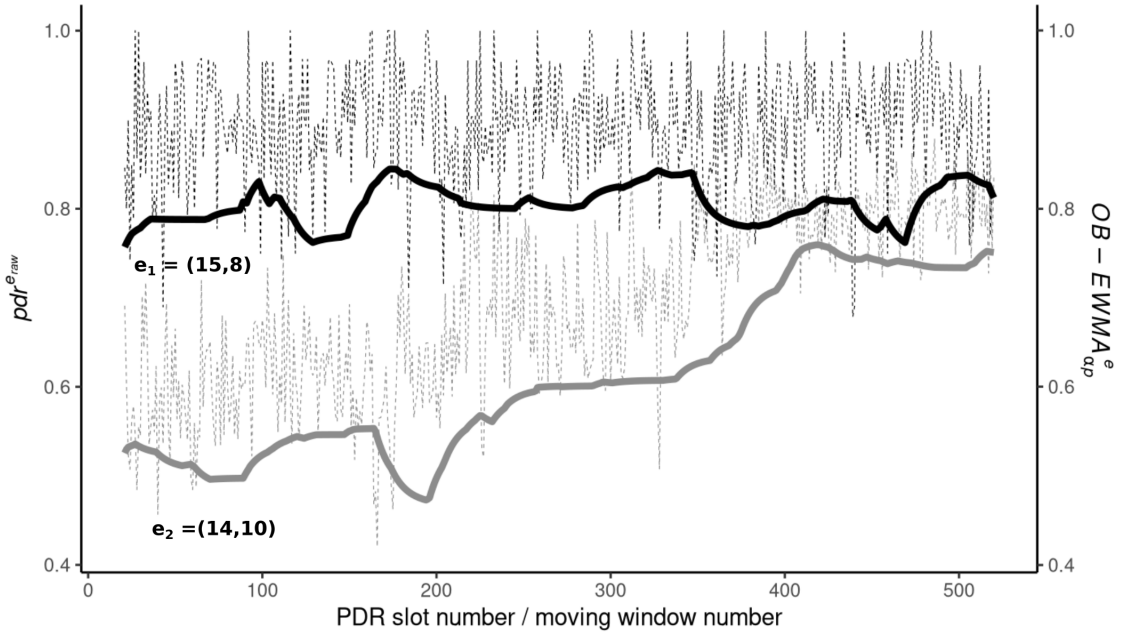



Figure 6.4: Observed and estimated ($\text{OBEWMA}^{e}_{\alpha,p}$) link qualities of link $e_1 = (15, 8)$ (black coloured) and $e_2 = (14, 10)$ (grey coloured) during an experiment where the estimation metric uses a PDR slot duration of $d_{ps} = 10$ and moving window size $s_{mw} = 30$.

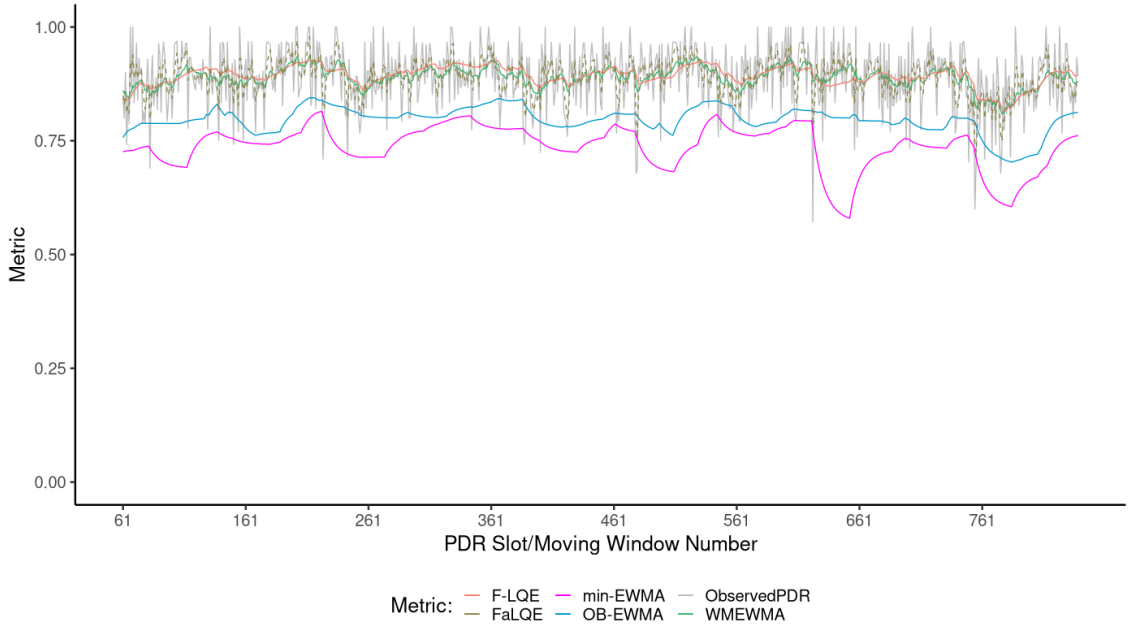
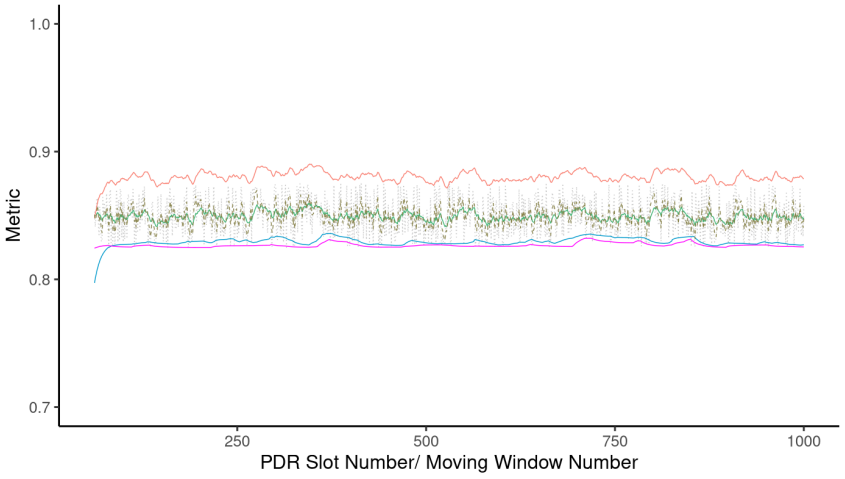
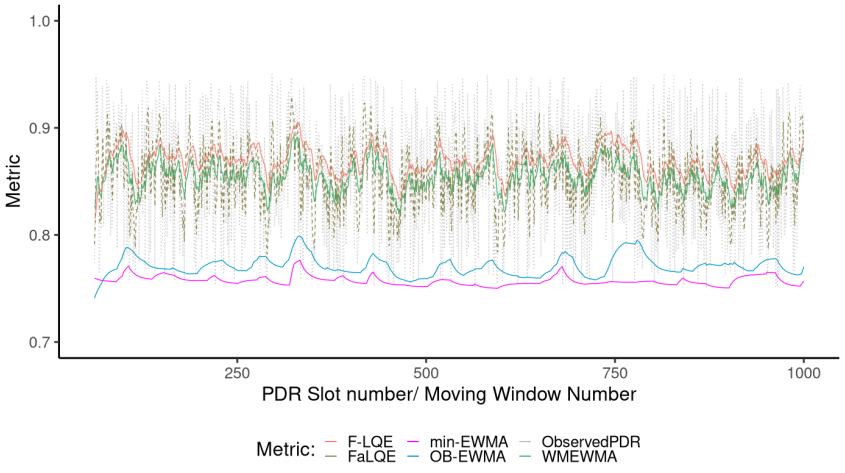



Figure 6.5: Applying link metric on link $e_1 = (15, 8)$ where $d_{ps} = 10$ and $s_{mw} = 30$.




(a) link $e_3$ with a SD of 0.014 and mean value of 0.85.

(b) link $e_4$ with a SD of 0.029 and mean value of 0.85.




(c) link $e_5$ with a SD of 0.057 and mean value of 0.85

Figure 6.6: Three links $e_3$, $e_4$, and $e_5$ having synthetically generated PDR values where synthesized data have the same mean (0.85) and different standard deviation.

I.e., OB-EWMA using low values for $p$ accounts for the variance of observed PDR values and evaluates the links differently.

Examining Fig. 6.6, the results show that *min*-EWMA (OB-EWMA with $p = 0$) also implicitly considered the variance in its calculation. However, lowering the $p$ to 0 or being too conservative makes the metric (estimation) prone to outliers, and in practice, outliers are often. Fig. 6.7 shows the observed PDR values for link $e = (3, 5)$ (result from a real-world experiment) and the estimations by the different link metrics with PDR slot duration $d_{ps} = 30s$ and moving window size $s_{mw} = 30$. There are multiple moving windows where the most conservative metric *min*-EWMA reacted to outliers. For instance, the link observed a significant drop in PDR during the PDR slots $n_{ps}$ 124 and 1217, and recovered immediately. *min*-EWMA was unable to capture the extreme outliers even by being too conservative. Further, it reacted to this extreme outliers, which resulted in a significant variation for the subsequent link quality estimates. Furthermore, this may result in unnecessary route recalculations (if a routing protocol uses the link metric for route calculation). Examining the results further, between PDR slots $n_{ps}$ (or moving windows) 385 and 395, the observed PDR on the link dropped for multiple PDR slots. Since the observed link quality dropped for multiple PDR slots, OB-EWMA and other link metrics reacted to these events.

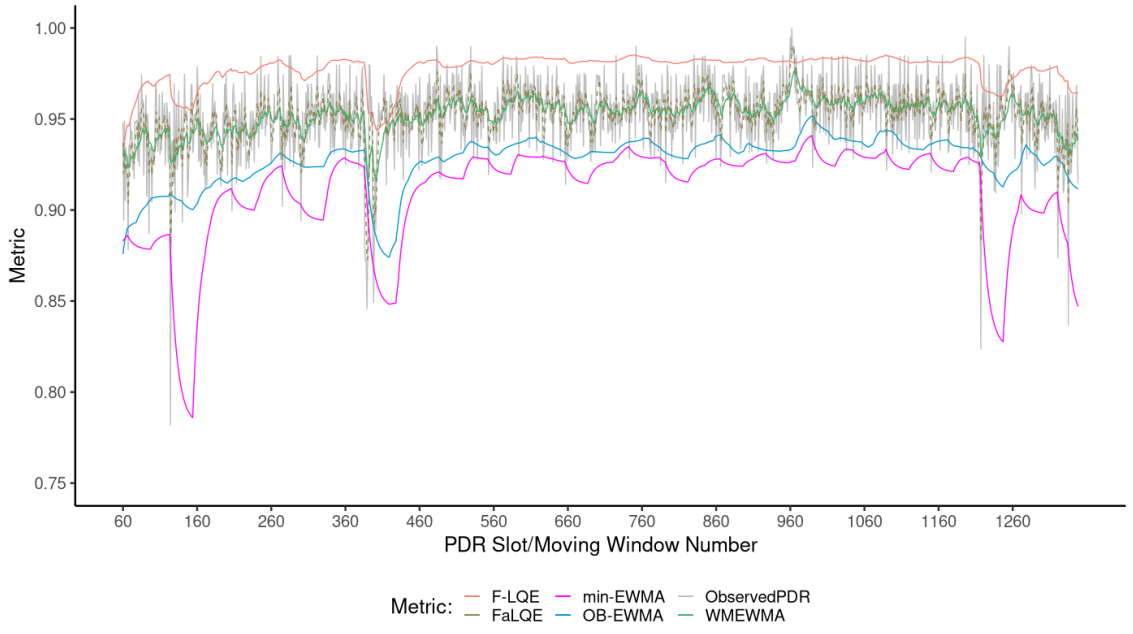



Figure 6.7: Applying metrics to raw data for link $e = (3, 5)$

To further assess the performance of OB-EWMA and compare it with other metrics, we introduce the estimate quality $q^e_{est}(n_{mw}, s_{mw})$ and other quality indicators. The estimate quality $q^e_{est}$ is defined as,

$$\begin{aligned} q^e_{est}(n_{mw}, s_{mw}) &= \frac{pdr^e_{obs}(n_{mw})}{pdr^e_{est}(n_{mw}, s_{mw})} \\ &= \frac{pdr^e_{raw}(n_{mw})}{\text{OB-EWMA}^e_{\alpha,p}(n_{mw}-1, s_{mw})} \end{aligned} \tag{6.20}$$

A $q^e_{est}$ value greater than 1 indicates a good estimate, i.e., the observed link quality is at least the expected or estimated quality by the link metric. Contrary, a value below 1 indicates a wrong estimate. However, slightly wrong estimates may still be acceptable for reliability-constrained routing depending on the path requirements. For instance, in the case of $r_{min}$-routing, there is a safety margin for wrong estimates, as the retransmissions per link are whole-numbered. We introduce further quality indicators to capture good and wrong estimates. To begin with, we define the sets of window numbers of good and wrong estimates during an experiment. Let $N^e_{q^+_{est}}$ and $N^e_{q^-_{est}}$ be the set of moving window numbers of good and bad estimates during an experiment.

$$N^e_{q^+_{est}} = \{k \mid s_{mw}+1 \leq k \leq n_{mw} \wedge\ q^e_{est}(k, s_{mw}) \geq 1\} \tag{6.21}$$

$$N^e_{q^-_{est}} = \{k \mid s_{mw}+1 \leq k \leq n_{mw} \wedge\ q^e_{est}(k, s_{mw}) < 1\} \tag{6.22}$$

Based on the sets $N^e_{q^+_{est}}$ and $N^e_{q^-_{est}}$, the following quality indicators are defined for $n_{mw} > s_{mw}$:

$$r^e_-(n_{mw}, s_{mw}) = \frac{\mid N^e_{q^-_{est}}(n_{mw}, s_{mw}) \mid}{n_{mw} - s_{mw}} \tag{6.23}$$

$$q^e_{estAvg^+}(n_{mw}, s_{mw}) = \frac{1}{\mid N^e_{q^+_{est}}(n_{mw}, s_{mw}) \mid} \cdot \sum_{k \in N^e_{q^+_{est}}(n_{mw}, s_{mw})} q^e_{est}(k, s_{mw}) \tag{6.24}$$

The ratio of wrong estimates, $r^e_-(n_{mw}, s_{mw})$, is a quantitative indicator that indicates the ratio of wrong estimates by a link metric during the experiments. For OB-EWMA, it is expected to be around the value $p$, which is used to determine the $p$-percentile. Further, the quality indicator $q^e_{estAvg^+}(n_{mw}, s_{mw})$ determines the average of all good estimates. A higher value for $q^e_{estAvg^+}$ indicates that estimates have an increased safety margin.

Next, we examine the effect of different settings of $\alpha$, $s_{mw}$, and $d_{ps}$ with $p = 0.1$ on a link. Tab. 6.1 shows the values for the quality indicators defined using the

| $\alpha$ | $s_{mw}$ | $d_{ps}$ | $r^e_-(\%)$ | $q^e_{estAvg^+}$ |
|---|---|---|---|---|
| 0.5 | 20 | 10 | 11.2 % | 1.09 |
| | | 20 | 10.6 % | 1.06 |
| | | 30 | 11.1 % | 1.05 |
| | 30 | 10 | 10.6 % | 1.09 |
| | | 20 | 11.1 % | 1.06 |
| | | 30 | 8.78 % | 1.05 |
| 0.7 | 20 | 10 | 10.9 % | 1.10 |
| | | 20 | 11.1 % | 1.06 |
| | | 30 | 12.9 % | 1.05 |
| | 30 | 10 | 10.9 % | 1.09 |
| | | 20 | 10.2 % | 1.06 |
| | | 30 | 8.82 % | 1.05 |
| 0.9 | 20 | 10 | 11.9 % | 1.10 |
| | | 20 | 11.1 % | 1.06 |
| | | 30 | 11.6 % | 1.05 |
| | 30 | 10 | 10.1 % | 1.06 |
| | | 20 | 10.3 % | 1.07 |
| | | 30 | 8.65 % | 1.09 |

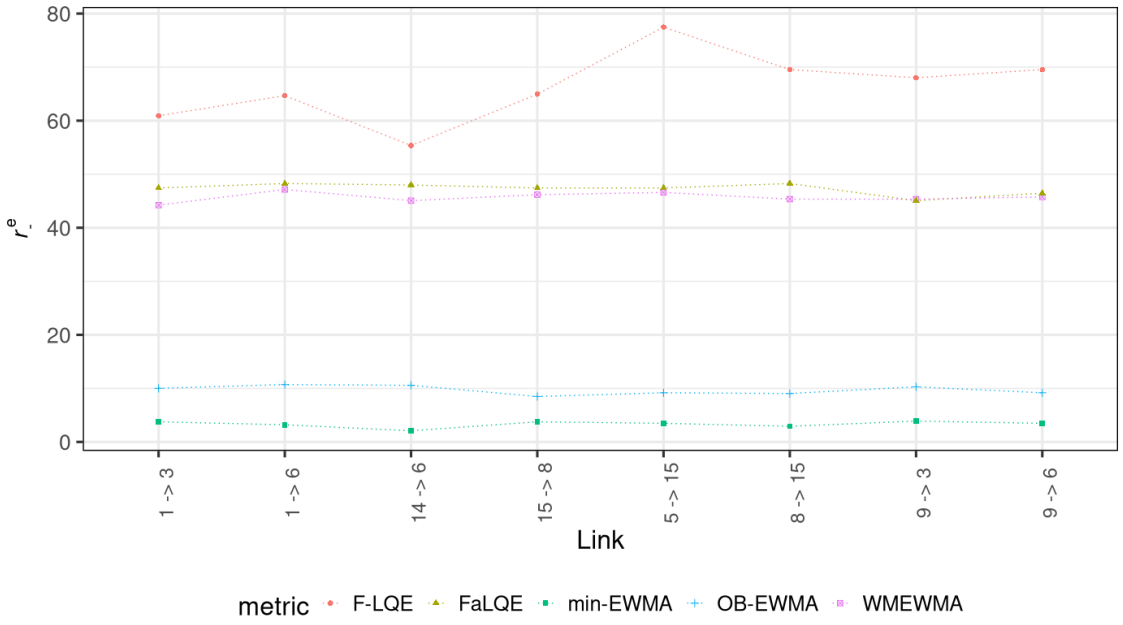
Table 6.1: Effect of different configuration parameters on link $e$ = (15,8) until the last moving window during the experiment for $p = 0.1$

link metric OB-EWMA on link $e = (15, 8)$. The link metric is applied to the same raw data set. Therefore, the number of transmission events in the PDR slots with duration $d_{ps}$ of 30 is higher than 20, similarly for the slots with duration 20 is higher than 10. Results show that the safety margin of good estimates tends to be higher for shorter PDR slots, while larger PDR slots yield a more conservative estimate. Concerning the ratio of wrong estimates $r^e_-(n_{mw}, s_{mw})$, as expected, it is around $p$, i.e., $p = 0.1$.

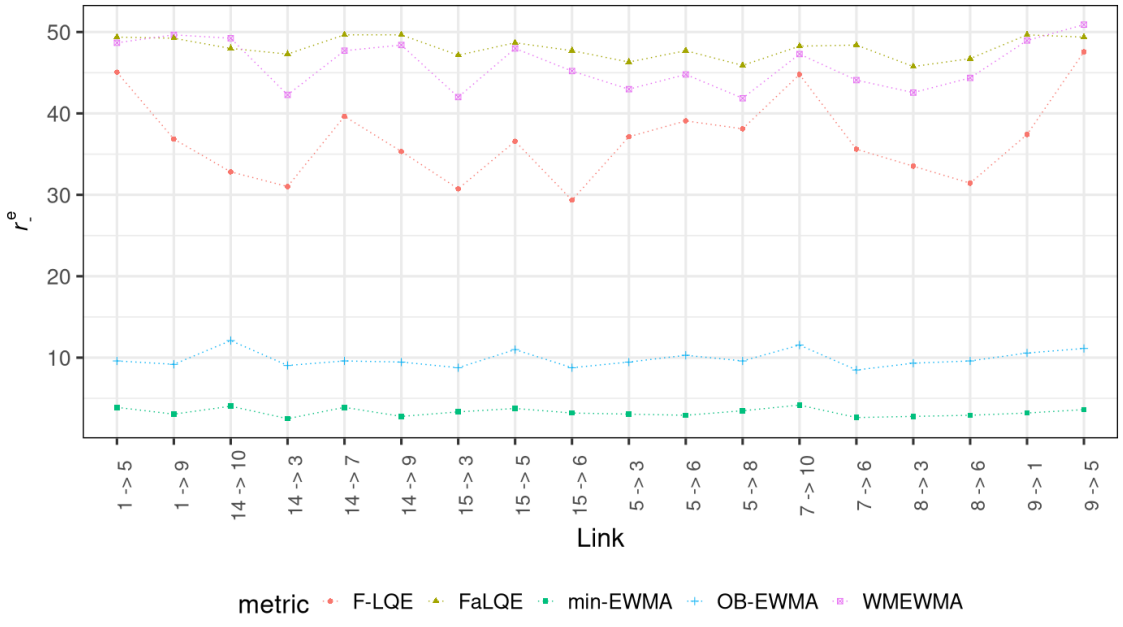
| Metric | $r^e_-$ | $q^e_{estAvg^+}$ |
|---|---|---|
| OB-EWMA | 8.65 % | 1.09 |
| *min*-EWMA | 2.88 % | 1.08 |
| $\text{WMEWMA}_{0.9}$ | 46.1 % | 1.02 |
| FaLQE | 48.01 % | 1.01 |
| F-LQE | 51.1 % | 1.03 |

Table 6.2: Effect of different metrics on link $e$ = (15,8) until the last moving window during the experiment, $d_{ps} = 30s$.

Further, we compare the quality parameters of different metrics for the configuration $\alpha = 0.9$, $s_{mw} = 30$, and $d_{ps} = 30$. Tab. 6.2 shows the results for each metric on link $e = (15, 8)$. The ratio of wrong estimates for $\text{WMEWMA}_{0.9}$, FaLQE, and




(a) Results for links with mean PDR above 0.80.




(b) Results for links with mean PDR between 0.60 and 0.80.

Figure 6.8: The ratio of wrong estimates $r^e_-$ during an experiment.

F-LQE is more than 45% of the total estimate. I.e., compared to OB-EWMA and *min*-EWMA, they have a higher number of overestimated windows.

Fig. 6.8 shows the ratio of wrong estimates $r^e_-(n_{mw}, s_{mw})$ for each link metric during another experiment. Fig. 6.8a shows the ratio of wrong estimates for links with mean PDR above 0.80, and Fig. 6.8b for links between 0.80 and 0.60. Link metrics $\text{WMEWMA}_{0.9}$ and FaLQE overestimated the link quality for 40% to 50% of the moving windows for all links (see Figures 6.8a and 6.8b). F-LQE overestimated link quality for more than 50% of the moving windows for links with mean PDR above 0.80, and 30% to 50% for links between 0.80 and 0.60. I.e., as expected, F-LQE overestimates for high-quality links. As stated in Sec. 6.4.3, the mapping functions used by F-LQE are designed for CC2420 transceivers. A different mapping function considering the aspects of WiFi networks may improve the estimation quality of F-LQE. The proposed metric OB-EWMA, for $p = 0.1$, overestimates up to 12% as expected. Notably, the most conservative metric *min*-EWMA has a ratio of wrong estimates in the range of 3-5%. It shows that even the most conservative metric was unable to anticipate outliers.

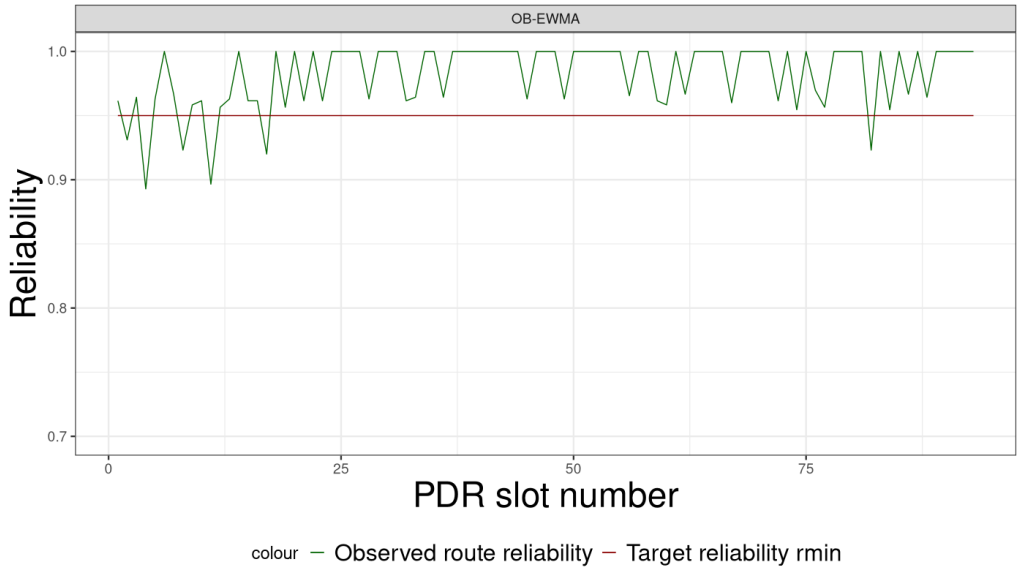
### 6.5.3 Routing Experiments

To show the applicability of OB-EWMA for reliability-constrained routing in contention-based wireless networks, we conducted real experiments in the WiFi testbed with external nodes in the neighborhood. In our experiments, we applied OB-EWMA to $r_{min}$-routing. We also applied $\text{WMEWMA}_{0.9}$ for comparison. The results discussed in this section are first reported in [MG21].

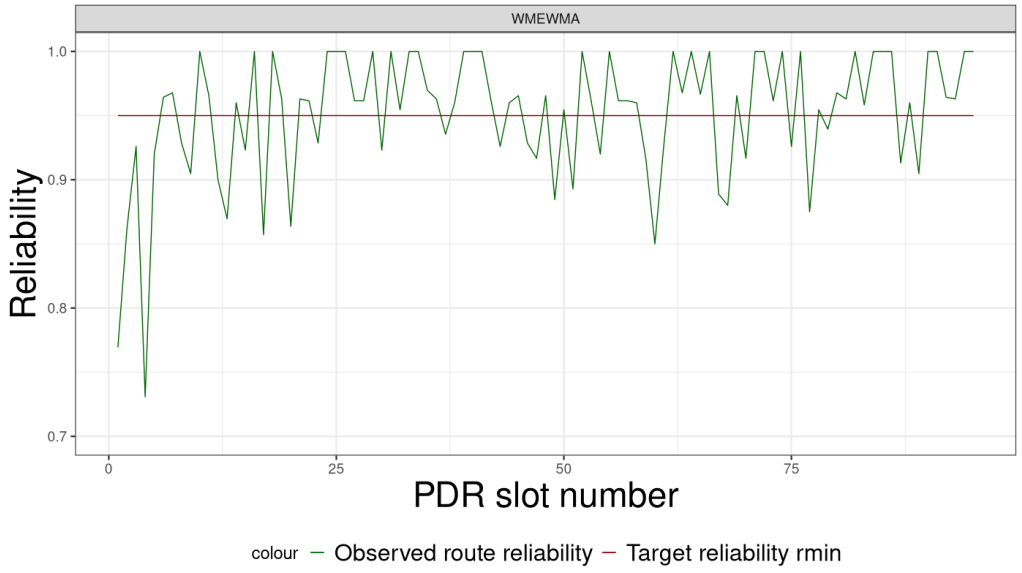
Experiments are conducted on top of the WiPS framework (see Sec. 2.4). We have conducted these experiments with different routes, and the experiments have proceeded as follows.

- The Topology Explorer (ToEx) [Her19] of each node collects and aggregates link information about the incoming links using one of the link metrics OB-EWMA or $\text{WMEWMA}_{0.9}$ at the beginning. The link information is exchanged among nodes and collected by a centralized controller node. After a short stabilization phase, all nodes, including the centralized controller node, have sufficiently accurate global network status information. The topology explorer, ToEx used in the experiments is an extended version of the RRP protocol [Guo+14] modified for 802.11 based WNCS.

- Next, to compare the performance of link metrics, a route $p$ with statistical minimum reliability $r_{min}$ is determined, using OB-EWMA as link quality $q$.

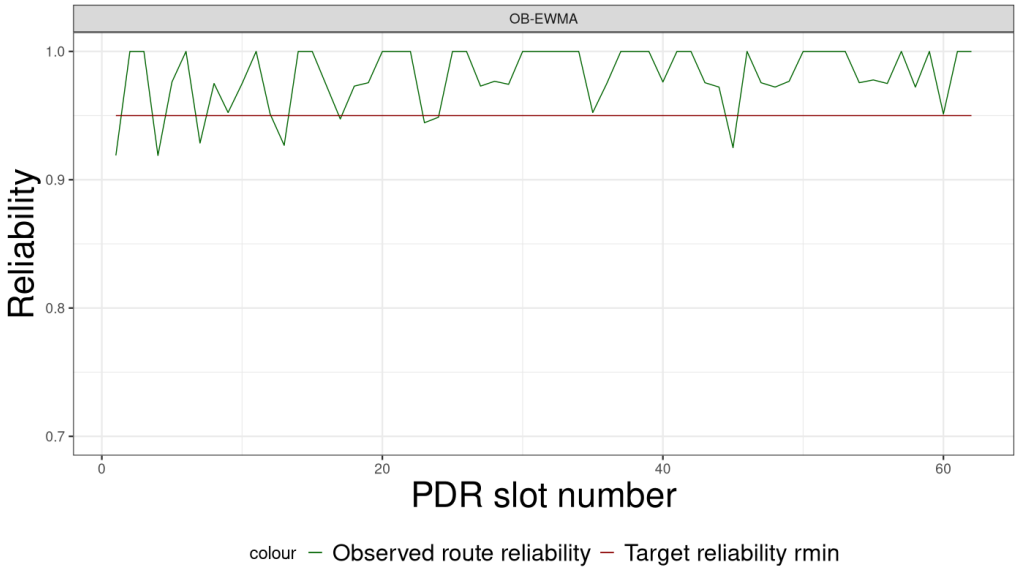



(a) Observed route reliability $r^p_{obs}$ with link metric OB-EWMA




(b) Observed route reliability $r^p_{obs}$ with link metric $\text{WMEWMA}_{0.9}$

Figure 6.9: $r_{min}$-routing experiments on path $p = (5, 3, 8, 15)$ where links used a PDR slot duration $d_{ps} = 20$s. Red line denotes the reliability target $r_{min} = 0.95$ where the green line denotes observed path reliability $r^p_{obs}$




(a) Observed route reliability $r^p_{obs}$ with link metric OB-EWMA

(b) Observed route reliability $r^p_{obs}$ with link metric $\text{WMEWMA}_{0.9}$

Figure 6.10: $r_{min}$-routing experiments on path $p = (5, 3, 8, 15)$ where links used a PDR slot duration $d_{ps} = 30$s. Red line denotes the reliability target $r_{min} = 0.95$ where the green line denotes observed path reliability $r^p_{obs}$

For correct comparison of the metric, we have fixed route $p = (v_1, ..., v_{|p|})$ and use $r_{min}$-routing to determine the maximum number of (re-)transmissions on each link. The maximum number of (re-)transmissions $n^e_{max} \geq 1$ on each link $e = (v_i, v_{i+1}), 1 \leq i \leq \mid p \mid -1$ may differ depending on the underlying link metric. The centralized controller node calculates the number of transmissions required to meet the target reliability.

- The selected route $p$ is operated by sending packets from node $v_1$ to $v_{|p|}$ at a random interval, applying retransmissions up to the maximum number $n^e_{max}$ determined for each link $e$. The destination node $v_{|p|}$ monitors the observed route reliability $r^p_{obs}$ by calculating the actual packet delivery ratio (end-to-end) for each PDR slot.

- If the link qualities of the links used in the route substantially change during the operation, route repairs are performed by adapting the maximum number of transmissions for each link to satisfy the reliability target $r_{min}$. For instance, if the observed link quality of a link used by path $p$ dropped significantly, additional transmission(s) is required to satisfy the reliability target. I.e., during an experiment, the centralized controller periodically receives the latest link information, and upon receiving an update, it recalculates the number of transmissions required to meet the target path reliability. If the requirements change, it updates the source node $v_1$ with the new budget allocation on each link.

In our experiments, we selected node 1 as the centralized controller (see Fig. 6.2). Experiments are conducted during office hours. Further, the statistical link properties are not evaluated in these experiments. Figures 6.9 and 6.10 show $r_{min}$-routing experiment results for the 3-hop route $p = (5, 3, 8, 15)$ with a specified target reliability for the experiments, $r_{min}$, of 0.95 (straight red line in the graph). Fig. 6.9 shows the results of an experiment where the links in the network use a PDR slot duration $d_{ps} = 20$s for link quality estimation, whereas Fig. 6.10 shows results of an experiment with $d_{ps} = 30$s. In both experiments, the size of the moving windows $s_{mw}$ is 30. The experiments were conducted for 30 minutes, where more than 2500 packets were transferred on the route. ToEx used probe frames to satisfy $n_{f,min}$ of 30, i.e., to ensure that at least 30 frames are transmitted during a PDR slot. Experiment results show that $r_{min}$ is missed occasionally when using OB-EWMA. It is due to varying link quality and the fact that some estimates are not correct. I.e., as mentioned in Sec. 6.5.2, such sudden drops in PDR are unpredictable. In the case of varying link quality, the estimates will vary, too, which triggers the route

repair mechanism. Considering the path reliability using $\text{WMEWMA}_{0.9}$, the path reliability $r^p_{obs}$ often drops, in some cases, even below 0.8.

| $avg\,(n^p_{max})$ / Link Metric | Experiment using $d_{ps} = 20$ | Experiment using $d_{ps} = 30$ |
|---|---|---|
| OB-EWMA | 8.85 | 9 |
| $\text{WMEWMA}_{0.9}$ | 7.7 | 7.52 |

Table 6.3: Average transmission budget allocation on route $p = (5, 3, 8, 15)$ calculated by $r_{min}$-routing using different metrics during different experiments

Tab. 6.3 compares the average transmission budget allocation $avg\,(n^p_{max})$ for each link metric, where $n^p_{max}$ is the maximum number of transmissions on path $p$ as determined by $r_{min}$-routing. $n^p_{max}$ of path $p$ may vary during an experiment depending on the changes in the link quality of the links used by the path. The table shows that the results for experiments using PDR slot duration $d_{ps} = 20$s (Fig. 6.9) and $d_{ps} = 30$s (Fig. 6.10). As expected, using OBEWMA as the link metric results in a higher budget allocation. On average, 8.85 and 9 transmissions are distributed over three hops for experiments using a PDR slot of duration $d_{ps}$, 20s and 30s, respectively. Using $\text{WMEWMA}_{0.9}$ is results in a slightly lower budget allocation. However, the route often fails to meet the target reliability using $\text{WMEWMA}_{0.9}$ as the link metric.

## 6.6 Conclusion

This chapter introduced the link metric OB-EWMA[MG21] designed for reliability-constrained routing in contention-based wireless networks. OB-EWMA provides adaptive, conservative, and sufficiently stable link quality estimates. The performance of OB-EWMA is evaluated by analyzing link data collected from real experiments in a WiFi testbed. Results show that the proposed metric meets the original requirements. Further, we evaluated OB-EWMA as a link metric for $r_{min}$-routing, comparing it to another link metric WMEWMA. Based on the experimental findings, we can conclude that OB-EWMA is a suitable link metric for reliability-constrained routing, especially for $r_{min}$-routing, and it outperforms known metrics.

A drawback of the metric OB-EWMA is that it requires sufficient statistical data for its calculations, i.e., it is undefined until $n_{mw} < s_{mw}$. A temporary solution is to use an alternate metric in that phase, switch to OB-EWMA once sufficient data is collected, or delay the higher layer network operations until sufficient data is collected. The parameters used by OB-EWMA to select history weight, PDR slot

size, and moving window size varied to some degree between experiments. However, an arbitrary value for $p$ ($p = 0.1$) is selected for all links in our evaluation. It remains to be studied whether this is a suitable value for $p$ for all links. Further, it is an open question on how these parameters can be adapted dynamically to tune OB-EWMA during run-time.

# 7 Conclusions

## Contents

**7.1 Summary** . . . . . **196**
**7.2 Limitations and Future Works** . . . . . **198**

As the number of consumer electronics equipped with wireless communication technologies such as WiFi or Bluetooth is increasing, it is becoming challenging for wireless control networks built on top of IEEE 802.15.4 to guarantee sufficiently predictable behavior of the technology. The evolution of WiFi, notably, a significantly increased transmission speed and the availability of off-the-shelf hardwares with WiFi at a lower price created the opportunity to use WiFi networks for applications such as industrial automation. However, additional measures are needed to guarantee the specified statistical reliability required for industrial applications. In this thesis, we presented protocols and algorithms for reliability-constrained QoS routing in wireless networks operating in contention mode, such as WiFi. This chapter summarizes this thesis's findings, presents the conclusion, and presents potential future works.

## 7.1 Summary

Chapter 2 presents the novel routing approach, $r_{min}$-routing, which calculates bandwidth-efficient routes that satisfy a minimum specified statistical route reliability $r_{min}$. Together with other state-of-the-art routing metrics, $r_{min}$-routing uses the novel routing metric SMTX to calculate $r_{min}$ routes. Simulation experiments are conducted to exemplify the performance of $r_{min}$-routing using the SMTX metric. Experiment results demonstrate the advantages of the novel routing approach to meet the specified statistical route reliability and, notably, the quality of SMTX to calculate bandwidth-efficient routes by optimally distributing the transmission budget across the path. Nevertheless, the successful operation of $r_{min}$-routes in practice requires certain prerequisites, which are addressed in the following chapters.

Chapter 3 lays the foundations to explore the statistical link properties required to boost the link quality through retransmissions (called *link boosting*). Different methods were applied to the transmission outcomes on a link (modeled as Bernoulli sequences) to measure certain statistical link properties. We used the channel memory parameter from the Gilbert-Elliot model [Gil60; Ell63] to measure the independence between outcomes of successive transmission events. To understand the randomness of transmission event outcomes, we used the Wald-Wolfowitz Runs-test [Bra68]. Further, we used the "Chi-square test for goodness to fitness" to measure the fitness of observed Bernoulli trials to a Bernoulli distribution with success probability $p$, respectively. Based on the results, the transmission outcomes on a link are classified either as statistically independent, statistically dependent (bursty), or oscillatory. Results show that links in a collision-free environment are more likely

to have bursty or statistically dependent event outcomes. However, in the presence of foreign traffic, most of the links have statistically independent transmission outcomes depending on medium conditions and the characteristics of the traffic generated. When the medium load or traffic generated by the testbed increased during the experiments, the likelihood to observe bursty and non-random event outcomes increased.

Chapter 4 examines the necessity of certain statistical properties to boost the link quality successfully. The metric *target missed* was introduced to measure the effectiveness of link boosting, which denotes the difference between target reliability and observed boosted reliability on the link by performing ($n^e_{max}$ -1) retransmissions. Evaluation results show that the link boosting failed (i.e., $r^e_{\Delta,missed} > 0$), and the *target missed* increases as the event outcomes become statistically dependent. As expected, link boosting was successful for sub-sequences or windows with statistically independent or random event outcomes. Notably, link boosting was also successful for oscillatory sequences. Moreover, it was found that the link boosting was successful for certain moving windows with weaker statistical link properties. Exploring further, in those windows, the difference between boosted link reliability and minimum target reliability is high.

Chapter 5 presents the traffic shaping and monitoring metric used by the Medium Management Layer (MML) of the WiPS. Traffic shaping is conducted using the Time Token Bucket Algorithm (TTBA), and medium overload at nodes is monitored using hardware independent metric, derived from *unusable wasted tokens*. Later, we examined the influence of applications on statistical link properties and proposed the *dynamic refilling* mechanism for token refilling that reduce the influence.

Chapter 6 presents the novel link quality metric OB-EWMA. The metric provides adaptive, conservative, and sufficiently stable link quality estimates. Experiments results show that OB-EWMA is a suitable metric for reliability-constrained routing such as $r_{min}$-routing.

In conclusion, in this work, we laid the foundations for building a wireless control network using off-the-shelf IEEE 802.11 (WiFi) hardware operating in contention mode that can satisfy the specified reliability requirements. The novel routing methodology introduced in the thesis focuses on one of the drawbacks of wireless networks operating in contention mode, i.e., the problem of controlling the reliability of frame transmissions. Investigating the requirements for the routing protocol, we evaluated the statistical link properties required for a WiFi link to boost the reliability through retransmissions. As shown by the author in [Bia00], the frame collision probability in IEEE 802.11 networks is independent "when the number of

nodes competing for the medium is fairly large"; the evaluation of link data from the testbed network in the presence of external traffic compromises with it. However, transmission outcomes were either bursty or oscillatory under certain traffic conditions. Furthermore, the outcomes were more likely to be bursty or oscillatory in the absence of external traffic.

As observed, the statistical link properties may be affected during the operation. Therefore, it is necessary to monitor and control (or shape) the outgoing traffic of the control network. Since it is difficult to control the outflow of traffic from external devices [20], the remaining possibility is to monitor the statistical properties of the links and act accordingly. We introduced measures to evaluate the statistical link properties of the historical transmission events, monitor medium overload locally, and shape traffic flow from nodes. Further, we also introduced a link metric suitable for reliability-constrained routing.

## 7.2 Limitations and Future Works

Evaluation of $r_{min}$-routing and experiments using $r_{min}$-routing determined the $r_{min}$-routes at a centralized controller node with the global network information. Several other functionalities are required to implement the routing protocol in practice. In [KG21] and [KG22], the authors presented multiple potential solutions in this direction. For instance, a protocol to disseminate the link quality across the network such that the nodes have the latest information is required. Contrary to traditional flooding approaches, the protocol should consider minimizing the medium usage without compromising data (link quality information) dissemination speed. In [KG21], the authors present a novel protocol to measure and distribute the link quality information across the network. In [KG22], the authors present a functionally complete decentralized routing algorithm, $dR_{min}$-routing, which uses $r_{min}$-routes.

Chapter 4 presented metrics ($\mu$ and $z$-value) to measure statistical link properties necessary for link boosting. Further, we also presented a metric $r^{e}_{mosRequired,0.1}$ to estimate the required margin of safety to tolerate the weakness in statistical link properties to some extent, i.e., for $\mu < 1$. However, the findings are not yet incorporated into the $r_{min}$-route calculation process. One direction is to consider the historical statistical properties of the links while determining $r_{min}$-routes. Further, we only considered the statistical properties of transmission event outcomes

[20] It is possible to influence the external traffic to some extent using specialized hardwares, or by "hacks," for instance, by transmitting a WiFi frame with a higher NAV value to reserve the medium.

on a link, necessary for link boosting. However, for the consecutive process, i.e., multiplying the link reliabilities to measure the reliability of a path, the outcomes of subsequent transmissions across the path must be statistically independent (the outcome of an event must be independent of the previous event). Even though we introduced certain measures to reduce the dependencies of nodes by traffic shaping, it is necessary to measure whether a path holds this property. In [Sri+10], the authors proposed offline metrics to measure this correlation between links [21]. However, further research is required to develop a metric that can be applied at runtime.

Link quality estimation using the link metric presented in this thesis, OB-EWMA, depends on the configurable parameters to control the moving window size, PDR slot size, history weight, and the selected percentile value. We study the effect by changing parameters to an extent. Future research on the influence of select dynamic $p$-percentile values depending on the link during the operation may improve the estimation quality.

[21] The authors state it as "inter-link reception correlation".

# List of own publications


[KMG19a] C. Kohlstruck, K. Mathews, and R. Gotzhein. “rmin -Routing – Discovery and Operation of Routes in Wireless Ad-hoc Networks with Specified Statistical Minimum Reliabilities”. In: *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. 2019, pp. 430–437. DOI: 10.1109/LCN44214.2019.8990685.

[KMG19b] Christopher Kramer, Kiran Mathews, and Reinhard Gotzhein. “Cooperative Fair Bandwidth Scaling in Contention-based Wireless Networks using Time Token Bucket”. In: *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*. 2019, pp. 1–9. DOI: 10.1109/IPCCC47392.2019.8958764.

[MG21] Kiran Mathews and Reinhard Gotzhein. “OB-EWMA: A Link Metric for Reliability-constrained Routing in Wireless Networks”. In: *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. 2021, pp. 1–7. DOI: 10.1109/WCNC49053.2021.9417117.

[MKG17] Kiran Mathews, Christopher Kramer, and Reinhard Gotzhein. “Token bucket based traffic shaping and monitoring for WLAN-based control systems”. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. 2017, pp. 1–7. DOI: 10.1109/PIMRC.2017.8292201.

# Bibliography


[80216] IEEE Std 802.11. "IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". In: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (2016), pp. 1296–1297. DOI: 10.1109/IEEESTD.2016.7786995.

[IEE] IEEE Std 802.11. "IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". In: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (). DOI: 10.1109/IEEESTD.2016.7786995.

[Ach+08] Prashanth Aravinda Kumar Acharya et al. "Congestion-aware rate adaptation in wireless networks: A measurement-driven approach". In: *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON'08. 5th Annual IEEE Communications Society Conference on*. IEEE. 2008, pp. 1–9.

[Ali+11] Muhammad Hamad Alizai et al. "Efficient online estimation of bursty wireless links". In: *2011 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2011, pp. 191–198.

[Bac+10] Nouha Baccour et al. "F-lqe: A fuzzy link quality estimator for wireless sensor networks". In: *European Conference on Wireless Sensor Networks*. Springer. 2010, pp. 240–255.

[Bau+07] Rainer Baumann et al. "A survey on routing metrics". In: *TIK report* 262 (2007), pp. 1–53.

[Bia00] Giuseppe Bianchi. "Performance analysis of the IEEE 802.11 distributed coordination function". In: *IEEE Journal on selected areas in communications* 18.3 (2000), pp. 535–547.

[Bil13] Ana Bildea. "Link quality in wireless sensor networks". PhD thesis. Université de Grenoble, 2013.

[Bil+15] Ana Bildea et al. "Link quality estimation with the Gilbert-Elliot model for wireless sensor networks". In: *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE. 2015, pp. 2049–2054.

[BM04] Sanjit Biswas and Robert Morris. "Opportunistic routing in multi-hop wireless networks". In: *ACM SIGCOMM Computer Communication Review* 34.1 (2004), pp. 69–74.

[Bra68] James V Bradley. "Distribution-free statistical tests". In: (1968).

[Cam+08] Miguel Elias M Campista et al. "Routing metrics and protocols for wireless mesh networks". In: *IEEE network* 22.1 (2008), pp. 6–12.

[Cer+05] Alberto Cerpa et al. "Temporal properties of low power wireless links: modeling and implications on multi-hop routing". In: *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. 2005, pp. 414–425.

[CFG] CFG80211. *Linux Kernel, 802.11 device and configuration interface, line 915 - 930*. `https://elixir.bootlin.com/linux/v5.15.7/source/include/net/cfg80211.h`. Accessed: 08-12-2021.

[Com+10] International Electrotechnical Commission et al. "Industrial communication networks—wireless communication network and communication profiles—WirelessHART (IEC 62591 ed 1.0)". In: *Geneva, Switzerland, April* (2010).

[De +03] Douglas SJ De Couto et al. "A high-throughput path metric for multi-hop wireless routing". In: *Proceedings of the 9th annual international conference on Mobile computing and networking*. 2003, pp. 134–146.

[DBN03] Budhaditya Deb, Sudeept Bhatnagar, and Badri Nath. "ReInForM: Reliable information forwarding using multiple paths in sensor networks". In: *28th Annual IEEE International Conference on Local Computer Networks, 2003. LCN'03. Proceedings*. IEEE. 2003, pp. 406–415.

[DKS10] Peter Dely, Andreas J Kassler, and Dmitry Sivchenko. "Theoretical and experimental analysis of the channel busy fraction in IEEE 802.11". In: *Future Network and Mobile Summit, 2010*. IEEE. 2010, pp. 1–9.

[Dod08] Yadolah Dodge. "Bernoulli Distribution". In: *The Concise Encyclopedia of Statistics*. New York, NY: Springer New York, 2008, pp. 36–37. ISBN: 978-0-387-32833-1. DOI: 10.1007/978-0-387-32833-1_24. URL: https://doi.org/10.1007/978-0-387-32833-1_24.

[Dua+09] Zhi-feng Duan et al. "Shortest Path Routing Protocol for Multi-layer Mobile Wireless Sensor Networks". In: *2009 International Conference on Networks Security, Wireless Communications and Trusted Computing*. Vol. 2. 2009, pp. 106–110. DOI: 10.1109/NSWCTC.2009.282.

[Ell63] Edwin O Elliott. "Estimates of error rates for codes on burst-noise channels". In: *The Bell System Technical Journal* 42.5 (1963), pp. 1977–1997.

[Erg04] Sinem Coleri Ergen. "ZigBee/IEEE 802.15. 4 Summary". In: *UC Berkeley, September* 10.17 (2004), p. 11.

[FLE06] Emad Felemban, Chang-Gun Lee, and Eylem Ekici. "MMSPEED: multipath Multi-SPEED protocol for QoS guarantee of reliability and. Timeliness in wireless sensor networks". In: *IEEE transactions on mobile computing* 5.6 (2006), pp. 738–754.

[Fon+07] Rodrigo Fonseca et al. "Four-bit wireless link estimation." In: *HotNets*. 2007.

[FM14] Nancy Forbes and Basil Mahon. *Faraday, Maxwell, and the electromagnetic field: How two men revolutionized physics*. Illustrated edition. Prometheus Books, 2014. ISBN: 1616149426.

[Gay14] Warren Gay. *Raspberry Pi Hardware Reference*. 1st. USA: Apress, 2014. ISBN: 1484208005.

[Gil60] Edgar N Gilbert. "Capacity of a burst-noise channel". In: *Bell system technical journal* 39.5 (1960), pp. 1253–1265.

[Got20] Reinhard Gotzhein. "ProNet 4.0 Case Studies". In: *Real-time Communication Protocols for Multi-hop Ad-hoc Networks*. Springer, 2020, pp. 249–263.

[Gra03] Benjamin Graham. *The Intelligent Investor Rev Ed*. 2003.

[Guo+14] Jianlin Guo et al. "Reliable routing in large scale wireless sensor networks". In: *2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE. 2014, pp. 99–104.

[HH08] Gerhard Haßlinger and Oliver Hohlfeld. "The Gilbert-Elliott model for packet loss in real time services on the Internet". In: *14th GI/ITG Conference-Measurement, Modelling and Evalutation of Computer and Communication Systems*. VDE. 2008, pp. 1–15.

[He+03] Tian He et al. “SPEED: A stateless protocol for real-time communication in sensor networks”. In: *23rd International Conference on Distributed Computing Systems, 2003. Proceedings.* IEEE. 2003, pp. 46–55.

[Her19] Florian Herrmann. “Reliable Protocol for Collecting Global Link Quality Information of 802.11 based WNCS”. Bachelors Thesis. 2019.

[Hum77] Paul W Humphreys. “Randomness, independence, and hypotheses”. In: *Synthese* (1977), pp. 415–426.

[JM09] Van Jacobson and S McCanne. “libpcap: Packet capture library”. In: *Lawrence Berkeley Laboratory, Berkeley, CA* (2009).

[JS16] Payal Jadhav and Rachna Satao. “A survey on opportunistic routing protocols for wireless sensor networks”. In: *Procedia Computer Science* 79 (2016), pp. 603–609.

[Jak+11] Gentian Jakllari et al. “Link positions matter: A noncommutative routing metric for wireless mesh networks”. In: *IEEE Transactions on Mobile Computing* 11.1 (2011), pp. 61–72.

[Jar+05] Amit P Jardosh et al. “Understanding congestion in IEEE 802.11 b wireless networks”. In: *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. USENIX Association. 2005, pp. 25–25.

[JM96] David B Johnson and David A Maltz. “Dynamic source routing in ad hoc wireless networks”. In: *Mobile computing*. Springer, 1996, pp. 153–181.

[Kho+18] Evgeny Khorov et al. “A tutorial on IEEE 802.11 ax high efficiency WLANs”. In: *IEEE Communications Surveys & Tutorials* 21.1 (2018), pp. 197–216.

[Kim+19] Sangdae Kim et al. “Opportunistic multipath routing in long-hop wireless sensor networks”. In: *Sensors* 19.19 (2019), p. 4072.

[KMG19a] C. Kohlstruck, K. Mathews, and R. Gotzhein. “rmin -Routing – Discovery and Operation of Routes in Wireless Ad-hoc Networks with Specified Statistical Minimum Reliabilities”. In: *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. 2019, pp. 430–437. DOI: 10.1109/LCN44214.2019.8990685.

[Koh18] Christopher Kohlstruck. “QoS Routing – Discovery and Assessment of Routes with Statistical Minimum Reliability”. Technische Universität Kaiserslautern. MA thesis. 2018.

[KG21] Christopher Kohlstruck and Reinhard Gotzhein. “cTEx–A Configurable Topology Explorer for Wireless Ad-hoc Networks”. In: *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE. 2021, pp. 1–9.

[KG22] Christopher Kohlstruck and Reinhard Gotzhein. “dRmin-Routing – A Decentralized Algorithm for Reliability-constrained Routing in Wireless Ad-hoc Networks”. In: *International Wireless Communications and Mobile Computing Conference (IWCMC 2022), held on May 30 - June 3, 2022 in Dubrovnik, Croatia* (2022), "to appear".

[Kra21] Christopher Kramer. “Protokolle und Algorithmen für zuverlässige drahtlose Kommunikationssysteme”. In: (2021).

[KMG19b] Christopher Kramer, Kiran Mathews, and Reinhard Gotzhein. “Cooperative Fair Bandwidth Scaling in Contention-based Wireless Networks using Time Token Bucket”. In: *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*. 2019, pp. 1–9. DOI: 10.1109/IPCCC47392.2019.8958764.

[LXL19] Wei Liu, Yu Xia, and Rong Luo. “FaLQE: Fluctuation adaptive link quality estimator for wireless sensor networks”. In: *China Conference on Wireless Sensor Networks*. Springer. 2019, pp. 48–60.

[Mar09] Guglielmo Marconi. “Wireless telegraphic communication”. In: *Nobel Lecture* 11 (1909), pp. 198–222.

[MG21] Kiran Mathews and Reinhard Gotzhein. “OB-EWMA: A Link Metric for Reliability-constrained Routing in Wireless Networks”. In: *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. 2021, pp. 1–7. DOI: 10.1109/WCNC49053.2021.9417117.

[MKG17] Kiran Mathews, Christopher Kramer, and Reinhard Gotzhein. “Token bucket based traffic shaping and monitoring for WLAN-based control systems”. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. 2017, pp. 1–7. DOI: 10.1109/PIMRC.2017.8292201.

[Max73] James Clerk Maxwell. *A treatise on electricity and magnetism*. Vol. 1. Clarendon press, 1873.

[Mil14] Jeremy Miles. "R Squared, Adjusted R Squared". In: *Wiley StatsRef: Statistics Reference Online*. American Cancer Society, 2014. ISBN: 9781118445112. DOI: https://doi.org/10.1002/9781118445112.stat06627. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118445112.stat06627. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118445112.stat06627.

[MF07] Anelise Munaretto and Mauro Fonseca. "Routing and quality of service support for mobile ad hoc networks". In: *Computer Networks* 51.11 (2007), pp. 3142–3156.

[NET] NETLINK. *Linux Kernel, 802.11 netlink interface public header, line 4156 - 4185*. https://elixir.bootlin.com/linux/v5.15.7/source/include/uapi/linux/nl80211.h. Accessed: 08-12-2021.

[Pas+06] Diego Passos et al. "Mesh network performance measurements". In: *International Information and Telecommunicatios Technologies Symposium (I2TS)*. 2006, p. 8.

[PCE] PCEngines. *APU2*.

[PB94] Charles E Perkins and Pravin Bhagwat. "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers". In: *ACM SIGCOMM computer communication review* 24.4 (1994), pp. 234–244.

[PR99] Charles E Perkins and Elizabeth M Royer. "Ad-hoc on-demand distance vector routing". In: *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*. IEEE. 1999, pp. 90–100.

[R C17] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2017. URL: https://www.R-project.org/.

[Rab16] Marc Raboy. *Marconi: The man who networked the world*. Oxford University Press, 2016.

[Rad] Radiotap. URL: http://www.radiotap.org/.

[RH10] George F Riley and Thomas R Henderson. "The ns-3 network simulator". In: *Modeling and tools for network simulation*. Springer, 2010, pp. 15–34.

[Sch+17] Matthias Schulz et al. “Massive reactive smartphone-based jamming using arbitrary waveforms and adaptive power control”. In: *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 2017, pp. 111–121.

[SAB08] Irfan Sheriff, Prashanth Aravinda Kumar Acharya, and Elizabeth M Belding. “Measurement-driven admission control on wireless backhaul networks”. In: *Computer Communications* 31.7 (2008), pp. 1354–1371.

[Sri10] Kannan Srinivasan. *Towards a wireless lexicon*. Stanford University, 2010.

[Sri+10] Kannan Srinivasan et al. “The $\kappa$ factor: inferring protocol performance using inter-link reception correlation”. In: *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. 2010, pp. 317–328.

[Sri+08] Kannan Srinivasan et al. “The $\beta$-factor: measuring wireless link burstiness”. In: *Proceedings of the 6th ACM conference on Embedded network sensor systems*. 2008, pp. 29–42.

[SR12] Mirko Stoffers and George Riley. “Comparing the ns-3 propagation models”. In: *2012 IEEE 20th international symposium on modeling, analysis and simulation of computer and telecommunication systems*. IEEE. 2012, pp. 61–67.

[TG04] Godfrey Tan and John V Guttag. “Time-based Fairness Improves Performance in Multi-Rate WLANs.” In: *USENIX annual technical conference, general track*. 2004, pp. 269–282.

[Tan03] Andrew S. Tanenbaum. *Computer Networks, Third edition*. Reading, Massachusetts: Prentice Hall PTR, 2003, pp. 380–384.

[Teg20] Christian Tegeler. *Monitoring and Influencing "Link Quality" Correlation Between Links*. 2020.

[Tur86] J. Turner. “New directions in communications (or which way to the information age?)” In: *IEEE Communications Magazine* 24.10 (1986), pp. 8–15. DOI: 10.1109/MCOM.1986.1092946.

[Vla+08] Angelos Vlavianos et al. “Assessing link quality in IEEE 802.11 wireless networks: Which is the right metric?” In: *2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE. 2008, pp. 1–6.

[Wen+07] Hao Wen et al. "Retransmission or redundancy: Transmission reliability in wireless sensor networks". In: *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*. IEEE. 2007, pp. 1–7.

[WC03] Alec Woo and David E Culler. *Evaluation of efficient link reliability estimators for low-power wireless networks*. Computer Science Division, University of California Oakland, CA, USA, 2003.

[WTC03] Alec Woo, Terence Tong, and David Culler. "Taming the underlying challenges of reliable multihop routing in sensor networks". In: *Proceedings of the 1st international conference on Embedded networked sensor systems*. 2003, pp. 14–27.

[XG03] Qi Xue and Aura Ganz. "Ad hoc QoS on-demand routing (AQOR) in mobile ad hoc networks". In: *Journal of parallel and distributed computing* 63.2 (2003), pp. 154–165.

[Yaj+99] Maya Yajnik et al. "Measurement and modelling of the temporal dependence in packet loss". In: *IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*. Vol. 1. IEEE. 1999, pp. 345–352.

[YYL11] Shengbo Yang, Chai Kiat Yeo, and Bu-Sung Lee. "Toward reliable data delivery for highly dynamic mobile ad hoc networks". In: *IEEE transactions on mobile computing* 11.1 (2011), pp. 111–124.

[Yen70] Jin Y Yen. "An algorithm for finding shortest routes from all source nodes to a given destination in general networks". In: *Quarterly of Applied Mathematics* 27.4 (1970), pp. 526–530.

[Yu+07] Ming Yu et al. "A link availability-based QoS-aware routing protocol for mobile ad hoc sensor networks". In: *Computer Communications* 30.18 (2007), pp. 3823–3831.

[Zhe+09] Yangbo Zheng et al. "Industrial wireless deterministic communication based on WLAN: Design, implementation and analysis". In: *2009 IEEE International Conference on Communications Technology and Applications*. 2009, pp. 274–278. DOI: 10.1109/ICCOMTA.2009.5349194.

[Zon+14] Amir Ehsani Zonouz et al. "Reliability-oriented single-path routing protocols in wireless sensor networks". In: *IEEE Sensors Journal* 14.11 (2014), pp. 4059–4068.

# Kiran Mathews
Curriculum Vitae

## EDUCATION

**Doctoral Student** 2017 - 2022
*Technische Universität Kaiserslautern*

**Master of Science (M.Sc)** 2014 - 2017
*Technische Universität Kaiserslautern*
Concentration: Computer Science
Specialization: Communication Systems
Minor: Electrical engineering
Title of the master's thesis: "Network Sensing in WLAN"

**Bachelor of Technology (B.Tech)** 2009 - 2013
*Cochin University of Science and Technology (CUSAT)*
Concentration: Computer Science & Engineering
Title of the bachelor's main project: "CYBERMATE"

**Secondary education** 2007 - 2009
*Vimala Public School, Thodupuzha, Kerala, India*
Affiliated to: CBSE - Central Board of Secondary Education
Concentration: Computer Science

**Primary education** Mar. 2007
*Vimala Public School, Thodupuzha, Kerala, India*
Affiliated to: CBSE - Central Board of Secondary Education

## WORK EXPERIENCE

**Wissenschaftlicher Mitarbeiter** Apr. 2019 - Sept. 2022
*Networked Systems Group*, *Technische Universität Kaiserslautern*

**Graduate Student Research Assistant** Feb. 2017 - Apr. 2019
*Networked Systems Group*, *Technische Universität Kaiserslautern*

**Undergraduate Research Assistant** Nov. 2014 - Feb. 2017
*Networked Systems Group*, *Technische Universität Kaiserslautern*

**Software Engineering Trainee** Apr. 2013 - Jul. 2013
*SectorQube, Inc.,* *Kochi, Kerala*