UNIVERSIDAD DE MÁLAGA
Escuela Técnica Superior de Ingeniería Informática
Programa de Doctorado en Tecnologías Informáticas

UNIVERSIDAD DE MÁLAGA

Model Learning for Time-Sensitive Networking over 5G networks

TESIS DOCTORAL

Francisco José Luque Schempp

Málaga, Julio 2024

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

AUTOR: Francisco José Luque Schempp

http://orcid.org/0000-0001-8389-5660

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial-SinObraDerivada 4.0 Internacional:
https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode
Cualquier parte de esta obra se puede reproducir sin autorización
pero con el reconocimiento y atribución de los autores.
No se puede hacer uso comercial de la obra y no se puede alterar, transformar o hacer obras derivadas.

Esta Tesis Doctoral está depositada en el Repositorio Institucional de la Universidad de Málaga (RIUMA): riuma.uma.es

UNIVERSIDAD
DE MÁLAGA

UNIVERSIDAD DE MÁLAGA
Escuela Técnica Superior de Ingeniería Informática
Programa de Doctorado en Tecnologías Informáticas

Model Learning for Time-Sensitive Networking over 5G networks

Thesis author:
Francisco José Luque Schempp

Thesis supervisors:
Pedro Merino Gómez
María del Mar Gallardo Melgarejo

Málaga, Julio 2024

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD DE LA TESIS PRESENTADA PARA OBTENER EL TÍTULO DE DOCTOR

D./Dña FRANCISCO JOSÉ LUQUE SCHEMPP
Estudiante del programa de doctorado TECNOLOGÍAS INFORMÁTICAS de la Universidad de Málaga, autor/a de la tesis, presentada para la obtención del título de doctor por la Universidad de Málaga, titulada: MODEL LEARNING FOR TIME-SENSITIVE NETWORKING OVER 5G NETWORKS

Realizada bajo la tutorización de PEDRO MERINO GÓMEZ y dirección de MARÍA DEL MAR GALLARDO MELGAREJO Y PEDRO MERINO GÓMEZ (si tuviera varios directores deberá hacer constar el nombre de todos)

DECLARO QUE:

La tesis presentada es una obra original que no infringe los derechos de propiedad intelectual ni los derechos de propiedad industrial u otros, conforme al ordenamiento jurídico vigente (Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia), modificado por la Ley 2/2019, de 1 de marzo.

Igualmente asumo, ante a la Universidad de Málaga y ante cualquier otra instancia, la responsabilidad que pudiera derivarse en caso de plagio de contenidos en la tesis presentada, conforme al ordenamiento jurídico vigente.

En Málaga, a 22 de MAYO de 2024

Fdo.: Doctorando/a	Fdo.: Tutor/a
Fdo.: Director/es de tesis	

Edificio Pabellón de Gobierno. Campus El Ejido. 29071
Tel.: 952 13 10 28 / 952 13 14 61 / 952 13 71 10
E-mail: doctorado@uma.es

UNIVERSIDAD
DE MÁLAGA

Por la presente, Dra. María del Mar Gallardo Melgarejo y Dr. Pedro Merino Gómez, profesores del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, CERTIFICAN:

Que Francisco José Luque Schempp ha realizado en el Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga bajo su dirección, el trabajo de investigación correspondiente a su TESIS DOCTORAL titulada:

"Model Learning for Time-Sensitive Networking over 5G Networks"

En dicho trabajo se han propuesto aportaciones originales en el campo de aprendizaje basado en modelos para Time-Sensitive Networking (TSN) sobre redes 5G. Los resultados expuestos han dado lugar a las siguientes publicaciones en revistas:

1. F. Luque-Schempp, L. Panizo, M. -d. -M. Gallardo, P. Merino and J. Rivas, "Toward Zero Touch Configuration of 5G Non-Public Networks for Time Sensitive Networking," in IEEE Network, vol. 36, no. 2, pp. 50-56, March/April 2022, doi: 10.1109/MNET.006.2100442.

2. F. Luque-Schempp, L. Panizo, M. -D. -M. Gallardo and P. Merino, "AutomAdapt: Zero Touch Configuration of 5G QoS Flows Extended for Time-Sensitive Networking," in IEEE Access, vol. 11, pp. 82960-82977, 2023, doi: 10.1109/ACCESS.2023.3302264.

Estas publicaciones en coautoría que avalan la tesis no han sido utilizadas en tesis anteriores. Por todo ello, consideran que esta Tesis es apta para su presentación al Tribunal que ha de juzgarla. Y para que conste a efectos de lo establecido, AUTORIZA la presentación de esta Tesis en la Universidad de Málaga.

En Málaga, a 22 de mayo de 2024.

Fdo.: Dra. María del Mar Gallardo Melgarejo	Fdo.: Dr. Pedro Merino Gómez

UNIVERSIDAD
DE MÁLAGA

"The only way to do great work is to love what you do."
(Steve Jobs)

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisors, Dr. Pedro Merino and Dra. María del Mar Gallardo, for their invaluable guidance, support, and encouragement throughout my research career, which has been instrumental in shaping this thesis. I am also immensely grateful to my colleagues, who have been a constant source of support. Special thanks to Laura and Delia for their help, collaboration, and the many stimulating discussions we have shared. I also would like to thank the people who made me feel at home during my stay at inIT, Technische Hochschule OWL.

I have no words to express my gratitude to my family for their support and love. My parents, Gloria and Francisco, and sisters, Lola y Lidia, have always believed in me and encouraged me to pursue my dreams. I also extend my heartfelt thanks to my friends who have provided me with emotional support and much needed breaks from my work. Finally, María has accompanied me along the way, supporting and encouraging me to overcome the many challenges I have faced, for which I will forever be grateful.

Thank you for being an integral part of this journey. This thesis was only possible with all of you.

This thesis has been supported by the Ministry of Science, Innovation and Universities of Spain under grant agreement PRE2019-089212 and by the RFOG project under grant agreement RTI2018-099777-B-I00.

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Abstract

The aim of deterministic networks is to implement deterministic communications for real-time applications, such as remote control or industrial automation, which implies limited latency, low data loss rates, and jitter. Specifically, Time-Sensitive Networking (TSN) emerges to encompass the characteristics demanded by the market, enabling the transmission of time-sensitive data over Ethernet networks. Due to the growing demand to bring the advantages of wireless communication, such as mobility or wiring avoidance, to industrial environments, the trend is to use a hybrid approach, combining wired and wireless technologies. 5G is emerging as one of the main technologies that offers high reliability and low latency. 3GPP is pushing to define an architecture in which the 5G network can interact with the TSN network, for which new entities, such as TSN translators or TSN Application Function, are needed to support interoperability. However, automatic network configuration is critical to meet the requirements of TSN traffic. For this purpose, approaches such as Zero-Touch are born with the objective of automating and simplifying processes to minimize or even eliminate human intervention in network operations. In addition, the use of learning techniques can help in these automatic network reconfigurations.

The objective of this thesis is to build a 5G network behavior model which can be used to monitor the quality of end-to-end traffic sessions. For this purpose, the 5G network model is built using automata learning, where network behavior is learned based on the resulting KPIs when using different traffic profiles with a variety of 5G network configurations. Following a closed-loop approach, it is possible to use the learned automata to make predictions of possible deviations from the KPIs associated with TSN traffic, and therefore suggest suitable network configurations to continue to meet the traffic requirements, which will be applied automatically.

In this context, this thesis presents an incremental development. First, the design of a learning algorithm that allows one to represent an abstract model of the network behavior, including the resulting traffic Key Performance Indicators and network configuration. Second, the extension of the learning algorithm by defining a new type of time-oriented automata, namely Traffic-Oriented input/output Automata, capable of representing the evolution of the TSN session in relation to time. This enables interaction with the system to make predictions of possible deviations, which derive in auto-reconfigurations to continue to meet the requirements. Third, design of the prediction and reconfiguration algorithm.

Furthermore, this thesis produces a tool that includes the monitoring, learning, prediction, and reconfiguration modules called AutomAdapt, which implements the algorithms mentioned above designed to pursue the main objective of the thesis. Furthermore, AutomAdapt has been extensively evaluated in the TSN over 5G testbed available at the University of Malaga, where it is shown to be able to monitor and learn from the network in real time, allowing predictions of possible future deviations and reconfiguring

UNIVERSIDAD DE MÁLAGA

the 5G network automatically, improving performance by up to 35-45% in terms of time the network remains in the desired state, that is, meeting requirements of the TSN traffic session.

In conclusion, the results obtained demonstrate that the 5G network is capable of meeting the requirements of a considerable part of the existing industrial traffic patterns. Furthermore, it is vital to use approaches such as Zero-Touch together with learning techniques such as learning automata to avoid human interaction in complex tasks such as network configuration in order to meet traffic requirements. In addition, the explainability offered by automata learning makes it possible to better understand the behavior of the network, allowing one to know the evolution of the network over time and why certain actions have been taken. Therefore, operational efficiency can be considerably improved by automating network management.

UNIVERSIDAD DE MÁLAGA

Resumen

El propósito de las redes deterministas es implementar comunicaciones deterministas para aplicaciones en tiempo real, como el control remoto o la automatización industrial, lo que implica latencia acotada, bajas tasas de pérdida de datos y jitter. Específicamente, la Time-Sensitive Networking (TSN) surge para abarcar las características demandadas por el mercado, permitiendo la transmisión de datos sensibles al tiempo a través de redes Ethernet. Debido a la creciente demanda de llevar las ventajas de la comunicación inalámbrica, como la movilidad o la evitación de cableado, a entornos industriales, la tendencia es utilizar un enfoque híbrido, combinando tecnologías cableadas e inalámbricas. 5G emerge como una de las principales tecnologías que ofrece alta fiabilidad y baja latencia. Adicionalmente, 3GPP está impulsando para definir una arquitectura en la que la red 5G pueda interactuar con la red TSN, para lo cual se necesitan nuevas entidades, como traductores TSN o TSN Application Function, para apoyar la interoperabilidad. Sin embargo, la configuración automática de la red es crítica para cumplir con los requisitos del tráfico TSN. Para este propósito, nacen enfoques como Zero-Touch con el objetivo de automatizar y simplificar procesos para minimizar o incluso eliminar la intervención humana en las operaciones de red. Además, el uso de técnicas de aprendizaje puede ayudar en estas reconfiguraciones automáticas de la red.

El objetivo de esta tesis es construir un modelo de comportamiento de la red 5G que pueda usarse para monitorizar la calidad de las sesiones de tráfico de extremo a extremo. Con este propósito, el modelo de red 5G se construye utilizando autómatas aprendizaje, donde el comportamiento de la red se aprende en base a los KPI resultantes al usar diferentes perfiles de tráfico con una variedad de configuraciones de red 5G. Siguiendo un enfoque de bucle cerrado, es posible usar los autómatas aprendidos para hacer predicciones de posibles desviaciones de los KPI asociados con el tráfico TSN, y por lo tanto sugerir configuraciones de red adecuadas para continuar cumpliendo con los requisitos del tráfico, que se aplicarán automáticamente.

En este contexto, esta tesis presenta un desarrollo incremental. Primero, el diseño de un algoritmo de aprendizaje que permite representar un modelo abstracto del comportamiento de la red, incluyendo los indicadores clave de rendimiento del tráfico resultante y la configuración de la red. Segundo, la extensión del algoritmo de aprendizaje definiendo un nuevo tipo de autómata orientado al tiempo, concretamente, Traffic-Oriented input/output Automata, capaz de representar la evolución de la sesión TSN en relación al tiempo. Esto permite la interacción con el sistema para hacer predicciones de posibles desviaciones, que derivan en reconfiguraciones automáticas para continuar cumpliendo con los requisitos. Tercero, diseño del algoritmo de predicción y reconfiguración.

Además, esta tesis produce una herramienta que incluye los módulos de monitorización, aprendizaje, predicción y reconfiguración llamada AutomAdapt, que implementa los algoritmos mencionados anteriormente diseñados para perseguir el objetivo principal de la tesis. Además, AutomAdapt ha sido evaluado de forma extensa en el testbed

UNIVERSIDAD DE MÁLAGA

TSN sobre 5G disponible en la Universidad de Málaga, donde se ha demostrado que es capaz de monitorizar y aprender de la red en tiempo real, permitiendo predicciones de posibles desviaciones futuras y reconfigurando la red 5G automáticamente, mejorando el rendimiento hasta en un 35-45% en términos de tiempo que la red permanece en el estado deseado, es decir, cumpliendo con los requisitos de la sesión de tráfico TSN.

En conclusión, los resultados obtenidos demuestran que la red 5G es capaz de satisfacer los requisitos de una parte considerable de los patrones de tráfico industrial existentes. Además, es vital utilizar enfoques como Zero-Touch junto con técnicas de aprendizaje como el aprendizaje de autómatas para evitar la interacción humana en tareas complejas como la configuración de la red con el fin de satisfacer los requisitos de tráfico. Además, la explicabilidad que ofrece el aprendizaje de autómatas permite comprender mejor el comportamiento de la red, permitiendo conocer su evolución en el tiempo y el porqué de determinadas acciones. Por lo tanto, la eficacia operativa puede mejorar considerablemente automatizando la gestión de la red.

UNIVERSIDAD DE MÁLAGA

Contents

Acknowledgments . . . III
Abstract . . . V
Resumen . . . VII
List of Figures . . . XIV
List of Tables . . . XVI
Acronyms . . . XX

1 Introduction **1**
1.1 Motivation . . . 2
1.1.1 Time-Sensitive Networking . . . 2
1.1.2 5G Network . . . 5
5G Architecture . . . 6
Key Technologies . . . 8
1.1.3 Time-Sensitive Networking over 5G . . . 9
1.2 Research Outline . . . 10
1.2.1 Research Questions . . . 11
1.2.2 Methodology . . . 11
Main Objectives . . . 12
Secondary Objectives . . . 13
1.2.3 Contributions . . . 13
1.3 Document Structure . . . 16

2 Fundamentals and State of the Art **19**
2.1 Advanced Networking Technologies . . . 21
2.1.1 Industrial Communications standards and protocols . . . 21
2.1.2 Deterministic Networking and Time-Sensitive Networking . . . 22
TSN - Traffic Scheduling . . . 24
TSN - Time Synchronization . . . 25
2.1.3 Hybrid Networks for Industry . . . 26
TSN over 5G integration . . . 27
TSN over 5G - Time Synchronization . . . 29
TSN over 5G - TSN translators . . . 29

UNIVERSIDAD DE MÁLAGA

2.1.4 Testbeds . . . 30
2.2 Enabling Techniques and Methods for Reconfiguration . . . 31
2.2.1 Self-Organizing Networks . . . 31
2.2.2 Closed Control Loops . . . 32
2.2.3 Intent-Based Networking . . . 33
2.2.4 Zero-Touch Networks . . . 34
2.2.5 Network reconfiguration . . . 36
2.3 Learning Approaches . . . 38
2.3.1 Behavioral Modeling . . . 38
2.3.2 Artificial Intelligence and Machine Learning . . . 39
2.3.3 Model Learning . . . 40
2.4 Evaluation and Challenges . . . 41
2.4.1 Networking Technologies Integration . . . 42
2.4.2 Combined Reconfiguration Methods . . . 43
2.4.3 Application of Learning Techniques . . . 44
2.5 Analysis and Promising Research Topics . . . 45

3 Toward TSN over 5G Testbed Implementation 49
3.1 Testbed Components . . . 50
3.1.1 TSN network components . . . 51
3.1.2 5G network components . . . 51
Option #1 - Nokia RAN and Open5GS core network . . . 53
Option #2 - Keysight UXM 5G . . . 53
3.1.3 TSN over 5G network components . . . 54
Time-Sensitive Networking Translators . . . 54
Time-Sensitive Networking Application Function . . . 55
3.2 Challenges . . . 55
3.2.1 Time synchronization . . . 55
3.2.2 Traffic translation . . . 56
3.3 Relevant 5G Network Parameters . . . 58
3.3.1 5G Core Network parameters . . . 58
3.3.2 5G Radio Network parameters . . . 60
3.4 Key Findings . . . 64

4 Model Learning Approach 67
4.1 Overall Approach . . . 68
4.2 Phase 1. Passive Automata Learning Approach . . . 69
4.2.1 Capturing Network Traces . . . 69
4.2.2 Construction of Learned Automaton . . . 70
4.3 Phase 2. Traffic-Oriented Input/Output Automata . . . 74
4.3.1 Real-time Network Traces . . . 74
4.3.2 Traffic-Oriented Input/Output Automata . . . 76
4.3.3 Construction of Learned Automata . . . 80
4.3.4 Automata for Prediction and Reconfiguration . . . 84

UNIVERSIDAD DE MÁLAGA

5 Implementation and Evaluation of the Proposal **89**
5.1 Proposal Implementation . . . 90
5.1.1 Monitoring . . . 90
5.1.2 Learning . . . 96
Initial Learning Phase . . . 97
Online Learning . . . 98
5.1.3 Prediction and Reconfiguration . . . 99
Nokia RAN . . . 102
Keysight UXM 5G . . . 103
5.2 Evaluation . . . 105
5.2.1 Evaluation of the *Learn* algorithm . . . 107
5.2.2 Evaluation of the AutomAdapt Initial Learning Phase . . . 110
5.2.3 Evaluation of the Complete AutomAdapt Operation Cycle . . . 111
5.2.4 Example Trace . . . 113
5.2.5 Analysis of Results . . . 116

6 Conclusions and Future Work **121**
6.1 Conclusions . . . 122
6.2 Future Work . . . 124
6.3 Publications and Projects . . . 125
6.3.1 Journals . . . 125
6.3.2 Other Activities . . . 125
6.3.3 Related Projects and Funding . . . 126
6.3.4 Research Stay . . . 126

Bibliography **127**

A Resumen en Español **149**
A.1 Introducción . . . 149
A.1.1 Motivación . . . 149
A.1.2 Preguntas de Investigación y Objetivos . . . 150
Objetivos principales . . . 150
A.1.3 Objetivos secundarios . . . 151
A.1.4 Contribuciones . . . 151
A.2 Antecedentes . . . 153
A.2.1 Tecnologías Avanzadas de Redes . . . 153
A.2.2 Técnicas y Métodos para Reconfiguración . . . 155
A.2.3 Métodos de aprendizaje . . . 156
A.3 Implementación del Testbed TSN sobre 5G . . . 157
A.3.1 Componentes del Testbed . . . 157
A.3.2 Retos y Parámetros 5G Relevantes . . . 158
A.3.3 Hallazgos clave . . . 161
A.4 Construcción Automática de Modelos con Autómatas de Aprendizaje . . . 162
A.4.1 Fase 1. Autómata de Aprendizaje Pasivo . . . 163

UNIVERSIDAD DE MÁLAGA

A.4.2 Fase 2. Autómata de Entrada/Salida Orientado a Tráfico 164
A.5 Implementación y Evaluación de la Propuesta 166
A.5.1 Implementación 166
A.5.2 Evaluación 168
A.5.3 Análisis de los Resultados 174
A.6 Conclusiones y Trabajo Futuro 175

UNIVERSIDAD DE MÁLAGA

List of Figures

1.1 Fully centralized TSN network architecture - Copyright © 2022, IEEE 802.1Qcc [9] . . . 4
1.2 System Architecture Evolution: 4G, 5G NSA and 5G SA . . . 6
1.3 5G System Architecture (source [26]) . . . 8
1.4 5G Service Categories and KPIs . . . 9
1.5 Time-Sensitive Networking over 5G architecture . . . 10
1.6 QoS request and automatic reconfiguration of the network . . . 11
1.7 Dissertation timeline . . . 12
1.8 Relationship between research questions, objectives and contributions . . . 13

3.1 TSN over 5G testbed at the University of Malaga . . . 50
3.2 Relyum equipment IEEE 802.1Qbv Time-Aware Shaper configuration . . . 52
3.3 Nokia AirScale Pico RRH 5G . . . 53
3.4 Nokia AirScale Micro RRH 5G . . . 53
3.5 Keysight E7515B UXM 5G wireless test solution . . . 54
3.6 Proposed solution for time synchronization and traffic translation . . . 56
3.7 ADVA FSP 150 EMS configurations . . . 57
3.8 5G Quality of Service Architecture . . . 59
3.9 5G Network Protocol Stack - Control Plane and User Plane . . . 61
3.10 QoS Flow mapping within a PDU session on DRBs . . . 64

4.1 Overall approach . . . 68
4.2 Example of passive learned automaton . . . 72
4.3 Example of learned TOA . . . 78

5.1 Proposal implementation architecture . . . 90
5.2 Monitoring module implementation . . . 91
5.3 Delay and Jitter graphs plotted in InfluxDB . . . 94
5.4 RLC AM downlink parameters in InfluxDB . . . 95
5.5 Learning module implementation . . . 96
5.6 Learning module flow chart . . . 99
5.7 Nokia Web Element Manager . . . 102
5.8 Keysight UXM 5G Test Application . . . 104

UNIVERSIDAD DE MÁLAGA

5.9 Keysight UXM 5G Test Automation Platform 104
5.10 RRC message sent to the UE after a network reconfiguration 105
5.11 Evolution of the number of states and creation time of automata created with different cycle time values. 107
5.12 Sequence coverage - restrictive 108
5.13 Sequence coverage - less restrictive 109
5.14 Sequence coverage heatmap: percentage of sequences accepted by each automaton 109
5.15 Transition type and *qty* values coverage heatmap 110
5.16 AutomAdapt impact: Elapsed time in desired network state 113
5.17 Example network trace: Observed KPIs during the initial learning phase . 114
5.18 Example network trace: Observed KPIs during the complete operation cycle 115
5.19 Example network trace: Relationship between the observed KPIs of the two configurations used. 116
5.20 Relationship between RLC AM 5G RLC network parameters 119

A.1 Relación entre las preguntas de la investigación, objetivos y contribuciones 151
A.2 Arquitectura de red TSN totalmente centralizada - IEEE 802.1Qcc [9] . . 154
A.3 Evolución de la arquitectura: 4G, 5G NSA y 5G SA 155
A.4 Arquitectura Time-Sensitive Networking sobre 5G 156
A.5 Testbed TSN sobre 5G en la Universidad de Málaga 158
A.6 Solución propuesta para sincronización de tiempo y traducción del tráfico 158
A.7 Asignación de flujos QoS dentro de una sesión PDU en DRBs 161
A.8 Enfoque global 163
A.9 Ejemplo de TOA generado 165
A.10 Impacto de AutomAdapt: Tiempo transcurrido en el estado de red deseado 171
A.11 Ejemplo de traza de red: KPI observados durante la fase inicial de aprendizaje 172
A.12 Ejemplo de traza de red: KPI observados durante el ciclo completo de funcionamiento 173

UNIVERSIDAD DE MÁLAGA

List of Tables

1.1 IEEE standards of TSN core elements 4
1.2 Industrial automation traffic types and requirements [23], [24] 5

2.1 Networking technologies proposals 42
2.2 Combined reconfiguration proposals 43
2.3 Learning techniques proposals 44
2.4 Proposals combining networking technologies, reconfiguration methods and learning approaches 46

3.1 5G Quality of Service values for common service example [26] 60
3.2 Impact of RLC parameters on end-to-end delay and jitter 66

4.1 Example of *qty* function values for a given traffic pattern 75

5.1 Example of a learned automaton with different values of the cycle time. . 98
5.2 *qty* function values used to learn automaton during the initial learning phase for a given traffic pattern 98
5.3 Evaluation traffic pattern examples 106
5.4 5G radio network configuration (RLC AM) examples 106
5.5 Automata learned during initial learning phase 111
5.6 5G network configurations (RLC AM) used to obtain baseline results for comparison 112
5.7 Resulting automata during prediction and reconfiguration phase 113
5.8 5G network configuration (RLC AM) examples 114
5.9 Count of the type of transitions of the learned states of the data set . . . 117
5.10 Number of states learned for the *qty* function definition for evaluation . . 117

A.1 Ejemplo de valores de la función *qty* para un patrón de tráfico determinado 165
A.2 Ejemplo de autómatas aprendidos con diferentes valores del tiempo de ciclo. 168
A.3 Autómatas aprendidos durante la fase inicial de aprendizaje 169
A.4 Configuraciones de red 5G (RLC AM) utilizadas para obtener resultados de referencia 170
A.5 Autómatas resultantes durante la fase de predicción y reconfiguración . . 171

UNIVERSIDAD DE MÁLAGA

A.6 Ejemplos de configuración de red 5G (RLC AM) 172
A.7 Recuento del tipo de transiciones de los estados aprendidos del conjunto de datos 173
A.8 Número de estados aprendidos para la definición de la función *qty* para la evaluación 173

UNIVERSIDAD DE MÁLAGA

Acronyms

3GPP 3rd Generation Partnership Project.

5GC 5G Core Network.

5GS 5G System.

5QI 5G Quality of Service Identifier.

AI Artificial Intelligence.

AM Acknowledged Mode.

ARP Allocation and Retention Priority.

BTS Base Transceiver Station.

CCL Closed Control Loop.

CNC Centralized Network Configuration.

CUC Centralized User Configuration.

DCS Distributed Control System.

DetNet Deterministic Networking.

DN Data Network.

DRB Data Radio Bearer.

DS-TT Device-Side Time-Sensitive Networking Translator.

eMBB enhanced Mobile Broadband.

UNIVERSIDAD DE MÁLAGA

FA Factory Automation.

FRER Frame Replication and Elimination.

GBR Guaranteed Bit Rate.

GM Grand Master.

GPS Global Positioning System.

gPTP generalized Precision Time Protocol.

HARQ Hybrid Automatic Repeat reQuest.

IBN Intent-Based Networking.

IEEE Institute of Electrical and Electronics Engineers.

IETF Internet Engineering Task Force.

IP Internet Protocol.

KPI Key Performance Indicator.

LAN Local Area Network.

MAC Medium Access Control.

MAC Media Access Control.

MCS Modulation and Coding Scheme.

ML Machine Learning.

mMTC massive Machine Type Communications.

NAS Non-Access Stratum.

NFV Network Function Virtualization.

NTP Network Time Protocol.

NW-TT Network-Side Time-Sensitive Networking Translator.

UNIVERSIDAD DE MÁLAGA

OPC Open Platform Communications.

OPC UA OPC Unified Architecture.

OSI Open Systems Intercommunication.

PA Process Automation.

PCP Priority Code Point.

PDCP Packet Data Convergence Control.

PDU Packet Data Unit.

PLC Programmable Logic Controllers.

PTP Precision Time Protocol.

QFI QoS Flow Identifier.

QoS Quality of Service.

RAN Radio Access Network.

RL Reinforcement Learning.

RLC Radio Link Control.

RRC Radio Resource Control.

SDAP Service Data Adaptation Protocol.

SDN Software Defined Networking.

SLA Service Level Agreement.

SNTP Simple Network Time Protocol.

SON Self-Optimizing Networks.

SUL System Under Test.

TCP Transmission Control Protocol.

TG Task Group.

UNIVERSIDAD DE MÁLAGA

TM Transparent Mode.

TOA Traffic Oriented input/output Automata.

TSC Time-Sensitive Communication.

TSCAI TSC Assistance Information.

TSN Time-Sensitive Networking.

TSN AF Time-Sensitive Networking Application Function.

TTs TSN translators.

UE User Equipment.

UM Unacknowledged Mode.

UPF User plane Function.

URLLC Ultra-Reliable Low Latency Communications.

WG Working Group.

WLAN Wireless Local Area Network.

XAI Explainable AI.

XML Extensible Markup Language.

ZSM Zero-Touch Network and Service Management.

UNIVERSIDAD DE MÁLAGA

Chapter 1

Introduction

Contents

1.1 Motivation 2
1.1.1 Time-Sensitive Networking 2
1.1.2 5G Network 5
1.1.3 Time-Sensitive Networking over 5G 9
1.2 Research Outline 10
1.2.1 Research Questions 11
1.2.2 Methodology 11
1.2.3 Contributions 13
1.3 Document Structure 16

Summary: This chapter introduces the thesis work, details the research motivation together with the associated technologies, and presents a summary of the research outline.

UNIVERSIDAD DE MÁLAGA

1.1 Motivation

Competition in the industry is intensifying because of the demand for products that have a certain level of quality and dependability while still being cost-effective. Industrial automation is becoming increasingly important in the context of Industry 4.0, and many new applications present novel challenges that must be addressed. Industrial protocols and standards [1], such as PROFINET and Open Platform Communications (OPC), are used to improve Ethernet networks, ensuring reliable communication between devices. The Institute of Electrical and Electronics Engineers (IEEE) 802.1 Working Group (WG) has developed a series of standards known as Time-Sensitive Networking (TSN) to ensure deterministic connectivity (with bounded latency, low jitter and minimal packet loss) over traditional Ethernet networks. Features such as time synchronization and scheduled traffic are two of the main components that guarantee Quality of Service (QoS) for these stringent communications. Therefore, these mechanisms are necessary to meet communication requirements in wired environments such as factories.

Integration of the standards discussed above with wireless communication technologies enables industrial networks to take advantage of the benefits of these technologies. Wireless technologies can extend industrial networks, enabling advantages such as accessibility, flexibility, and mobility. However, due to the stringent requirements of the applications, it is not feasible to completely replace wired solutions in industrial settings. This has led to the emergence of hybrid networks, which are a combination of wired and wireless technologies. Two wireless technologies are driving the development of hybrid TSN networks: IEEE 802.11 and 5G. The former has been proposed as a potential technology for wireless TSN due to its precise clock synchronization and moderately time-aware scheduling [2]. However, it cannot guarantee deterministic communications as it does not protect against interference from other users or radio services operating in the same frequency band. 5G, on the other hand, is the main candidate for the implementation of wireless TSN, as it offers greater quality with reserved spectrums. The techniques and procedures related to TSN over 5G integration are described in [3].

The combination of the above technologies brings with it a number of challenges to be taken into account (e.g. development of new entities with the ability to translate between the two domains). In addition, there are a considerable number of different traffic patterns that can coexist in a network, each with its own requirements (e.g. maximum delay or loss tolerance). Therefore, the task of manually configuring each traffic session to meet these requirements becomes impossible. Techniques such as Closed Control Loop (CCL) and Zero-Touch Network and Service Management (ZSM) aim to control and configure the network with minimal or ideally no human intervention.

1.1.1 Time-Sensitive Networking

Deterministic Networking is an attempt of Internet Engineering Task Force (IETF) Deterministic Networking (DetNet) WG to implement deterministic communications for real-time applications such as remote control or industrial automation [4], [5]. This

UNIVERSIDAD DE MÁLAGA

implies bounded latency, exceptionally low data loss rates, and jitter. The operation of DetNet focuses on the network layer, that is, layers 2 and 3 of the Open Systems Intercommunication (OSI) model [6].

IEEE 802.1 Time-Sensitive Networking (TSN) emerged as an evolution of Audio-Video Bridging (AVB) to encompass features demanded by the market [7]. It enables the transmission of data from time-sensitive and/or mission-critical applications over a bridged Ethernet network, which is shared by different types of applications with heterogeneous Quality of Service (QoS) requirements. Mainly, there are two traffic groups, TSN traffic (critical) and non-TSN traffic (best effort). TSN guarantees data transport with bounded latency, low jitter, and minimal packet loss, as well as ensures zero congestion loss by allocating resources for critical traffic and utilizing various queuing and shaping techniques. This leads to TSN being able to guarantee worst-case end-to-end latency for critical data. Additionally, it has a data packet-level reliability mechanism to provide ultra-reliability for data traffic, as well as protection against malfunctioning, malicious software, etc. Moreover, time synchronization is essential to provide other TSN functions. Therefore, TSN is a set of standards that can be considered a packet service containing a set of tools (combinable with each other) to meet the requirements of time-sensitive traffic [8]. Figure 1.1 represents the fully centralized TSN network architecture, which is made up of the following elements:

- ***TSN endpoints***. Final devices that send and receive traffic, in this case, talker (source) and listener (destination), respectively. These devices communicate the traffic QoS requirements in advance to the Centralized User Configuration.
- ***TSN bridges***. Particular switches in charge of the Ethernet frames transmission of TSN flows.
- *Centralized Network Configuration* (CNC). Device responsible for the TSN features configuration in the network to meet QoS traffic requirements.
- *Centralized User Configuration* (CUC). Element that exchanges the necessary information with the CNC to configure the TSN network on behalf of the TSN endpoints.

This architecture allows the network to obtain the capabilities and user requirements of the final devices. The key benefits of the core elements of TSN are as follows. In addition, IEEE (Institute of Electrical and Electronics Engineers) standards related to these core elements are presented in Table 1.1.

- ***Time synchronization***. Precise synchronization signal (<1 μs) distributed among the devices using the generalized Precision Time Protocol (gPTP).
- ***Traffic shaping***. Control mechanism for traffic injection to guarantee latency and minimum jitter/packet loss.
- ***Reliability***. Control and reservation of different paths ensure expected performance using Frame Replication and Elimination (FRER).

UNIVERSIDAD DE MÁLAGA

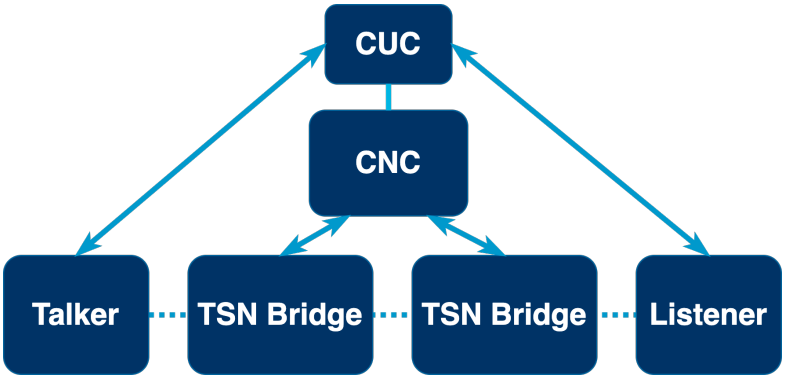


Figure 1.1: Fully centralized TSN network architecture - Copyright © 2022, IEEE 802.1Qcc [9]

Time synchronization	*Traffic shaping*	*Reliability*	*System configuration*
802.1 **AS** [10]	802.1 **Qav** [11]/**Qbv** [12] Qcr [13]/Qch [14] Qbu [15]/BR [16]	802.1 **Qbb** [17] Qca [18]/Qci [19]	802.1 **Qcc** [9] CS [20]/Qat [21]

Table 1.1: IEEE standards of TSN core elements

– ***System configuration***. Automatic configuration of the network to meet the requirements of the different traffic flows, regardless of the equipment supplier (interoperability).

In addition, a TSN network supports a variety of types of traffic associated with specific service requirements, which have been defined by 3rd Generation Partnership Project (3GPP) [22], Institute of Electrical and Electronics Engineers (IEEE) [23] and the Industrial Internet Consortium (IIC) [24]. A summary of these types of traffic and their associated requirements is presented in Table 1.2. Specifically, traffic can be periodic (P), where packets are sent systematically in a cyclic pattern; or sporadic (S), that is, a packet can be transmitted at any time. The main characteristics to be defined are the following, where 'na' means that the characteristic does not apply, and τ is the latency. Additionally, if the application can cope with the characteristic, the value is 'Yes', and the specific amount needs to be specified. In contrast, the value is 'None', meaning that the application cannot tolerate the characteristic.

- *Typical period*. Interval between consecutive packets of the application.
- *Tolerance to jitter*. The jitter tolerance that the application can withstand, which is relevant for periodic traffic.
- *Tolerance to loss*. Indicates whether the application can tolerate loss or failure to meet requirements in a sequence of consecutive packets.

UNIVERSIDAD DE MÁLAGA

Traffic types (Periodic/Sporadic)	*Typical period*	*Tolerance to Jitter*	*Tolerance to loss*	*Typical data size (bytes)*
Isochronous (P)	100μs $\sim$ 2 ms	None	None	Fixed: 30 $\sim$ 100
Cyclic -Synchronous (P)	500μs $\sim$ 1 ms	$\leq \tau$	None	Fixed: 50 $\sim$ 1000
Cyclic -Asynchronous (P)	2 ms $\sim$ 20 ms	$\leq \tau$	1 $\sim$ 4 Frames	Fixed: 50 $\sim$ 1000
Events: control (S)	10 ms $\sim$ 50 ms	na	Yes	Variable: 100 $\sim$ 200
Events: alarm & operator commands (S)	25	na	Yes	Variable: 100 $\sim$ 1500
Network control (P)	50 ms $\sim$ 1 s	Yes	Yes	Variable: 50 $\sim$ 500
Configuration & diagnostics (S)	na	na	Yes	Variable: 500 $\sim$ 1500
Video (P)	na	na	Yes	Variable: 1000 $\sim$ 1500
Audio/Voice (P)	na	na	Yes	Variable: 1000 $\sim$ 1500
Best effort (S)	na	na	Yes	Variable: 30 $\sim$ 1500

Table 1.2: Industrial automation traffic types and requirements [23], [24]

- *Typical data size.* The size of the packet application encapsulated in the Ethernet payload, which can range from 46 to 1500 bytes, as reported in [25].

1.1.2 5G Network

5th Generation of Mobile Telephony (5G or 5GS) is a global standard for wireless communications presented as an evolution of previous generations (e.g. 4th Generation). However, 5G wireless technology will do much more than simply significantly enhance the network connection, enabling a new type of network that opens up new opportunities for telecom operators and end users. 5G networks provide high data rates, ultra-low latency, and considerable connection capacity. This leads to a significant improvement in the QoS offered, and therefore in an improvement perceived by users called Quality of Experience (QoE). Therefore, 5G enables the use of cellular networks in new sectors (also known as verticals) such as automotive, public safety, health, and discrete automation and also for time-critical applications.

UNIVERSIDAD DE MÁLAGA

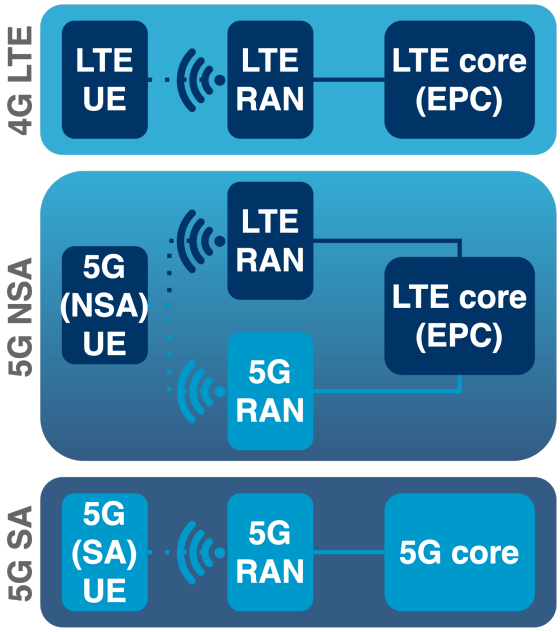


Figure 1.2: System Architecture Evolution: 4G, 5G NSA and 5G SA

5G Architecture

Similarly to older cellular technologies, the 5G system architecture is composed of the User Equipment (UE), the Radio Access Network (NG-RAN) and the Core Network (5GC). Figure 1.2 represents the evolution of the architecture since 4G, also known as Long Term Evolution (LTE). Note that the 5G architecture presents two deployment options. On the one hand, 5G Non-Stand Alone (NSA), which is built on top of the 4G infrastructure to leverage the 4G core network, also uses 5G Base Stations (BS) together with 4G BS to provide end-to-end communication. On the other hand, the implementation of 5G Stand-Alone (SA) enables new components, enabling new capabilities and features. This thesis work uses the 5G Standalone (5G SA) deployment depicted in Figure 1.3 as the reference architecture throughout the rest of the document, which is defined as service-based in which functionalities are delivered through interconnected entities. Therefore, the following is a brief description of the main components of the 5G SA architecture, as well as the main elements that make up it.

- ***User Equipment*** (UE). End-user device (e.g. mobile phone, laptop) for radio communications, that is, transmission and reception.
  - Mobile Station (MS). Entity with radio access capabilities (one or more radio interfaces).
  - Universal Subscriber Identity Module (USIM). Component that stores user information (e.g. identification, authentication and ciphering keys)

UNIVERSIDAD DE MÁLAGA

- ***Radio Access Network*** (RAN). Also known as Next Generation RAN (NG-RAN). Element to interconnect UEs through radio links with the core network.
  - gNodeB (gNB). 5G Base Station (BS) responsible for radio communication with UEs within a specific area (i.e. cell). A gNB can be a solely physical element or a partially software-based element, such as a software defined radio (SDR) where some operations (e.g. signal modulation and demodulation) are implemented in software instead of hardware or Open RAN (O-RAN) where gNB is split into the Radio Unit (RU), the Distributed Unit (DU), and Centralized Unit (CU).
- ***5G Core Network*** (5GC). The 5GC is in charge of various tasks to enable end-to-end communication. They are divided into the so-called Network Functions (NF) which provide different services. Additionally, most of them are software-based, simplifying the deployment and adaptation to the requirements.
  - **Application Function** (AF). Provides services that can play a key role in traffic management and QoS assignments interacting with other elements.
  - **Access and Mobility Management Function** (AMF). Responsible user access (e.g. registration) and mobility management (e.g. handovers).
  - Authentication Server Function (AUSF). User authentication.
  - Data network (DN). Internet access and service provider/third party services.
  - Policy Control Function (PCF). Management and implementation of QoS policy rules.
  - **Session Management Function** (SMF). Management of traffic session information (e.g. establishment, modification, and release).
  - Unified Data Management (UDM). Repository of user data.
  - **User plane Function** (UPF). Handling user data (e.g. anchor point for external connections, routing, forwarding, Internet Protocol (IP) address allocation, etc.).

Note that the architecture presented is an elemental configuration. Recent 3GPP standards [26] define more than 30 network entities (in conjunction with their interfaces) and functions depending on the services provided. In addition to those NFs described above, the following stand out:

- Network Exposure Function (NEF). Entity to enable the exposure of 5G capabilities through Application Programming Interfaces (API). Facilitates the development of third-party applications by taking advantage of the potential of 5G networks.
- Network Data Analytics Function (NWDAF). Data collection and exposure that can be analyzed to obtain predictive and real-time analytics.

UNIVERSIDAD DE MÁLAGA

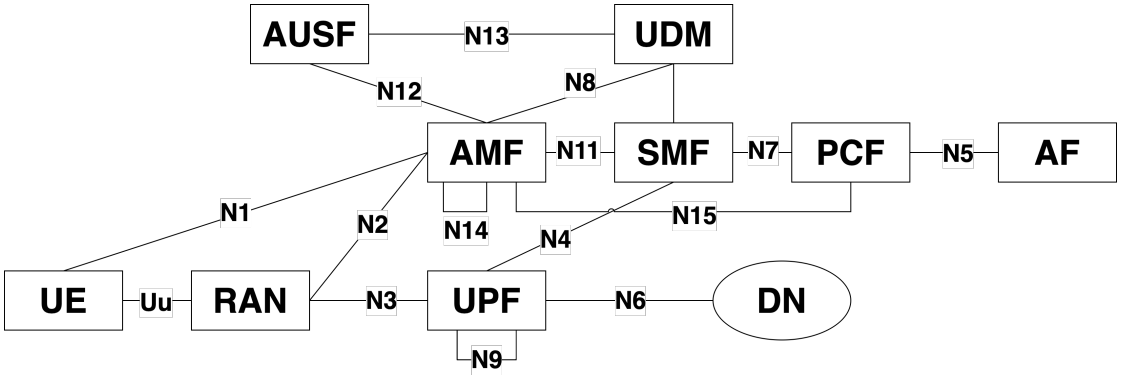


Figure 1.3: 5G System Architecture (source [26])

– NF Repository Function (NRF). Centralized repository for all NFs that can be consulted by others.

Key Technologies

There are significant indicators that measure progress toward achieving expected quality, called Key Performance Indicators (KPIs). The International Telecommunications Union (ITU) defined the requirements for 5G networks (for example, 1 ms latency and 99.9999% availability). 5G KPIs are grouped mainly into three categories: enhanced Mobile Broadband (eMBB), Ultra-Reliable Low Latency Communications (URLLC), and massive Machine Type Communications (mMTC), which define different use cases and requirements. More categories have been added to recent standards [26], for example, Vehicle-to-everything, also known as V2X. Additionally, there are key technologies to improve and extend the capabilities of 5G networks [27], which are introduced below.

- Software Defined Networking (SDN). Architectural approach to separate the control plane from the data plane in network devices. In this way, the control can be dynamically managed from an external and centralized element called the SDN controller. It allows the 5G network to provide efficient resource allocation and flexibility, since the network can be tailored to traffic patterns.
- Network Function Virtualization (NFV). Concept of network architecture to manage the network on demand using virtual and software-based applications. It enables flexibility, scalability, and cost-effectiveness by allowing virtual resources to be created over a physical infrastructure. In the case of 5G network, not only the traditional dedicated hardware (e.g. firewall) can be virtualized but also vital elements such as base stations and core network.
- Edge computing. Paradigm that brings computing and data storage closer to the user location. It grants the improvement of response times and saves bandwidth, which is essential for low-latency applications and services.

UNIVERSIDAD DE MÁLAGA

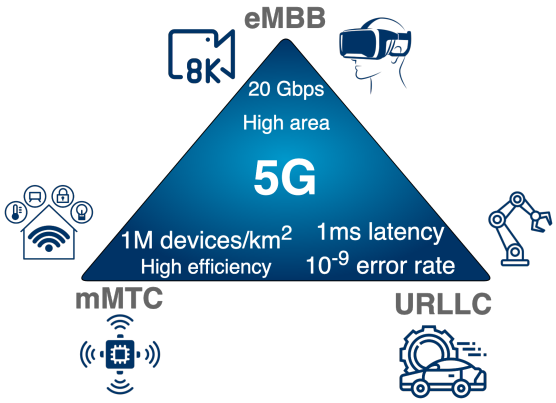


Figure 1.4: 5G Service Categories and KPIs

- Network slicing. Concept that involved the creation of multiple virtual networks on a shared physical infrastructure, allowing the allocation of specific resources on each network slice according to the needs of the application and users. Therefore, the requirements of different 5G KPI categories can be met, allowing the execution of various uses cases in the same 5G network.

1.1.3 Time-Sensitive Networking over 5G

3GPP has made notable progress in leveraging the benefits of wireless networks for Industry 4.0 (e.g. automation), in particular, the operation of TSN standards over 5G networks. TSN is considered an enabler for deterministic communications in the factories of the future. Therefore, 5G TSN will provide this kind of communication with high reliability and low-latency. Time synchronization is fundamental between the devices of the system to ensure the QoS characteristics required by the applications. From Release 16 [28] onward, 3GPP is pushing to define an architecture in which the 5G network, called 5G System (5GS), can act as a transparent TSN bridge, in this case labeled as *5GS Bridge* [26], [29]. This architecture is depicted in Figure 1.5, where, in addition to the common elements of the TSN and 5G networks, new entities specialized to support Time-Sensitive Communication (TSC) are included. These entities are the TSN translators, Device-Side Time-Sensitive Networking Translator (DS-TT) next to the 5G UE and Network-Side Time-Sensitive Networking Translator (NW-TT) next to the UPF, and the Time-Sensitive Networking Application Function (TSN AF).

TSN Translators

The TSN translators (TTs) allow interoperation between the TSN system and the 5GS, including the control and user planes. The whole end-to-end system can be treated as an time-aware system according to IEEE 802.1AS [10]. While the 5G network components (UE, gNB and 5GC) are synchronized with the 5G Grand Master (GM) and the TSN elements (TSN endpoints and bridges) with the TSN GM, the TTs must be

UNIVERSIDAD DE MÁLAGA

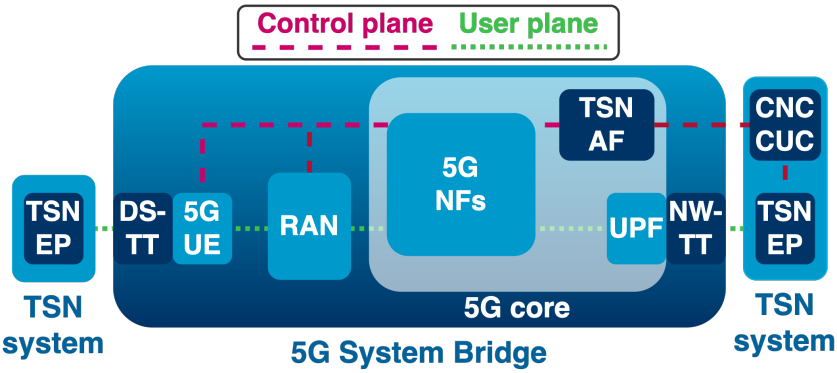


Figure 1.5: Time-Sensitive Networking over 5G architecture

synchronized with the 5G GM and support IEEE 802.1AS functions, for example, the gPTP support [30].

TSN Application Function

Time-Sensitive Networking Application Function (TSN AF) is a 5G NF in charge of communication between the 5GS and the TSN network, specifically with CNC. The main objective is to configure 5GS to act as a TSN bridge by exchanging information such as the capabilities and configurations of both domains, 5G and TSN [31]. In addition, it provides the control plane of the TSN translators for the integration of the two networks.

Figure 1.6 illustrates an example of the ideal operation of the system. Initially, the TSN endpoint requests the creation of a traffic session (i.e. end-to-end connectivity) with specific QoS to the TSN AF (through CUC/CNC entities), specifying the traffic pattern and the expected KPIs. Subsequently, the TSN AF and the CNC exchange information on the capabilities and topology, then configure the 5G network and the TSN translators. Once the network is adapted to fulfill such requirements, the TSN AF notifies the TSN endpoints that traffic can be sent and received. During the traffic session, if a deviation is observed from the expected KPIs, a reconfiguration of the network is performed to continue meeting the traffic requirements. However, this operation cannot be natively supported, due to the fact that neither TSN nor 5G have mechanisms for self-reconfiguration [32], [33], it is necessary to add intelligence to the system through the use of learning techniques.

1.2 Research Outline

The main objective of this thesis is to achieve automatic configuration in the context of a 5G private network in industry to meet the TSN traffic requirements demanded by applications, following the Zero-Touch approach. In this way, automata learning, also known as model learning, is applied to design and build a continuous learning and modeling environment of the system. On the basis of this objective, the following research

UNIVERSIDAD DE MÁLAGA

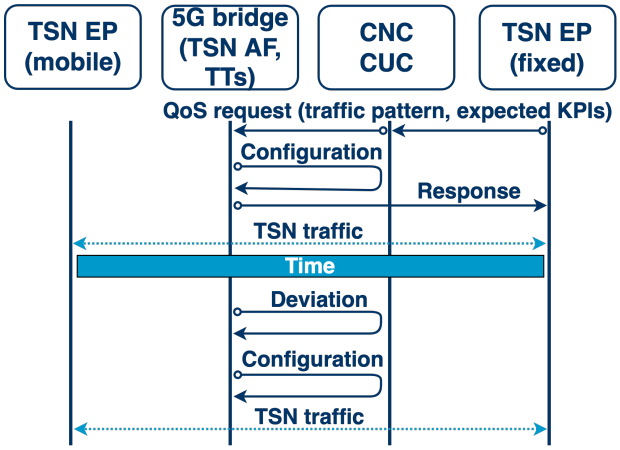


Figure 1.6: QoS request and automatic reconfiguration of the network

questions arise as a guide to carry out the research.

1.2.1 Research Questions

+ **Research Question 1 (RQ1)**: Does existing equipment support the integration of Time-Sensitive Networking into the 5G system?
+ **Research Question 2 (RQ2)**: What automated methods can be used to configure the 5G network to meet the requirements of TSN traffic?
+ **Research Question 3 (RQ3)**: How can the explainability of the system be enhanced to ensure a clearer understanding of its functioning?

1.2.2 Methodology

A methodology composed of six phases is defined to facilitate the investigation. Figure 1.7 shows the two complete iterations of these six phases that have been carried out over 36 months.

1. Study of related literature.
2. Identification and review of requirements of URLLC applications in industrial environment.
3. General architecture of the proposal, including the learning and configuration process, as well as integration with the protocol developed for URLLC communications.
4. Work on the construction of 5G network behavior models.
5. Design and analysis of automatic configuration algorithms.
6. Prototyping and implementation of a tool containing the automatic configuration algorithms, including the construction of the 5G TSN testbed, on which the tool is tested.

UNIVERSIDAD DE MÁLAGA

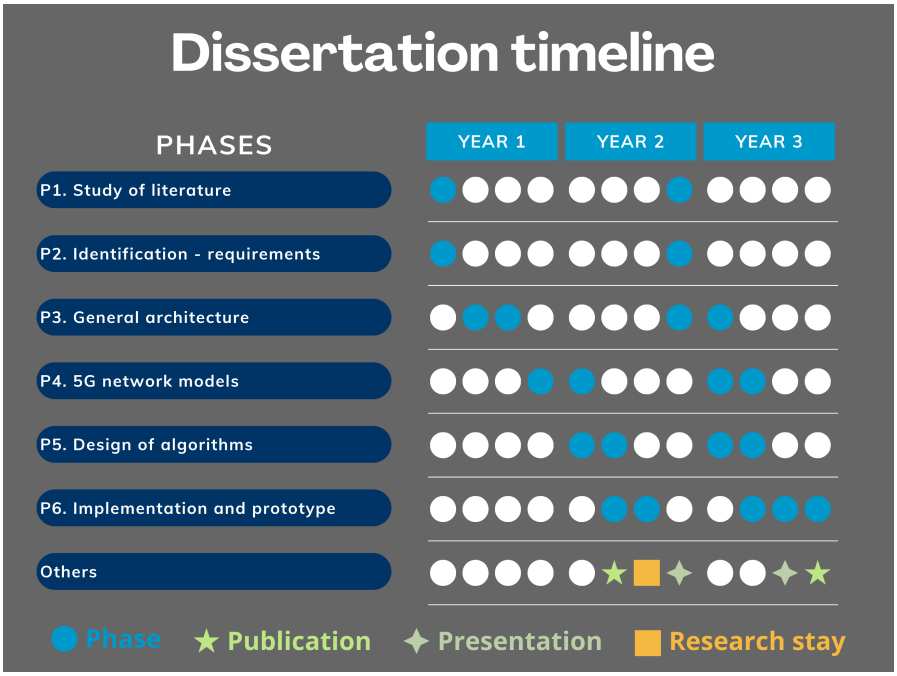


Figure 1.7: Dissertation timeline

The first iteration serves as a basis to know the feasibility of the research questions posed, allowing to know the points that can be improved in a complete cycle in order to improve them during the second iteration. The research work has been divided into three main objectives to address the research questions. Furthermore, some secondary objectives are defined to demonstrate the applicability of the results.

Main Objectives

- **Objective 1 (O1)**: Prove the feasibility of TSN over 5G networks by developing a prototype of entities and procedures required for traffic translation between both domains.
- **Objective 2 (O2)**: Demonstrate the ability to automatically build models based on automata learning for efficient self-configuration of an industrial 5G network.
- **Objective 3 (O3)**: Design and implement automated industrial 5G network reconfiguration algorithms from the models to ensure the requirements imposed by deterministic real-time traffic (TSN) in different traffic situations, users, and end-to-end conditions.

UNIVERSIDAD DE MÁLAGA

Secondary Objectives

- **Objective 4 (O4)**: Produce a catalog of 5G network automata that includes suitable 5G network configurations for a set of TSN over 5G traffic profiles.
- **Objective 5 (O5)**: Design of a tool that integrates both the algorithms and the modules needed to reconfigure the 5G network without human intervention.

1.2.3 Contributions

This doctoral thesis focuses on the objectives defined above to address the research questions. During the development of the thesis work, a series of contributions have been obtained, which are presented below. Figure 1.8 presents the relationship between the research questions, objectives and contributions of this thesis.

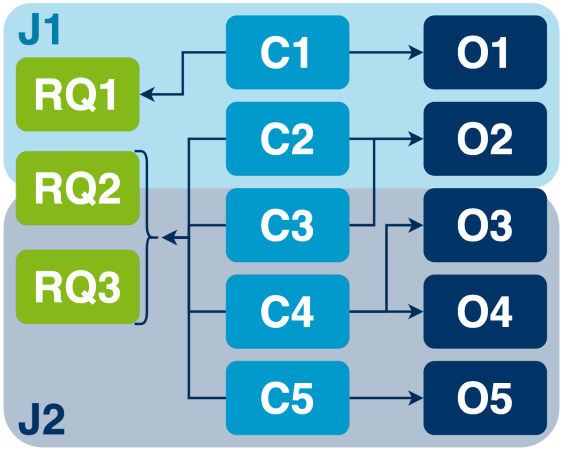


Figure 1.8: Relationship between research questions, objectives and contributions

+ **Contribution 1 (C1): Construction of a testbed for the validation of TSN over 5G proof of concept.**

  RQ1 seeks to understand the current state of integration of both technologies, TSN and 5G. First, an analysis has been performed from 3GPP Release 16 [28] onward to get to know both the architecture and the new entities and procedures needed for the integration. Second, a study of published work related to the 5G TSN testbed has been carried out, as well as the development of these entities and procedures. In parallel, the requirements of the different existing TSN traffic profiles have been studied to determine their viability over 5G networks, which allows us to understand the related benefits. The construction of a testbed that enables TSN validation over 5G testing allows us to address **RQ1** and **O1**. It is worth mentioning that the testbed includes the development of a TSN translators (NW-TT and DS-TT) prototype using the P4 programming language and the proposition of an alternative solution to achieve time synchronization in TSN translators.

UNIVERSIDAD DE MÁLAGA

+ **Contribution 2 (C2): Algorithm design to automatically build a 5G network model from network traces (passive automata learning).** The objective of **RQ2** is to study and determine the appropriate approach to achieve automatic reconfiguration of the 5G network based on the requirements of TSN traffic. Furthermore, **RQ3** pursues the investigation of learning techniques and approaches to achieve network reconfiguration.

  On the one hand, the current literature contemplates the use of learning techniques to analyze network data and predict potential problems and deviations. Techniques such as Artificial Intelligence (AI) and Machine Learning (ML) can be used for this purpose and try to optimize the values of different parameters. However, the particular scenario of TSN over a 5G network with time-varying parameters, such as buffers and timers, raises the importance of not only knowing the current state of the network. Therefore, it is essential to take into account both the KPIs of the traffic being sent and the network configuration. Model (or Automata) learning can be used to produce behavioral models of the System Under Test (SUL) which is represented with an automaton. On the other hand, there are several approaches in the context of self-configuration applied for networks, such as Self-Optimizing Networks (SON), Intent-Based Networking (IBN) and Closed Control Loop (CCL). These approaches and techniques collectively contribute to the concept of Zero-Touch Network and Service Management (ZSM) that aims to automate the deployment, configuration, and management of the network without human intervention.

  Consequently, we developed an algorithm to build a model that represents the behavior of the 5G network to address **RQ2** and **O2**. The model is represented by an automaton consisting of network states for different traffic profiles with specific requirements, which focuses on **RQ3**. Each state includes the network configuration and KPIs. In addition, the automaton is automatically created from network traces following the passive learning concept. Note that we are able to create different network models using a variety of network traces, that is, automata that symbolize the behavior of the 5G network using different traffic patterns and network configurations.

+ **Contribution 3 (C3): Extension of the automatic 5G network model building algorithm (C2) from real-time traffic (passive automata learning extension).** Following the work of **C2**, we defined a new kind of automata called Traffic Oriented input/output Automata (TOA), which directly contributes to **O2** and allows us to represent the evolution of network behavior (i.e. TSN traffic sent through the 5G network) in relation to time in the states of the network mentioned above. The algorithm was then extended to generate TOA based on TSN traffic over 5G, either from network traces and real-time monitoring, where the automaton can partially interact with the system. Accordingly, this development answers to **RQ2** and **RQ3**.

UNIVERSIDAD DE MÁLAGA

+ **Contribution 4 (C4): Algorithm design for the prediction and reconfiguration of 5G networks.** To finish the answer of **RQ2** and address **RQ3**, the development of an automatic prediction and reconfiguration algorithm that can take advantage of the information learned about the behavior of the 5G network is essential. Therefore, this algorithm has to exploit TOA for a particular scenario (for example, a specific traffic pattern) where it is assumed that the network follows the behavior learned previously. The objective is to predict possible deviations of the observed KPIs to undesirable quality values. Then, an alternative network configuration is proposed in advance based on a learned TOA to avoid such deviations, which is the main objective of **O3**. Note that the use of different TOAs learned with different types of traffic, as well as the number of states included in each automaton, will contribute to the effectiveness of the proposed prediction and reconfiguration, since the automaton contains a larger amount of information on which it relies to make the decision. Furthermore, this leads to the production of a catalog that includes the 5G network models and configurations, which are feasible for diverse TSNs over 5G traffic profiles (**O4**).

+ **Contribution 5 (C5): Implementation of a tool to automatically reconfigure the 5G network to meet traffic requirements, including algorithms designed (C2, C3, and C4) and capable of running on the testbed (C1).** Implementation of a tool capable of integrating the algorithms developed, as well as the necessary modules (**O5**) is vital to achieve the primary objective of this thesis. The study of discrete relationships, time events or stochastic processes that can occur in the network is essential to respond to **RQ2** and **RQ3**. In addition, it is crucial to identify how they can be captured using the TSN over 5G testbed. First, the tool was implemented to capture this information from previously captured network traces (**C2**). In this case, the 5G model is built using these network traces. However, more network traces need to be captured to continue learning the behavior of the network. Second, the tool was extended to support real-time traffic. In this way, the 5G network model can be constructed in real time (**C3**), enabling continuous learning of the behavior of the network. In both cases, 5G network models can be used for network reconfiguration (**C4**). Moreover, the tool can be executed on the testbed (**C1**), allowing a complete integration into the system.

These contributions have resulted in the publication of two journals [34], [35] (see Section 6.3) containing key aspects of the work done during the thesis. The first journal (**J1**) contains the results of the work carried out during the first iteration of the methodology of this thesis, addressing the **RQ1** and **RQ2**, in addition to the contributions and objectives **C1**, **C2**, **O1** and **O2**, respectively. The second journal (**J2**) results from the work done in the second iteration of the methodology, being a continuation of the first, focusing mainly on responding to **RQ2** and **RQ3**, while addressing contributions **C3**, **C4**, **C5** and objectives **O3**, **O4** and **O5**.

UNIVERSIDAD DE MÁLAGA

1.3 Document Structure

This dissertation is organized in five chapters and two appendices. The structure of the document is presented as follows.

Chapter 1. Introduction

This chapter comprises the motivation of the work, including an overview of the key components, and the research outline, which presents the research questions, the methodology followed, as well as the objectives and contributions of this thesis.

Chapter 2. Fundamentals and State of the Art

The related work with regard to the key aspects of this thesis is studied in this chapter. First, the advanced networking technologies associated with this work, including the integration of Time-Sensitive Networking (TSN) over 5G networks. Second, techniques and methods (e.g. Zero-Touch) that serve as enablers for automatic reconfiguration of the network. Third, learning approaches such as automata learning applied to build 5G network models aimed at prediction and network reconfiguration.

Chapter 3. Toward TSN over 5G Testbed Implementation

This chapter presents the work related to the implementation of TSN over 5G testbed. On the one hand, the components and elements of both domains, TSN and 5G. On the other hand, the proposed solution to the challenges arising from the integration of these technologies. Furthermore, this chapter presents the study of the network parameters that can be essential to meet the stringent traffic requirements of TSN over 5G.

Chapter 4. Model Learning Approach

This chapter introduces the proposal to achieve the main aim of this work, self-reconfiguration of the 5G network based on traffic requirements through automatic model building of the TSN over the 5G network using model learning approach. In addition, the design of automata and the algorithms for learning, prediction, and reconfiguration are presented in depth.

Chapter 5. Implementation and Evaluation of the Proposal

This chapter presents the implementation of the proposal in detail, including the development of a tool called AutomAdapt capable of integrating the aforementioned algorithms and essential modules to address the objective of the thesis. Moreover, it introduces the evaluation of this tool conducted in the TSN over 5G testbed developed at the University of Malaga as part of this thesis.

UNIVERSIDAD DE MÁLAGA

Chapter 6. Conclusions and Future Work

The conclusions obtained during the development of the thesis and future work with the objective of extending this work are presented in this chapter. In addition, the publications and related projects are detailed.

Appendix A. Resumen en Español

This appendix summarizes this thesis in Spanish.

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Chapter 2

Fundamentals and State of the Art

Contents

2.1 Advanced Networking Technologies 21
2.1.1 Industrial Communications standards and protocols 21
2.1.2 Deterministic Networking and Time-Sensitive Networking . . . 22
2.1.3 Hybrid Networks for Industry 26
2.1.4 Testbeds 30
2.2 Enabling Techniques and Methods for Reconfiguration 31
2.2.1 Self-Organizing Networks 31
2.2.2 Closed Control Loops 32
2.2.3 Intent-Based Networking 33
2.2.4 Zero-Touch Networks 34
2.2.5 Network reconfiguration 36
2.3 Learning Approaches 38
2.3.1 Behavioral Modeling 38
2.3.2 Artificial Intelligence and Machine Learning 39
2.3.3 Model Learning 40
2.4 Evaluation and Challenges 41
2.4.1 Networking Technologies Integration 42
2.4.2 Combined Reconfiguration Methods 43
2.4.3 Application of Learning Techniques 44
2.5 Analysis and Promising Research Topics 45

UNIVERSIDAD DE MÁLAGA

Summary: This chapter offers the fundamentals of the research, providing the solutions studied in the literature related to this work. First, the key networking technologies. Second, the existing techniques and methods toward the implementation of automatic reconfiguration. Third, the learning techniques applied to support the automation of the network.

UNIVERSIDAD DE MÁLAGA

2.1 Advanced Networking Technologies

2.1.1 Industrial Communications standards and protocols

The standardization process [36] is fundamental in addressing the specific needs of industrial communication systems. Industrial communication standards and protocols are essential to allow data exchange in Industry 4.0 [37]. They are necessary to connect different systems and devices, allowing interoperability and support of different applications, such as Process Automation (PA) and Factory Automation (FA). In the context of Industry 4.0, communication links between the various components must be reliable and capable of deterministic and real-time transmission. There exist diverse communication protocols that provide these capabilities, which are mainly divided in two categories, *Fieldbuses* and *Industrial Ethernet* protocols [38], [39].

Fieldbuses enable direct communication between input devices (e.g. sensors) and output devices (i.e. actuators). These simple design and implementation protocols lead to cost effectiveness by avoiding the connection of each element to a controller, for example, Programmable Logic Controllers (PLC). In addition, in the automation field, an indispensable requirement is support for real-time transmission, where a small number of bytes are transmitted over short distances. Some of the best-known **fieldbus** protocols are as follows.

- PROFIBUS [40]. Process Field Bus (PROFIBUS) is an open fieldbus standard for communication between field devices (sensors) and control systems (controllers). It uses cycles for data transmission, while alarms and diagnostics are transmitted sporadically (only when necessary).
- CANBUS [41]. Controller Area Network (CANBUS) was originally created to ensure communication between field devices in automotive. However, its use has been extended to other areas, such as industrial automation. The CANBUS controller is connected to the network components using two dedicated cables. Every node receives all the data, although it only responds when its unique identification matches.
- MODBUS [42]. MODBUS is an old open protocol, in which devices only provide information when requested by the controller. It is still used in factories. However, it is vulnerable to external attacks (e.g. service denials).

Ethernet/IP networks are widely used; in particular **Industrial Ethernet** [43] that is employed in many industries to provide determinism and real-time control. The following Industrial Ethernet protocols play an major role in these communications.

- PROFINET [44]. The Process Field Network (PROFINET) is an extended PROFIBUS protocol implemented over Ethernet that provides real-time (cyclic) connectivity. In addition, it supports non-real-time communication (acyclic). Therefore, it is suitable for a wide range of services, connecting PLCs and Distributed Control System (DCS) with sensors and actuators.

UNIVERSIDAD DE MÁLAGA

- EtherCAT [45]. Ethernet for Control Automation Technology (EtherCAT) is an open, real-time protocol with high-performance capabilities, where the controller is the only one sending data through the network, avoiding collisions. Consequently, it is used in applications where fast communication is required, such as demanding motion control.
- POWERLINK [46]. Ethernet POWERLINK is an open, real-time protocol that supports low-latency and deterministic communication, making it feasible for applications such as automation. It operates on Ethernet hardware, allowing for cyclic and acyclic traffic.

There are other protocols that are less used in industries. For example, Sercos III [47] and CC-Link [48]. In addition, there exist several variants of the protocols presented above. For example, MODBUS PA for process automation environments [49], MODBUS based on Transmission Control Protocol (TCP) [50] or MODBUS Remote Terminal Unit (RTU) with a compact binary representation of the data [51]. Furthermore, OPC Unified Architecture (OPC UA) is a communication protocol not only for field devices, but also for other industry levels (e.g. mobile and cloud applications). Accordingly, it is a platform-independent service-oriented architecture that includes the classic features of OPC, facilitating secure and reliable data exchange in industrial systems [52]. In the same way, Time-Sensitive Networking (TSN) aims to improve Ethernet networks for deterministic and time-sensitive communication. For this purpose, TSN introduces mechanisms to ensure the traffic requirements of time-critical applications like industrial automation. However, TSN is not a standalone protocol, but a set of standards to enhance and extend traditional wired networks [53]. Moreover, TSN can be applied to higher-layer protocols, including those used for industrial automation, such as PROFINET [54] and EtherCAT [55]. Therefore, TSN is a remarkable solution that should be considered as a reference for industrial environments [56] and, when used together with OPC UA, it can contribute to network flexibility [57].

2.1.2 Deterministic Networking and Time-Sensitive Networking

As introduced in the previous section, there are several standards that define technologies, methods, and procedures necessary to provide reliability and predictability over traditional networks, which do not offer any packet delivery guarantee and sometimes experience poor performance (i.e. high packet loss and jitter) in congestion situations. For instance, the TSN Task Group (TG) as part of IEEE 802.1 WG aims to support determinism in Ethernet networks.

The IETF Deterministic Networking (DetNet) [58] WG focuses on the delivery of deterministic services in wired and wireless networks. Furthermore, the DetNet standards include the operation in layers 2 (data link) and 3 (network) of the OSI model. Similarly to Time-Sensitive Networking, *Deterministic Networking* provides capabilities for real-time applications such as extremely low packet loss, high reliability, and bounded latency and jitter. Despite the fact that both TSN and DetNet have the objective of creating

UNIVERSIDAD DE MÁLAGA

high-reliability and low-latency industrial networks applying different techniques, DetNet seeks to extend some methods to a higher layer (e.g. time synchronization and FRER). The Deterministic Networking architecture is defined in [4], where the network nodes that implement DetNet capabilities can meet the QoS requirements of traffic, namely DetNet flows. The Qos in deterministic networks concerns only the worst-case values of the KPIs, that is, maximum end-to-end latency and jitter, maximum loss ratio, and limit for out-of-order packet delivery. The three techniques for providing such QoS are briefly explained below. Note that these techniques can be used separately or together.

- Resource allocation. The assignment of resources (e.g. link bandwidth and buffer space) is essential to reduce (or even eliminate) the packet loss. However, this can only be achieved by limiting packet size and transmission rates, since flows are allocated in advance based on the network capacity. Moreover, this contributes to the bounded latency, due to the delimited buffer spaces on the nodes.
- Service protection. Some mechanisms such as FRER and packet encoding are applied to mitigate equipment failures and random media errors, respectively. In addition, the DetNet Packet Ordering Function algorithm can be implemented to provide in-order delivery, since out-of-order packets can affect KPIs like jitter.
- Explicit routes. In order to avoid interruptions caused by routing protocols (for example, the Open Shortest Path First protocol), the end-to-end paths for specific flows are explicitly defined.

In addition to the above techniques that focus on the fulfillment of time-sensitive (TS) application (critical traffic) requirements, the coexistence with non-time-sensitive applications (noncritical traffic) in the network must be taken into account [5]. Therefore, traffic scheduling must ensure the transmission of non-critical traffic with the highest priority. Also, filters and traffic policing are used for fault detection (e.g. a packet received at the wrong time or interface) with the aim of reducing possible issues. A DetNet end system can exist on top of lower-later technologies such as TSN. Several standards specified by IEEE 802.1 WG define techniques applied in deterministic networks to achieve the bounded latency due to the fact that the allocation of resources alone is not enough. For example, credit-based shaper in IEEE 802.1Qav and time-aware scheduler in IEEE 802.1bv (see Table 1.1 for further details). Furthermore, there are several types of DetNet flow depending on the end system composition and supported features.

In essence, TSN and DetNet aim to build high-reliable and ultra low-latency networks with similar characteristics, including traffic scheduling and time synchronization. Whereas TSN focuses on OSI layer 2, DetNet extends some techniques to OSI layer 3. Yang et al. [59] compare TSN and DetNet, emphasizing their features and published standards. In addition, they provide use cases and next steps for these technologies.

UNIVERSIDAD DE MÁLAGA

TSN - Traffic Scheduling

The *traffic scheduling* mechanism allows the management of traffic flows within a network to improve performance, specifically to achieve bounded latency and zero congestion loss. Some key aspects of traffic scheduling are as follows. First, priority queues where packets are assigned different priorities based on their criticality. Second, traffic shaping is used to control data rate, avoiding congestion with bursty traffic. Third, buffer management is vital to avoid packet loss and congestion, which also affects delay and jitter. In addition, several algorithms are used to determine the transmission order of the packets depending on different characteristics (e.g. bandwidth available or packet length). For example, Round Robin (RR), Weighted Fair Queuing (WFQ), and Deficit Round Robin (DRR) [60]. There are works that deal with the traffic scheduling problem in networks that implement DetNet [61]–[63], networks that implement TSN [64]–[66] and the combination of both approaches [67], [68].

From the perspective of DetNet networks, the authors of [61] propose an approach to selecting two paths, in which end-to-end delays are considered to schedule traffic to maximize accepted traffic between endpoints. On top of that, they propose two additional approaches for scalability, increasing the number of requests in an acceptable time. Similarly, the work in [62] presents two algorithms to maximize traffic acceptance, focusing on large-scale deterministic networks. One similarity of both papers is the use of the Cycle Specified Queuing and Forwarding (CSQF) mechanism, an extension of Cyclic Queuing and Forwarding (CQF) that enables various transmission queues simultaneously. Alternatively, the paper in [63] introduces a realistic implementation of a deterministic network that is capable of granting bounded end-to-end latency and jitter in a large-scale network. It also presents an algorithm to maximize accepted traffic.

On the TSN networks side, a complete evaluation of CQF-based variants is presented in [64], compared to TAS (802.1Qbv IEEE standard) in industrial TSN systems. In addition, they demonstrate that Multi-CFQ can manage flows with stringent time requirements. Nevertheless, it cannot be considered for large-scale networks, as it is based on the number of hops, which would directly increase end-to-end delays. Furthermore, the problem of considerable flow size is discussed in [65], where the results show that the proposed algorithm performs better in this case. However, it is not applicable to dynamic flow scheduling. On the contrary, a dynamic scheduling method is presented in [66]. The authors show that a substantial number of flows can be scheduled in reasonable time, which is vital for dynamic reconfiguration considering that the new configurations are based on the existing one and are evaluated in advance to avoid incompatibilities with current traffic.

In general, these investigations only take into account both approaches (TSN for layer 2 and DetNet for layer 3) independently. This leads to solving individual scheduling problems (e.g. specific traffic profiles). Nevertheless, time-critical applications such as remote control and industrial automation demand stringent requirement considering end-to-end path, not only particular layers or sections of the network. Huang et al. [67]

UNIVERSIDAD DE MÁLAGA

present a novel architecture that includes a method for end-to-end scheduling. One similarity to the aforementioned works is the use of CQF and CSQF mechanisms in TSN and DetNet, respectively. To achieve end-to-end scheduling, additional modules are necessary in the architecture. First, the interconnection of TSN and DetNet through Segment Routing IPv6 (SRv6), which is in charge of encapsulating and decapsulating packets to make them understandable for both approaches. Second, the network slicing technique, enabling the creation of virtual networks on top of the same physical structure (see the next Section for further details). Additionally, cycle alignment to ensure the same duration in the different queues of the network and global computing to be able to generate a network configuration that meets the traffic requirements. They also propose traffic analysis with AI techniques, including traffic classification, flow prediction, and QoS extraction, which can be used to configure the network according to TS and non-TS flows. Similarly, a reinforcement learning-based solution is presented in [68] for dynamic flow scheduling in deterministic asynchronous networks. In this case, the operation includes a 5G network, where the scheduled flows have characteristics comparable to four critical 5G QoS Identifiers (5QIs) included in 3GPP standards (for example, used for discrete automation). These results demonstrate the potential of learning techniques to take advantage of the benefits of TSN and DetNet on wireless networks such as 5G, as proposed in this thesis.

TSN - Time Synchronization

Time synchronization is a fundamental aspect of a network, which can directly affect many network operations, such as packet timestamping, monitoring, configuration, etc. It contributes to the reliability, efficiency, and security of various industries and applications [69]. Subsequently, time synchronization can be vital for successful communications. However, it should be noted that in certain scenarios, it is challenging (for example, due to clock limitations or prohibitive implementation cost) to achieve accurate synchronization across all network devices. This is the case for asynchronous TSN networks, where additional efforts are made to meet the requirements of industrial traffic flows. For example, the authors in [70] evaluated the capacity of these networks to accommodate five typical services in the industry by providing the formulation of the Flow Allocation Problem (FAP). They highlight the impact of network topology and traffic type on the results obtained (e.g. maximum delay o link utilization).

There exist several protocols and methods to address the time synchronization problem depending on the need. For example, industrial automation demands accurate time synchronization for coordinated operation and real-time control [71]; In telecommunication networks, it is also important to ensure seamless handovers between different base stations. Some of the most popular synchronization protocols and systems are as follows.

- Network Time Protocol (NTP) [72]. Networking protocol for time synchronization in networks with variable latency (e.g. computers in the Internet).
- Simple Network Time Protocol (SNTP) [73]. Simplified version of NTP designed

UNIVERSIDAD DE MÁLAGA

for time synchronization in low-resource devices (e.g. router).

- Global Positioning System (GPS) [74]. System that provides location and precise time signals through a satellite network (e.g. base stations in mobile networks).
- Precision Time Protocol (PTP) [75]. Protocol defined by IEEE 1588 for high-accuracy time synchronization (e.g. industrial automation). It achieves sub-microsecond accuracy by compensating network delays.

The standardized time synchronization mechanisms briefly described above together with other non-standard mechanisms are reviewed in [76]. The author compares both categories, highlighting essential features such as accuracy and the synchronization interval. For example, in the case of time synchronization in TSN a particular PTP profile is used to achieve the accuracy needed for deterministic communications (e.g. industrial automation). PTP uses a master-agent architecture, where agents receive time-stamped messages to calculate the relative offset between their clocks and the master clock, well-known as *Grandmaster* (GM) Clock, in which the best option is automatically selected based on parameters such as accuracy and designated priorities.

PTP is commonly adopted in Local Area Network (LAN) [77]. Furthermore, it can be the time synchronization protocol in Wireless Local Area Network (WLAN), fulfilling most of the requirements of typical applications [78], [79]. Furthermore, mobile networks are becoming relevant in factories, and therefore mechanisms for distributing timed messages are also being studied [80]. The integration of wired and wireless networks devoted to industry is explained in detail in the next section.

2.1.3 Hybrid Networks for Industry

There is an increasing demand to bring the benefits of wireless communication, such as mobility or the avoidance of cabling, to industrial environments [81], which are mainly wired systems. Therefore, the trend is to use a hybrid approach, combining wired and wireless communications [82]. However, this brings with it challenges that need to be addressed, such as end-to-end QoS management. This includes not only the previously explained features, such as traffic scheduling and time synchronization, but also other features, such as traffic priority or interoperability between domains. The design of a hybrid network is based on the needs of the application, considering factors like traffic profile, mobility, scalability, etc. Research on the extension of wired networks (commonly TSN, Profinet, Profibus, etc.) with wireless standards such as IEEE 802.11 (e.g. WiFi) and 802.15 (e.g. Bluetooth) for real-time communications [83] is not new. The authors of [84] demonstrate that these wireless technologies can serve the requirements of certain applications with control loops of tens of milliseconds. However, the expected QoS could not be met (for example, results with low throughput or, more crucially, high jitter values) when there is interference in the system. A real demonstration is also presented in [85], achieving extremely low-latency (150 μs) with three TSN hops, including one wireless. However, if more complex topologies are considered, a dynamic configuration of the network is required to meet the QoS requirements of TSN flows. Consequently, most

UNIVERSIDAD DE MÁLAGA

wireless solutions cannot meet the requirements of industrial applications (e.g. discrete automation) that demand strict real-time communications with cycle times in terms of low milliseconds [86] in real environments with a certain number of TSN flows.

On the side of wired networks, TSN is a set of standards that improves the capabilities of the Ethernet network to support deterministic communications (see Section 1.1.1). From the perspective of wireless technologies, 5G enables the use of mobile networks in new sectors (e.g. industry) by offering high data rates and ultra-low latency, leading to a considerable enhancement in the QoS provided (see Section 1.1.2). Both are potential technologies with the necessary features to meet stringent requirements for industrial use cases. Although wireless technologies are not currently widely consolidated in industries, their annual growth has been increasing in recent years [87]. Therefore, the integration of TSN in the 5G architecture can be considered an enabler for future automation in hybrid networks.

TSN over 5G integration

As introduced above, TSN is potentially one of the leading enablers for providing reliable, low-latency, deterministic communications in the industry. Furthermore, 5G offers the possibility of wirelessly extending these networks to take advantage of its benefits. Through techniques such as network slicing, it is possible to provide QoS that meets the demanding requirements of industrial traffic. Numerous related researches dealing with different aspects of TSN integration over 5G [3] can be found in the current literature. On the one hand, work related to the current state of the standards and the level of integration achieved will be presented. On the other hand, the main challenges are introduced, i.e. time synchronization and traffic translation between the TSN and 5G domains. Finally, some use cases are studied.

From Release 16 [28] onward, 3GPP defines the fundamentals to support Time-Sensitive Communication (TSC) and integrate TSN and 5G, including the functionalities and architecture. This includes the definition of new entities called TSN translators (TTs) placed at the edges of the 5G network, that is, interconnection with TSN. The Device-Side Time-Sensitive Networking Translator (DS-TT) connected to the 5G UE and the Network-Side Time-Sensitive Networking Translator (NW-TT) connected to the UPF [30]. The main objective of the TTs is to integrate TSN and 5G by configuring various parameters, making the 5G network a logical TSN bridge. In addition, the Time-Sensitive Networking Application Function (TSN AF) [31] is responsible for the exchange of information in the control plane between 5G, called 5G System (5GS) in this context, and the TSN network, specifically CNC. First, this information (e.g. details of Ethernet ports or traffic profiles) is employed to configure the TSN network to support QoS requirements. Furthermore, the 5GS calculates the residence time on the 5G network, considering ports and traffic profile. Second, the 5GS must be tailored to the TSN traffic flows (see Table 1.2). In this way, the TSN AF declares the QoS parameters (e.g. delay) depending on the information received from the CNC. Then, this information is sent to the PCF, which is in charge of mapping the TSN QoS parameters to be understandable for the

UNIVERSIDAD DE MÁLAGA

5G network. The use of TSC Assistance Information (TSCAI) is also introduced in the standards, which can be helpful in scheduling efficiency. The following articles look at such integration from different perspectives, e.g. focusing on end-to-end latency [88], [89] or traffic scheduling [90], [91].

Larrañaga et al. [88] analyze the integration of 5G and TSN focused on low latency and determinism for industrial applications, highlighting the contribution of the 5GS bridge and TTs to the results. They present an example in which a PLC is connected to a mobile robot through a 5G logical TSN bridge. The delays in the TSN bridge and the necessary bandwidth are calculated on the basis of the flows requirements. They assume that a 5QI is configured and associated with the QoS parameters in advance, and more importantly, a packet can be transmitted immediately, that is, the port gate is always open for this kind of traffic. In addition, they consider standardized values (e.g. priority) to define the QoS parameters of each flow. Note that the 5GS delay estimation and parameter mapping are not defined in the standards. Therefore, it must be addressed in the implementation.

Similarly, the authors of [89] introduce the integration of the TSN and 5G systems, focusing on how to use the information effectively to improve end-to-end latency. In this case, they simulate a scenario with multiple TSN streams, which are also assigned to a 5G configuration following the same process. Furthermore, they present a comparison between the performance results with the 5GS acting as a logical bridge and replacing it with Ethernet switches, removing the wireless links. Despite the results in the fully wired scenario being better, it is demonstrated that the 5GS is able to support good performance (for example, 1 ms delay) by offering QoS guarantees, which is enough for a considerable number of industrial applications. Putting all together, the main challenge appears when the 5G RAN cannot match the traffic periodicity of the TSN flows, since the use of Orthogonal Frequency-Division Multiplexing (OFDM) demands separated time slots for transmitting and receiving data, which also determines the minimum latency capacity of the network.

Other works address the integration of the two systems from the point of view of traffic scheduling when there are several traffic flows or users simultaneously. To this end, the case of how to allocate resources for periodic TSN flows is studied [90], even when there is traffic in the 5G network itself simultaneously [91]. Both papers present results on how different KPIs (throughput and delay) behave as the number of flows or users increases through simulations. However, these results do not take into account full integration, as they do not make use of fundamental elements such as 5GC (not only considering UPF) or TSN translators. Therefore, other challenges such as the implementation of TSN translators, time synchronization, and dynamic reconfiguration must be considered for successful end-to-end communications.

UNIVERSIDAD DE MÁLAGA

TSN over 5G - Time Synchronization

IEEE 802.1AS [10] is the reference standard for time-sensitive applications related to time synchronization. Procedures and mechanisms to meet wander (clock phase variations below 10 Hz), jitter (clock phase variations greater than 10 Hz), and synchronization requirements are defined in depth. Consequently, IEEE 802.1AS, which is a particular profile of IEEE 1588 [75], serves as the basis for most research in this area. Time synchronization with respect to TSN over 5G is a challenging task that is investigated by applying various methods and techniques [92]–[95].

The work by Garbugli et al. [92] demonstrates the possibility of running TSN applications in a virtual environment by designing a novel approach, which consists of the creation of virtualized PTP clocks for the correct operation of the network frame scheduler and the use of para-virtualization techniques. The results show that the solution meets the ultra-low latency constraints of critical applications running on virtual machines. The limitations in terms of time error introduced by the 5G network due to frequency offset are studied in [93], especially when transmission delay is high. The authors propose a method for estimating the frequency offset to reduce the time error. Further investigation is presented in [94], where it is shown that even small frequency offsets can introduce considerable synchronization error, having a devastating impact on industrial applications. The work in [95] shows a comparison of synchronization over the wireless link using Wi-Fi and 5G, in which 5G has better performance (e.g. delay and jitter). In addition, the importance of hardware-timestamping to obtain reasonably lower delay is proved in the results.

TSN over 5G - TSN translators

The TSN translators are responsible for allowing the interoperability of both domains, TSN and 5G [30]. This includes gPTP support for time synchronization in addition to other functions related to traffic itself, such as QoS mapping to maintain the requirements through the entire system. In this context, some articles study techniques for carrying out this procedure dynamically [96]–[99].

On the one hand, the authors in [96] present a solution for mapping and scheduling based on the requirements of different applications. However, the focus is on how to host Virtual Network Functions (VNFs) on available resources. On the other hand, Satka et al. [97] propose an algorithm to perform QoS mapping for TSN-5G flows, considering predefined 3GPP flow identifiers. In addition, a first implementation of this technique in the NeSTiNg simulator is presented in [98]. Nonetheless, perfect clock synchronization is assumed throughout the network, which could lead to inadequate results in a real environment where this does not occur. A more comprehensive approach is presented in [99], in which an algorithm to dynamically assign the QoS mapping and a method to allocate the resources in a flexible way. They demonstrate that the proposal has a better performance in terms of KPIs (e.g. throughput and delay) when the network is highly loaded. Note that the results are centered on the effective use of Resources Blocks

UNIVERSIDAD DE MÁLAGA

(RBs) for an uplink connection. Consequently, other network parameters corresponding to other layers such as Packet Data Convergence Control (PDCP) or Radio Link Control (RLC) are not considered to meet the requirements of end-to-end traffic.

Until now, these works have the potential to facilitate the development of release versions of TSN translators. However, currently there are no solid commercial or open source implementations of the TSN translators that are compliant with the standards [28]. Consequently, further research is necessary to achieve complete integration.

2.1.4 Testbeds

It is possible that due to the lack of commercial or open source solutions for key elements in the integration of TSN over 5G (e.g. TSN translators), there are also not many testbeds available where it is possible to evaluate related solutions in this field [100]. Some of the existing evaluation platforms and testbeds are presented below, in both simulation [101]–[103] and real [104]–[108] environments.

On the one hand, simulated environments can be used to pre-study the feasibility of certain key aspects. For example, an analysis of the relevant QoS parameters of eight TSN traffic classes is presented in [101], where the importance of having joint mechanisms to achieve a joint QoS model is highlighted. To this end, the key elements in the interoperability of the two domains are fundamental. The work in [102] adds a translator prototype to a 5G node using OMNET++, which focuses on the delay and packet error rate. However, vital aspects such as time synchronization and self-management are not included. In the process of creating a platform that can help in the development and validation of research work related to 5G TSN integration, the authors of [103] present a solution based on open-source projects (UERANSIM for the simulation of 5G UE and RAN, and free5GC as 5GC) to implement the data plane compliant with Rel.16 [28], including key features such as Ethernet PDU sessions, TSN translators, and 5GS bridge management.

On the other hand, real environments that include physical equipment are essential to validate the results and assess the current state toward full integration of TSN over 5G in real environments. Articles such as [104], [105] serve as a precedent, since they use prototype components or older technologies (e.g. 4G) to achieve this goal. For example, by demonstrating features (e.g. 5G network slicing) that allow these networks to function correctly for applications like discrete automation. Going a step further, more extensive evaluations are performed with commercial equipment, allowing us to know the capacity of the 5G network integrated in an industrial TSN network in real-world environments. In particular, TSN features such as traffic shaping and scheduling are evaluated in an industrial environment in [106]. In this case, a 5GC prototype developed by Ericsson together with an antenna module operating at 28 GHz (mmWave - N261) is used to demonstrate that the network is able to support the demanding requirements of industrial automation. Similarly, the works of [107], [108] study how to accommodate industrial use cases with commercial equipment. In the former, the authors evaluated the

UNIVERSIDAD DE MÁLAGA

performance of two 5G system deployments using Nokia RAN 4G and 5G NSA, in terms of Round Trip Time (RTT). In addition, the setup also works with 5G SA, providing better performance. The second is a more in-depth study, including experiments to show the capabilities of TSN over 5G in scenarios where background traffic may exist and prioritization of critical traffic is possible. In this case, the 5G system is based on the Keysight UXM 5G Wireless Test Platform that enables to emulate a 5G network, in which the traffic is sent between the TSN devices to control a ball-balancing table used as visual demonstrator. In both cases, the benefits of these networks for hosting industrial traffic are demonstrated, allowing them to meet the requirements of different types of traffic, even when background traffic is present.

The testbed on which the work of this thesis has been developed (see Section 3.1) is a combination of the equipment from [107], [108], including two options for the 5G network. The first option consists of the Nokia RAN (5G NSA and SA) and the Open5GS core network, and the second option consists of the Keysight UXM 5G, which together with the TSN and time synchronization equipment forms the TSN over 5G testbed. On the other hand, while these works do not include the TSN translators, this thesis develops a prototype of them to obtain a behavior closer to a future real environment that is compliant with that defined by the 3GPP.

2.2 Enabling Techniques and Methods for Reconfiguration

Configuring a network to meet all traffic needs is a complex operation that cannot be performed manually due to the considerable number of existing traffic patterns and network configurations, especially in TSN over 5G networks, where this thesis work is focused. Consequently, there is a tendency to apply techniques, methods, and approaches that focus on network automation [109], which are contemplated as strategies not only to provide self-configuration, but also self-monitoring, self-healing, and self-optimization according to a certain service level agreement (SLA). They can be implemented with human oversight or even without human intervention.

2.2.1 Self-Organizing Networks

Self-Optimizing Networks (SON) is a technological advancement created to streamline and accelerate planning, configuration, control, and recovery from mobile network failures. The characteristics and procedures of SON have been defined and detailed by entities such as 3GPP [110] and Next Generation Mobile Networks (NGMN) [111].

The deployment of Long-Term Evolution (LTE) networks forced a rethinking of network management [112] due to the exponential growth of the volume of network data, the addition of new traffic patterns, and the large number of cells to be managed. The initial approach of SON [113] was to facilitate the deployment and commissioning of the network. However, they can also assist in the subsequent network tuning required to efficiently manage coverage and capacity [114] according to the number of connected

UNIVERSIDAD DE MÁLAGA

users and reduce interference between near-cells [115]. One of the most widespread uses of SON is to support handovers. Two clear examples are the works presented in [116], [117], where optimization of the two main parameters of the HO (i.e. Hysteresis and Time-To-Trigger) is sought through simulations and where an improvement of the network is demonstrated in both throughput and delay. In addition, some learning techniques such as Reinforcement Learning (RL) are used to learn the behavior of the network and help with such HO parameters [118], [119], similar to the work in this thesis. There are also key characteristics associated with SON. For example, the Automatic Neighbor Relations (ANR) feature is used to automatically manage neighbors cells by creating the necessary links between them.

In summary, SON has multiple benefits for network operators [120] that can support improving the QoE of network users [121]. Despite the challenges involved in the implementation of these techniques in mobile networks [122], SON mechanisms have evolved to help with the management in 5G networks [123]. In particular, in heterogeneous or hybrid networks, such as the one proposed in this thesis, TSN over 5G, where nodes of different types are connected to each other, the need for these self-organization techniques is even greater [124].

2.2.2 Closed Control Loops

A Closed Control Loop (CCL) is designed to autonomously adjust a system to maintain a specific state or a set point without the need for human intervention. This approach is based on feedback from the system. In contrast to open-loop control, which lacks self-regulation and usually necessitates human involvement, closed-loop control operates independently. In addition, proper tuning of a closed-loop control system is crucial for optimal system performance.

CCLs can be applied to any type of system, but are particularly effective in complex systems. However, there is a trade-off between energy and time that must be adapted to the particular type of system [125]. Focusing on communication networks, there are different works that involve CCL for different purposes. The authors of [126] propose a prediction and control scheme for a network based on queues. However, the two automotive application examples presented use a wired communication link. In contrast, the works in [127], [128] develop CCL in wireless networks, where advantages such as increased system robustness and performance are demonstrated. On the one hand, a novel timing scheme is proposed in [127], avoiding the need for clock synchronization at individual nodes. On the other hand, in [128] an attempt is made to extract a configuration to cope with node failures. Both approaches are used in this thesis work, although there are slight differences since the type of network we are dealing with requires precise synchronization and, therefore, the use of hardware clocks. In addition, this work deals with end-to-end connections, so possible failures must be dealt with in any part of the system rather than in individual nodes. Similarly, an algorithm is presented in [129] to deal with the presence of packet loss and delay in the network, where the objective is to ensure a certain stability in the system where real-time communication over Ethernet

UNIVERSIDAD DE MÁLAGA

is used. Nonetheless, the testbed and communication technologies used (i.e. Bluetooth) are not comparable to those in this work. The authors discussed in [130] the automation aspects of multi-domain environments utilizing multiple CCLs for comprehensive end-to-end service management. They highlight the importance of intelligence to provide the expected requirements, as well as some key concepts and technologies such as Intent-Based Networking and Zero-Touch to achieve this, which are defined in the following sections.

2.2.3 Intent-Based Networking

Intent-Based Networking (IBN) is a concept commonly applied in business environments that captures the intent of the user and transforms it into policies that can be automated and continually implemented on the network [131]. The main objective is to monitor the network repeatedly in order to allocate the expected result. Additionally, it can also be applied to communication networks to provide flexibility, agility, and simplicity to network configuration with minimal external supervision, which can be assimilated with the CCL approach. Therefore, it is a fundamental component for network management and orchestration [132]. IBN has five main components to create the closed loop needed for this type of automaton: profiling, translation, resolution, activation, and assurance of the intents, which are extensively detailed in [133].

One of the areas where IBN is widely used is Software Defined Networking (SDN), whose objective is to facilitate the implementation and deployment of network services in a deterministic, dynamic, and scalable way, avoiding the need for the network administrator to manage these services at a low level. However, this abstraction and automation generate new challenges in the network. Ujcich et al. [134] propose a framework to follow the evolution of an intent while the state of the network and other intents change due to time passing. Similarly, the authors of [135] define an approach for SDN networks, where requests are not only monitored, but the expected QoS level is also automatically maintained, trying to minimize human intervention throughout the process as much as possible. This approach is very close to the one proposed in this thesis work. In line with this trend, IBN is also being implemented to complement the next-generation SDN. For example, the authors of [136] present a new approach of the P4 language, which allows handling of the P4 programs running on the devices on demand.

In the same way, IBN can be extrapolated to Next-Generation Networks (NGNs), such as the 5G network. Due to the immense number of applications demanding a set of individual requirements in addition to the complexity of the networks, manual configuration is prone to errors. Therefore, an intent-based network application [137] can help resolve these potential errors by knowing in advance the requirements of the active traffic sessions on the network. However, translating these intents, which may even include high-level requirements, into a specific configuration for the various network components is an arduous task. For example, Campanella [138] proposes the use of the ONOS SDN controller to assist this task. Some challenges are also discussed, such as the level of intent abstraction, which can be critical depending on the requirements of the application or the final user, since the network can be configured at a finer level of detail, the

UNIVERSIDAD DE MÁLAGA

more accurate the traffic intent and requirements are. In the particular case of 5G networks, the goal is to meet end-to-end requirements [139]. In this way, features such as elasticity and replication are crucial in creating a dynamic system capable of adjusting to the changes required in each of the elements that make up the complete pathway of the connection. One of the key features of the 5G network to enable this is network slicing, which allows the creation of logical networks that can provide different attributes (e.g. low-latency or high throughput). The authors of [140] define an approach to fully automate the network slicing process in real time. It includes the entire lifecycle, from creation and configuration to finalization. In addition, they propose a mechanism to manage the assurance and scalability of network resources. Another idea is presented in [141], where the authors consider the potential changes that come from an external environment. In this case, reactive configurations are used to extend the Intent-Based Network Modeling (NEMO) language developed in the NEMO project, which allows the network administrator to automatically update the configuration. Note that a minimum human operation is required here.

In summary, IBN offers relevant features (e.g. agility and robustness) that can be used to meet the stringent traffic requirements of complex networks such as 5G networks. The main architectures of the system and key techniques are surveyed in [142]. Additionally, three commercial demonstration platforms are presented, together with industrial products for IBN. Finally, the authors discuss the use of learning techniques (e.g. Artificial Intelligence) to improve the required operations in mobile networks (e.g. 6G), which is not yet fully adapted for IBNs. Other works present different methods and learning techniques with respect to IBN[143], [144]. Although both papers focus on the dynamic and flexible adaptation of system resources to meet objectives, they have different approaches. On the one hand, the authors of [143] propose a novel prediction model that aims to learn the relationship between prediction and the objective using deep learning. Similarly to the work in this thesis, they seek to obtain suitable functions in various situations, which can then be used to reduce human intervention. On the other hand, the use of closed loops of self-learning with AI/ML to enable automation is defined in [144] to dynamically adapt the system to user demands, which is also applied as part of this work. However, none of these papers focuses on the complex issues of 5G networks.

2.2.4 Zero-Touch Networks

The operation of network management has become more complicated as the complexity of the networks and the number of connected devices increases. A clear example is the Internet of Things (IoT) networks [145], where hundreds of new devices must be connected and configured. Consequently, it is an impossible task to manually manage. In order to save time, reduce complexity, and reduce the costs associated with network development, the Zero-Touch concept emerges. It refers to the automation and simplification of processes to minimize or even eliminate human intervention in network operations. There are some variants depending on the field to be applied. For example, Zero-Touch Provisioning (ZTP) [146], also known as Zero-Touch Deployment, consists of the onboarding and

UNIVERSIDAD DE MÁLAGA

deployment process of new devices, including its configuration in an automatic manner, which is commonly used in IoT networks [147] and network environments [148].

In the case of Zero-Touch Network and Service Management (ZSM) [149] it is intended to go one step further, achieving a fully autonomous network, with self-monitoring, self-healing, self-optimization, and self-reconfiguration capabilities based on policies and rules imposed by traffic requirements without any human intervention. In traditional networks, there is a clear trade-off in network operation between reliability, scalability, and efficiency, where only two of them can be achieved. However, all these characteristics can be achieved in Zero-Touch networks, where the operation of the network is entirely intent-driven [150]. The main claims of a Zero-Touch network are as follows.

- There is complete automation in each network operation performed. The only manual process that exists is the declaration of traffic intent.
- Any changes that are applied to any network element are declarative and are only performed taking into account the high-level intent of all active connections in the network.
- If the network exhibits unexpected behavior, an automatic network reconfiguration is performed to satisfy the traffic requirements. Note that the system does not allow any operation that could degrade the expected behavior.

Taking into account the characteristics associated with ZSM, this concept is applicable not only to IoT [151] to improve management efficiency, but also can be extended to more complex networks such as heterogeneous industrial networks [152] or cellular networks [153]. Therefore, in the case of 5G and beyond networks, ZSM can be a key enabler to obtain an autonomous network system with the necessary features to minimize human intervention. The authors of [154] reviewed the solutions available for these types of network, including essential components and additional services for automation (e.g. lifecycle management or security aspects). In particular, this concept can be applied to 5G network components individually (e.g. RAN or network core) or as a whole, that is, the complete end-to-end path. This approach is surveyed in [155], focusing on network automation solutions for 5G and 6G networks.

There are works in the current literature that partially represent the objective of this thesis in relation to ZSM. On the one hand, articles related to articles that study different services related to the mobile network. The authors of [156], [157] present a closed-loop to assist in the operation (e.g. monitoring and decisions). Both works use ML as learning solutions, specifically focused on security aspects of 5G networks, such as detection and mitigation of attacks. For example, the algorithms developed in [156] analyze different events to support attack predictions to be able to counter them, while a dynamic selection of a suitable network configuration is carried out (the algorithm selects a network slice instead of the 5G network configuration in this case) in [157]. In addition, the Service Level Agreement (SLA) concept is used in the latter as a target to detect service degradation or violations, which is also applied in [158] together with learning

UNIVERSIDAD DE MÁLAGA

techniques to generate predictions regarding network slicing performance.

On the other hand, papers consisting of the mobile network components. A clear example is the automation of the RAN, including different aspects such as radio resource allocation [159]–[161] or management and self-deployment [162]–[164]. In the case of the radio resource allocation, the authors of [159] solve the problem of allocation of resources by focusing on multicast services and, therefore, their requirements. Additionally, the autonomous network management is carried out in [160] according to QoS requirements. In the same way, the objective in [161] consists of automatically meeting the QoS requirements while the RAN nodes are shared. These papers propose several ML models to address the problem, of which the first two are evaluated by means of simulations, while the latter one uses a real setup. In the context of management and self-deployment, an algorithm for smart user association with base stations is presented [162]. Another is considered in [163], where a collection of measurements is used to estimate performance and also to help create a location heat map. The authors of [164] study the automatic deployment and configuration of the Cloud-RAN chain, highlighting the stringent requirements in terms of latency expected in the transition to the 6G network that need to be met. Again, the works use AI/ML-driven approaches, using simulations to evaluate the results. This insight is fundamental to pursuing the objective of this thesis. However, it is not sufficient for more demanding applications that require an approach that considers end-to-end communication.

2.2.5 Network reconfiguration

In general, there are several techniques and methods that can be considered to perform network reconfigurations. In addition, the combination of different techniques and approaches from different works is essential to perform these reconfigurations automatically and with minimal or even no human intervention. Especially in complex networks such as TSN over 5G on which this thesis is focused.

In the case of TSN networks, there exist a multitude of publications on the network configuration. In particular, implementing solutions to handle flow configuration and traffic scheduling, which mainly depend on the application and, therefore, the characteristics of the traffic. The paper in [165] shows that a centralized configuration model is vital to handle complex and real-time applications, since streams can be configured using a global perspective of the network. In addition, the influence of specific network parameters should be considered during the initial configuration, as well as the traffic sent to perform successful reconfigurations [166], especially in realistic industrial scenarios using real equipment. For example, the authors of [167] implement a CNC to schedule time-triggered traffic and configure the TSN network accordingly, validating the implementation through an automotive use case. Going one step further, a configuration solution is proposed in [32], [168]–[170], not only for networks with TSN equipment, but also for other technologies such as SDN and OPC UA, highlighting the complexity of combining different technologies and the need for standardization to achieve this successfully.

UNIVERSIDAD DE MÁLAGA

In the same way, there are articles that discuss the configuration of wireless networks [171]. In particular, the configuration of 5G networks is studied from different perspectives. First, the configuration of network slices, which allows the creation of isolated networks on top of a common physical infrastructure, to meet the diverse and demanding requirements of new applications [33] is a popular topic in the current literature. Additionally, in this context, it is essential to be able to perform dynamic reconfigurations due to the variation in the required requirements over time [172], which can also be extrapolated to other cases. Second, since mobile networks do not offer enough flexibility to adapt to changes caused by traffic, reconfiguration related to its architecture becomes important. Sabella et al. [173] present an architecture that provides flexibility and reconfigurability to the network. They propose the use of sensors for predictions (e.g. interferences in a specific area) and, consequently, configure some parameters. In addition, algorithms make use of the information related to the traffic to improve performance. The results show how the delay of real-time applications is reduced without interfering with other KPIs. The User plane Function (UPF) is one of the key elements of the 5GC associated with the QoS provided to users, and therefore its configuration and placement (especially in mobility scenarios) are critical. The authors of [174] study this problem to avoid possible quality degradations based on ML prediction events to maintain the system in the desired state (i.e. when traffic requirements are met) and reduce the number of reconfigurations. Third, dynamic reconfiguration is also applied in the context of security, where CCLs are proposed to mitigate possible attacks and maintain user capabilities [175]; and in the case of dense networks in which new cells can be massively deployed and automatically adjusted to allow efficient data flow [176].

In general, the literature covers the crucial aspects of reconfiguration in TSN and 5G networks independently. However, considering the integration of both technologies, there is a lack of research in terms of dynamic configuration. Considering wireless TSN networks with other technologies (not including 5G), there exist works which address some aspects. For example, the authors in [177] present a scheduler for dynamic configuration of traffic flows and a CCL to assist in real-time management of a TSN network combined with Wi-Fi connections. They demonstrate successful time synchronization and bounded low latency. In addition, a wireless TSN (using radio frequency dongles) network is considered in [178] to implement a CUC entity that aims to configure wireless end stations according to particular requirements. However, these articles approach reconfiguration from a network management perspective rather than from the perspective of configuring specific parameters to maintain traffic quality over time. Furthermore, this problem is not investigated considering a TSN over 5G. For this reason, this thesis aims to contribute to the lack of work in this area.

All things considered, the combination of the techniques presented is vital to achieving the objective of dynamic reconfiguration to meet the traffic requirements in TSN over 5G networks. In addition to these methods, the involvement of learning approaches can improve performance by adding intelligence to the system [179]. For example, by making predictions of failures or possible deviations from expected requirements and

UNIVERSIDAD DE MÁLAGA

adjusting parameters to increase the time in the state where the network is meeting those requirements. However, the capabilities and limitations [180] of the different learning techniques must be taken into account when trying to achieve full automation, even without human intervention, of the system following the Zero-Touch approach.

2.3 Learning Approaches

This section provides a study of different approaches for creating representations or models, including observations such as actions, events, and interactions that describe the behavior of a system. These observations can be used to learn and make decisions about system management. For example, a 5G network model can be used to monitor the QoS of end-to-end traffic sessions and apply a network reconfiguration to meet the requirements over the time of different traffic sessions. The works in the current literature identify two main approaches for developing and learning interaction models from observations, which are applicable to network trace analysis. Firstly, conventional machine learning-based Artificial Intelligence (AI) and Machine Learning (ML) strategies, such as reinforcement learning [181] and learning automata [182]. Secondly, approaches to automata/model learning [183], [184], which are explored within the context of formal methods. It is important to highlight the ambiguous use of terms, distinguishing between the AI method known as *learning automata* and the formal method referred to as *automata learning* (or model learning).

2.3.1 Behavioral Modeling

A behavioral model is an abstraction that represents the behavior and dynamic aspects of a system, commonly defined as a diagram. This diagram comprises a set of states and transitions known as the state-transition diagram. Behavioral models have been employed in various fields [185] until now, including psychology and economics, to understand people's decisions, and also in computer science and engineering to understand complex systems such as automobiles and computers. In addition, there are several types of behavioral model with different characteristics (e.g. number of states, or transition dependency) depending on the application.

In the case of computer science and engineering, they can be used from different perspectives. First, the models focused on the capture and specification of the desired behavior of software systems [186]. Second, models to verify hardware design through description languages such as Verilog [187]. Third, they are used to design and analyze the behavior of different types of systems (e.g. embedded, cyber-physical, and distributed), including the verification of meet timing, performance, or fault-tolerance mechanisms [188]. Furthermore, these models are associated with different learning techniques and methods such as AI/ML, automata learning, inference, verification, and validation, etc. These concepts complement each other, allowing us to represent, analyze, and reason about dynamic system behavior of complex systems. For example, in the case of mobile networks, the authors of [189] focus on the representation of traffic behavior, using them to have an

UNIVERSIDAD DE MÁLAGA

abstract view of congestion control in mobile ad hoc networks, allowing the representation of the network based on the congestion status of its nodes. Another example is presented in [190], where the authors demonstrate the potential of a behavioral model applied for 5G power amplifiers. Therefore, the use of behavioral modeling can significantly contribute to achieve the objective of this thesis.

2.3.2 Artificial Intelligence and Machine Learning

On the one hand, the concept of Artificial Intelligence (AI) can be understood as machine intelligence. Initially, the tasks that could be solved using AI were not very complex. However, the number of applications of this technology is growing [191], and recently it is capable of achieving goals such as intelligent decision making and problem solving in complex systems [192]. On the other, Machine Learning (ML) is considered a subcategory of AI that enables the system (machine) to learn and improve decisions automatically. Instead of explicit programming, ML applies algorithms such as supervised, unsupervised, semi-supervised, and reinforcement learning to make decisions. An algorithm is trained on data to obtain a model as output. Consequently, the more data used for training, the better the model obtained. In summary, both terms are closely connected, with ML being an AI application to learn from data automatically.

In general, different types of ML algorithms focus on specific applications and objectives. In the context of complex systems such as TSN and wireless networks, ML is applied to a wide variety of problem solving and decision making. One of the best known ML applications in TSN is the scheduling of different traffic patterns [193], which can also focus on specific features such as end-to-end latency optimization [194]. In the case of wireless networks (e.g. 5G and beyond), the ML techniques mentioned above, in addition to new techniques such as deep learning and federated learning, are reviewed in [195]. For example, supervised learning could be used for classification tasks such as service identification [196], unsupervised learning to identify anomalies [197], and reinforcement learning to optimize resource allocation [198]. Furthermore, two similar approaches using reinforcement learning for TSN over 5G networks are studied. First, to assist with the flow allocation problem focused on asynchronous TSN [199]. Second, schedule time-triggered flows to guarantee end-to-end latency [200]. Both works have been carried out through simulations. These works show the importance of AI/ML techniques in complex tasks. However, most of them are carried out in networks with simple topologies (e.g. 2 TSN nodes directly connected) or through simulations that have not been contrasted with a real network.

Reinforcement learning and learning automata are established AI/ML methods that acquire knowledge through observed interactions. These methods specialize in translating environment states into actions that aim to maximize a specific reward. This is done through an iterative process that concludes when the maximum reward is reached. These rewards may include system efficiency or resource utilization. In addition to the approaches seen above, algorithms have also been applied in environments related to this thesis, such as wireless network configurations [201], [202]. It should be noted that,

UNIVERSIDAD DE MÁLAGA

throughout their operational lifetime, algorithms in both AI/ML methods maintain awareness of the state of the environment. However, the intermediate states are not directly represented in the resulting model; instead, they are embedded within the derived probabilities. Subsequently, explaining a reconfiguration decision becomes challenging. Therefore, the main problem with the use of ML in this context is the inability to guarantee the accuracy of the decisions made.

Notwithstanding, the concept of Explainable AI (XAI) [203] is growing significantly in recent years. XAI is a set of methods and processes that enable the results of ML to be understandable to humans, i.e. to explain and justify the decision and actions taken. The key aspects of XAI are transparency, interpretability, and explainability. As a result, XAI is being applied especially in applications where these aspects are crucial. Industry 4.0 is an example [204], in which XAI can improve the system and their applications and contribute to product quality (for example, by supporting the development of a digital twin to test the system before real implementation). In addition, it can also be applied in other areas, such as mobile networks. Guo exposes the main 6G related areas in [205], where similarly to the previous work, a digital twin that works in parallel with the system can optimize performance. Additionally, the development of understandable interfaces for human communications is presented alongside mechanisms to detect problematic situations (e.g. attacks). In the same way, explainability methods are operated in [206] to capture the cause of the prediction of SLA violations. These approaches are similar to those used in this thesis. However, this thesis focuses on optimizing the Radio Link Control (RLC) layer instead of the physical layer (PHY) and Medium Access Control (MAC), and on detecting deviations of observed KPIs from traffic requirements during a TSN session (instead of focusing on network slicing SLAs).

It is true that XAI promises great potential. Nonetheless, there are currently certain limitations and challenges that need to be addressed [207]. First, a considerable number of AI algorithms (e.g. deep learning) are complex and not easy to interpret. XAI offers limited coverage of explanations for decisions made by complex models. There is a trade-off with the performance, since achieving explainability may decrease model performance. Although there exist techniques to simplify the complexity, it can lead to a reduction in their predictive accuracy. Consequently, understanding the behavior of certain systems, especially complex systems such as 5G, can still be highly complicated and opaque for XAI, that is, the interpretability is not enough.

2.3.3 Model Learning

Based on the findings of Automata Theory [208], a Finite Automata (FA) is a mathematical model consisting of a finite set of states: a finite set of input symbols (known as alphabet), a transition function that specifies how the automaton transits from one state to another based on alphabet, and one or more designated initial and final (accepting) states. There are two main types of Finite Automata, Deterministic Finite Automaton (DFA), whether there is a unique final state (or transition) corresponding to an input symbol; and Nondeterministic Finite Automaton (NFA) if one or more transitions exist

UNIVERSIDAD DE MÁLAGA

from one state that matches an input symbol. Subsequently, each DFA is also an NFA. Finite Automata is used to identify (accept or reject input symbols) from a given language, focusing on the recognition of languages. Therefore, these machines are not applied to model systems.

In this context, Finite-State Machine (FSM) is commonly used to model systems with discrete and sequential behavior, where the system transitions from one state to another in response to input signals. Additionally, FSMs can be categorized into two types based on the nature of their outputs and transitions. First, Moore machines, in which output values are determined only by the current state. Formally, a Moore machine is defined by a 6-tuple $(Q, q_0, \Sigma, \delta, \lambda, F)$, where Q is the set of states, q_0 the initial state, Σ the input alphabet, δ the transition function, λ the output function, and F is the set of final (accepting) states. Second, Mealy machines, where output values are determined by the current state together with input symbols. Similarly, a Mealy machine is defined by a 6-tuple $(Q, q_0, \Sigma, \delta, \lambda, F)$, with the only difference that the output function λ also considers the input symbols of the alphabet Σ that triggered the transition.

In contrast to AI/ML, model learning, also known as automata learning, aims to build a state diagram model from a black-box system, utilizing two primary approaches: active and passive learning [209]. On the one hand, active learning involves querying the system to aid in state machine construction. For example, active automata learning has been applied to develop models for the IEEE 802.11 4-Way Handshake state machine [210] and the NAS layer in 4G networks [211]. On the other hand, passive learning is based solely on the observed behaviors (i.e. examples) of the system without direct interaction. An example of this is the application of reverse engineering to derive state machines for the File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP) protocols in [212] and [213], respectively. Unlike conventional AI methods, model learning builds a formal system model that can be formally analyzed in relation to the original system. Moreover, decisions in the original model based on exploration of the formal model can be justified mathematically. Thus, this thesis adopts the model learning approach to depict the evolution of the entire TSN over 5G system, where transparency and traceability are crucial, allowing for the logical explanation and comprehension of each decision by humans.

2.4 Evaluation and Challenges

The evaluation and challenges found in the literature are described in the following, highlighting three key aspects. First, we look at how different technologies can be integrated and what difficulties they present. Second, we ask how the different techniques and methods can be combined to achieve the objective set out in this thesis. Third, we present how learning techniques can be integrated into the proposed approach. In addition, the most relevant references are introduced along with key aspects in Tables 2.1, 2.2 and 2.3, in which the symbols used are defined below.

UNIVERSIDAD DE MÁLAGA

⋆ Indicates that the reference is significant for this thesis

✓ Indicates that the topic is addressed

△ Indicates that the topic will be addressed in future work

▶ Indicates that the topic is addressed and will be improved in future work

2.4.1 Networking Technologies Integration

The integration of networking technologies plays a crucial role in building robust and flexible infrastructures capable of meeting emerging demands for new applications and use cases. A clear example is Industry 4.0, where the interoperability of various systems, devices, and applications is needed, and therefore, integration is vital to support seamless communications between them. However, there are challenges that need to be addressed. Table 2.1 provides the networking technologies studied in relevant articles, including the two main challenges mentioned (traffic scheduling and time synchronization) and the environment used for the evaluation (i.e. through simulation or real equipment).

Table 2.1: Networking technologies proposals

References	Networking Technology			Challenge or Topic			Environment Type	
	TSN	5G	Other	Sch.	Sync.	Other	Sim.	Real
[92]	✓	✓		△	✓	✓	△	✓
[93], [94]	✓	✓			▶		✓	
[95]	✓	✓	✓		✓	△		✓
[96], [99] ⋆	✓	✓		✓		✓	✓	
[97]	✓	✓		△		✓	✓	
[98] ⋆	✓	✓		✓	△		✓	
[107] ⋆		✓	✓	✓		✓		✓
[108] ⋆	✓	✓	✓	✓	✓	✓		✓

TSN Time-Sensitive Networking
Sch. Traffic scheduling
Sync. Time synchronization
Sim. Simulation

In the case of hybrid networks combining wired and wireless networks, TSN and 5G are potential technologies to provide reliable and low-latency communications in the industry with the benefits of wireless networks such as mobility and cable removal. It presents two fundamental challenges that are being studied. On the one hand, time synchronization, where there are several feasibility studies, even in virtual environments [92], and it is presented how the impact of synchronization errors introduced by the 5G network can be detrimental to industrial applications [93]–[95]. However, there is no full integration according to [28], which includes the procedures necessary on the TSN translators. On the other hand, the development of TSN translators for which several approaches are presented [96]–[99]. Nevertheless, there is no commercial or open source implementation for TSN translators that is compatible with the procedures defined by 3GPP. Since the

UNIVERSIDAD DE MÁLAGA

focus of this thesis is not on the solution for a complete integration of both technologies, a temporary solution is described in Section 3.2. Subsequently, the testbeds available to evaluate the associated solution in this are limited [107], [108], whose equipment has been combined to build the testbed developed as part of this thesis.

2.4.2 Combined Reconfiguration Methods

There exist several techniques and methods that enable the possibility of providing specific requirements. For this purpose, the monitoring, configuration, healing, and optimization of the network are carried out automatically, with minimal or no human intervention. Table 2.2 identifies the techniques and methods applied in the most important works.

Table 2.2: Combined reconfiguration proposals

References	Reconfiguration Technique/Method				
	SON	CCL	IBN	ZT	Other
[114], [115]	✓				
[124]	✓				▶
[127]		✓			
[130] ⋆	✓	✓	✓	✓	
[135]			✓		
[139] ⋆	✓		✓	✓	
[154], [155]	✓	✓	✓	✓	✓
[165]					✓
[172]					✓
[173] ⋆					✓
[174] ⋆					✓

SON Self-Optimizing Networks
CCL Closed Control Loop
IBN Intent-Based Networking
ZT Zero-Touch

First, Self-Optimizing Networks (SON) is a concept commonly applied for the planning, configuration, and recovery of mobile networks. Although this concept was first used in LTE networks for tasks associated with the radio part [114], [115], it has evolved over time, enabling the management of hybrid 5G networks [124]. Second, a Closed Control Loop (CCL) allows for autonomous adjustment of the system to keep certain states on the basis of feedback. Although they can be adapted to any system, they are especially effective in complex systems such as wireless networks [127] or complicated tasks such as automation [130]. Third, Intent-Based Networking (IBN) aims to capture user intent and transform it into automatable policies on a continuous basis. It is commonly used to facilitate the management of SDN networks [135], which can be extrapolated to high-level 5G networks [139]. Finally, Zero-Touch Network and Service Management

UNIVERSIDAD DE MÁLAGA

(ZSM) intents to go one step further by achieving a complete autonomous network. It is applied to networks such as IoT and heterogeneous industrial networks. In the context of 5G networks, ZSM can help minimize human intervention [154], applying to individual components or as an entire network [155].

Putting it all together, the combination of these methods should be considered to perform automatic network reconfigurations with minimal (or even any) human intervention in complex networks such as TSN over 5G. There are works that address this problem separately for the TSN [165] and 5G [172] domains. In the context of 5G, the approaches followed in [173] and [174] are fully aligned with this thesis. They consist of the prediction of certain events to trigger a network reconfiguration that can fulfill the pre-establish traffic requirements. However, this thesis focuses on reconfiguration of certain 5G network parameters to meet TSN traffic requirements (based on KPIs such as delay and jitter), while these articles address the problem from the perspective of 5G network architecture. In conclusion, the current literature on reconfiguration of wireless TSN networks is scarce and even not researched for networks combining TSN and 5G.

2.4.3 Application of Learning Techniques

Diverse learning approaches are applied to create models that include observations (e.g. events) and represent the behavior of a system, which are used to learn and make decisions. This thesis focuses on the creation of a 5G network model (i.e. behavioral modeling) to monitor end-to-end traffic quality and apply network reconfigurations to meet the requirements over time. This model (called state-transition diagram) is an abstraction of the system to represent its behavior and dynamic aspects, and is composed of a set of states and transitions. It can be used for different approaches, such as software and hardware design, but also includes verification of the system timing, performance, or faults. There are two main groups of learning techniques that can be associated with these models, which are presented in Table 2.3 together with the relevant references.

Table 2.3: Learning techniques proposals

References	Learning Approach	
	AL	AI/ML
[193], [194], [202], [206] ⋆		▶
[197], [198], [199] ⋆, [200] ⋆		✓
[201] ⋆		✓
[211] ⋆, [213]	✓	

[AL] Automata Learning (model learning)
[AI] Artificial Intelligence
[ML] Machine Learning

On the one hand, Artificial Intelligence (AI) and Machine Learning (ML) to provide intelligence, even in complex tasks such as decision making and problem solving. However,

UNIVERSIDAD DE MÁLAGA

a considerable amount of data is necessary to train the algorithms and obtain a model capable of representing the system accordingly. There exist different types of algorithm based on the application and objective. In the context of TSN, it is commonly used for traffic scheduling [193], [194], while it is used to identify anomalies [197] and optimize resource allocation [198] in the case of wireless networks (e.g. 5G). In addition, ML is studied in TSN over 5G networks, to assist in tasks such as flow allocation [199], [200]. Furthermore, wireless network configurations are addresses using ML algorithms [201], [202]. Nevertheless, they are only able to maintain awareness of the environment state during the operation, but intermediate states are not represented in the model explicitly. For this reason, it is difficult to explain a reconfiguration decision. Recently, Explainable AI (XAI) aims to make the results of ML understandable to humans by justifying the decisions taken, which can be useful for industry and complex networks. For example, to obtain the prediction cause related to SLA violations [206].

On the other hand, model learning aims to construct a state diagram model from black-box systems. Commonly, Finite-State Machine is adopted to model a system with sequential and discrete behavior, in which input signals are used to transition between states. There are two main approaches followed in model learning. First, the active learning approach that learns through queries and answers, for example to obtain the 4G NAS layer model [211]. Second, the passive learning approach uses examples (i.e. observations) to learn without interaction, applied to derive state machines of protocols such as SMTP [213]. In contrast to AI methods, these models can be formally analyzed with respect to the original system, whose decisions can be mathematically justified. In summary, due to the lack of ability to explain and ensure the accuracy of the decisions made by ML algorithms, this thesis follows the model learning approach. This allows us to describe the evolution of the TSN over 5G network while obtaining logical explanations (understandable by humans) for the decisions taken.

2.5 Analysis and Promising Research Topics

Once the three categories applicable to the objective defined in this thesis have been presented, evaluating the most relevant works and possible challenges, an analysis of the works marked as significant for this thesis follows, which includes the promising research topics. Similarly, Table 2.4 presents the topics in each category of these articles to check the relationship with this work.

First, successful integration of the TSN and 5G technologies requires the implementation of new entities such as TSN AF and TSN translators, where traffic translation procedures are implemented, allowing interoperability between the two domains. One of the main tasks is to ensure that the traffic meets end-to-end quality requirements. This requires QoS mapping (e.g. traffic priorities) and traffic scheduling accordingly [98], [99]. However, this requires that the network components support the procedures described in 3GPP Rel.16 [28] onward and the implementation of these new entities as defined in such standards. Because testbed components such as the RAN are currently compliant with

UNIVERSIDAD DE MÁLAGA

References	Networking Technology			Reconfiguration Technique/Method					Learning Approach	
	TSN	5G	Other	SON	CCL	IBN	ZT	Other	AL	AI/ML
[98], [99]	✓	✓						✓		
[107]		✓	✓							
[108]	✓	✓	✓					✓		
[130]				✓	✓	✓	✓			
[139]			✓	✓		✓	✓			✓
[173]		✓	△					✓		
[174]		✓	△					✓		▶
[199]	✓	✓					✓	✓		✓
[200]	✓	✓						✓		✓
[206]		▶	△	✓				▶		✓
[201]		✓	✓					✓		▶
[211]			✓					✓	✓	

Table 2.4: Proposals combining networking technologies, reconfiguration methods and learning approaches

Rel. 15+, alternative solutions are proposed (see Section 3.2). In addition, to complement the testbed developed as part of this thesis and those presented in [107], [108], a prototype TSN translator has been developed to bring it closer to a real environment such as a factory in industry.

Second, techniques and methods such as SON, CCL, IBN and Zero-Touch described above have key features [130], [139] to achieve the objective proposed in this thesis. Therefore, their combination is essential to enable the necessary functions to achieve automatic reconfiguration [173], [174], which in this case focuses on modifying the parameters of the 5G network to meet the requirements of the TSN traffic.

Third, due to the particular complexity of the combination of these technologies and the very demanding requirements of industrial traffic, these techniques alone are not sufficient. Subsequently, learning techniques are applied to provide the necessary capacity and intelligence to achieve such network reconfiguration with minimal (or even no) human intervention. On the one hand, AI/ML [199], [200] can be applied to complex tasks such as the traffic allocation problem to guarantee low latency. However, the poor explainability of these methods results in it becoming difficult to explain decisions (e.g. reconfigurations) taken. Although the concept of XAI is becoming particularly relevant in trying to explain the causes of decisions, even in scenarios similar to the one in this thesis [206], there are still limitations and challenges that are still relevant. On the other hand, automata learning or model learning [211] allows the creation of models that represent the behavior of the system. These models can be used to learn from the network and make appropriate decisions. For example, apply network reconfiguration to continue meeting traffic requirements over time.

UNIVERSIDAD DE MÁLAGA

Putting it all together, this thesis develops a TSN over 5G testbed, including a prototype of the TSN translators. In addition, the key features of the techniques presented are applied to achieve an automatic reconfiguration of the 5G network to meet the traffic requirements during a TSN traffic session. Furthermore, model learning is used to create a model that represents the behavior of the system, including events such as changes in network configuration and observed KPIs. This model allows us to make predictions of possible deviations of the observed KPIs from the requirements in order to obtain and apply a feasible network configuration that allows us to fulfill the requirements over time without human intervention. Note that this allows the complete history of the network to be stored, allowing for explanations of the decisions taken at any given instant.

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Chapter 3

Toward TSN over 5G Testbed Implementation

Contents

3.1 Testbed Components 50
3.1.1 TSN network components 51
3.1.2 5G network components 51
3.1.3 TSN over 5G network components 54
3.2 Challenges 55
3.2.1 Time synchronization 55
3.2.2 Traffic translation 56
3.3 Relevant 5G Network Parameters 58
3.3.1 5G Core Network parameters 58
3.3.2 5G Radio Network parameters 60
3.4 Key Findings 64

Summary: In this chapter, the implementation of TSN over 5G is discussed at the University of Malaga testbed, including the components of the TSN and the 5G network together with the elements to enable the integration of both domains. In addition, the main challenges and solutions are provided. Finally, the key 5G network parameters to meet traffic requirements are explained.

UNIVERSIDAD DE MÁLAGA

3.1 Testbed Components

This section introduces the testbed developed at the University of Malaga [214] as part of this thesis, where the TSN technology is integrated into the 5G network. All components that make up the testbed are detailed, especially the new components developed for the specific integration of TSN over 5G such as TSN translators, that is, Device-Side Time-Sensitive Networking Translator (DS-TT) and Network-Side Time-Sensitive Networking Translator (NW-TT), and the Time-Sensitive Networking Application Function (TSN AF) called *AutomAdapt*, which is the tool implemented in this thesis work and explained in detail in Section 5.1. In addition, the challenges (e.g. time synchronization) and proposed solutions are explained in depth.

Figure 3.1 shows the TSN over 5G testbed, including the components involved and its connections. First, the elements of a classical TSN network, such as the TSN endpoints, the TSN bridge, the CNC/CUC entities (see Section 1.1.1), and time synchronization equipment (Grand Master clocks). Second, the 5G network components like 5G User Equipment (UE), Radio Access Network (RAN) and 5G Core Network (5GC) (see Section 1.1.2). Note that there are two options for the 5G network; the first is composed of the Nokia RAN and the Open5GS core network; the second is the Keysight UXM 5G Wireless Test Platform. Third, the key elements that enable the integration of both domains, that is, TSN translators and TSN AF (see Section 1.1.3). It should be mentioned that these elements in combination with the 5G network act as a transparent TSN bridge bridge from the perspective of the TSN endpoints, also known as *5GS bridge*.

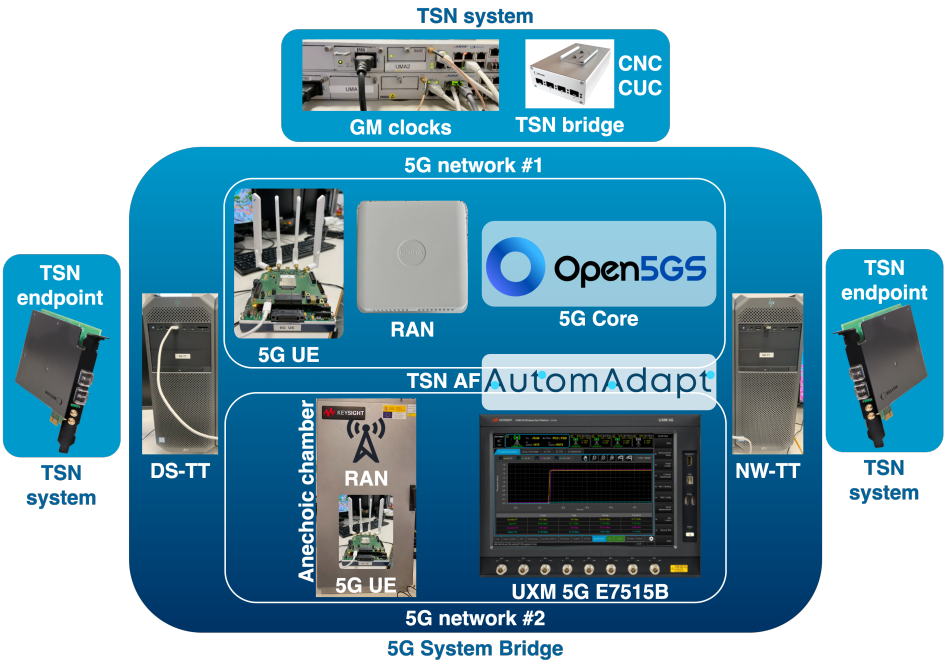


Figure 3.1: TSN over 5G testbed at the University of Malaga

UNIVERSIDAD DE MÁLAGA

3.1.1 TSN network components

The components of a classic TSN network are presented below, which can constitute a wired TSN network commonly used in the industry.

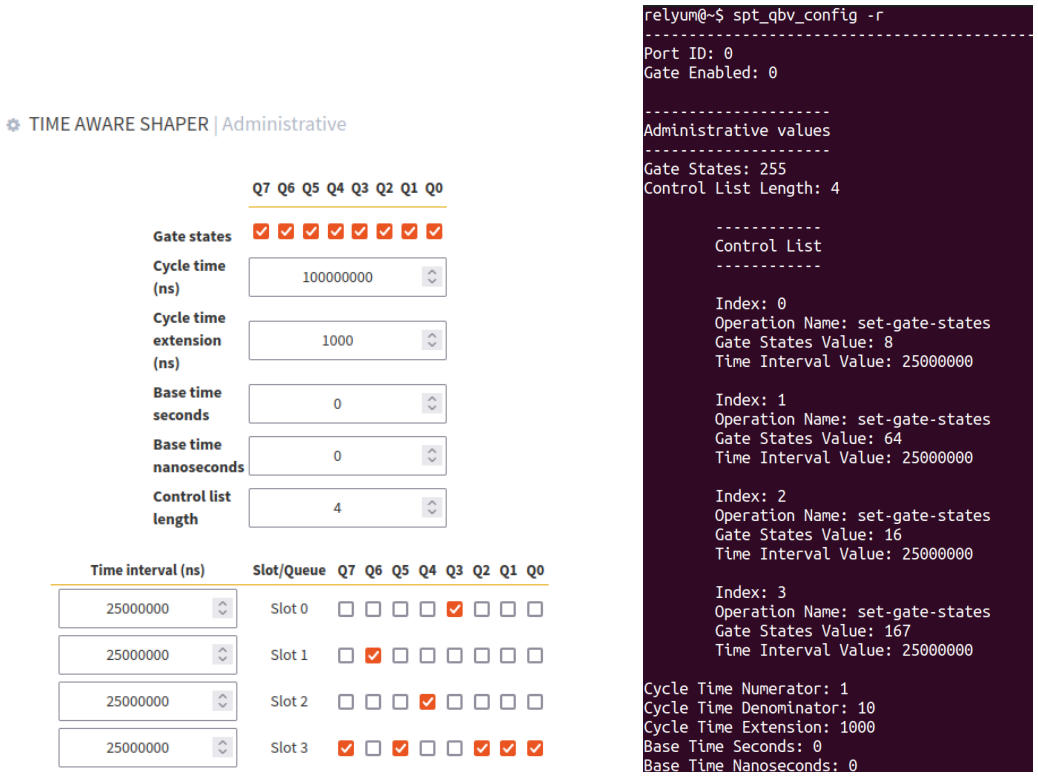
- **TSN endpoints**. There are two *Relyum TSN endpoints* PCIe (Peripheral Component Interconnect express) [215] with two ports that can act as a talker (traffic sender) or listener (traffic receiver). They support TSN features such as timing and synchronization (802.1AS), time-aware shaping (802.1Qbv), credit-based shaping (802.1Qav), and stream reservation protocol (802.1Qcc). In addition, filtering and surveillance (802.1Qci) and FRER (802.1CB) are provided.
- **TSN bridge**. A *Relyum TSN bridge* [216] with four ports that interconnect the TSN endpoints. It also supports the TSN features mentioned above.
- **CNC/CUC**. *Software entities* included in the TSN bridge, receiving requests from TSN endpoints, and applying the required configuration in the TSN network.
- **Time synchronization equipment**. *ADVA FSP 150 EMS* [217] providing a signal clock to be used as Grand Master (GM) clocks. Note that Relyum TSN equipment has hardware clocks that can be used for time synchronization purposes. However, due to the lack of ports available, ADVA synchronization equipment is needed.

Furthermore, Intel i210 [218] network controllers are available in the testbed to be used as additional TSN endpoints, allowing the possibility of creating scenarios with multiple connections simultaneously. The configuration of Relyum TSN equipment can be managed in two ways, including multiple aforementioned TSN features. First, through the graphical user interface (GUI), which enables a visual and easy configuration. For example, Figure 3.2a shows the TAS configuration, which allows the configuration of cycle time, the number of intervals and the selection of the queues allowed to transmit traffic with specific priorities in each interval. Second, using the command-line interface (CLI) as represented in Figure 3.2b, where the TAS configuration is shown (note that the configuration matches the one displayed on the dashboard).

3.1.2 5G network components

The two options available for the 5G network are presented. On the one hand, a commercial network composed of the Nokia RAN and the Open5GS network core. On the other hand, the Keysight 5G network emulator. Both options enable the creation of a 5G network compliant with 3GPP standards, allowing the connection of commercial **5G UE**, which in this case is a *Telit fn980m 5G sub-6GHz and mmWave modem (SA and NSA operations)* [219]. It is compliant with 3GPP Rel.15, includes 4x4 Multiple-Input Multiple-Output (MIMO) in 5G sub-6GHz bands, and supports Frequency Division Duplex (FDD) and Time Division Duplex (TDD). A custom Subscriber Identity Module (SIM) card has been written to connect the 5G modem to the network, including the

UNIVERSIDAD DE MÁLAGA



(a) GUI configuration **(b)** CLI configuration

Figure 3.2: Relyum equipment IEEE 802.1Qbv Time-Aware Shaper configuration

UNIVERSIDAD DE MÁLAGA

secret and operator keys (K and OPc, respectively) necessary for the UE attach request procedure.

Option #1 - Nokia RAN and Open5GS core network

The **RAN** is a *Nokia AirScale Pico RRH 5G* (APHA) for indoor deployment and a *Nokia AirScale Micro RRH 5G* (AWHQE) for outdoor deployment, presented in Figures 3.3 and 3.4, respectively. Both Nokia AirScale [220] are compliant with 3GPP Rel.15+, operating in the n78 band (TDD) with 100MHz channel bandwidth. They support the deployment of 5G NSA and 5G SA and the DL MIMO 4x4 modes.

Figure 3.3: Nokia AirScale Pico RRH 5G

Figure 3.4: Nokia AirScale Micro RRH 5G

The **5GC** is *Open5GS* [221], which is an open source implementation of 5G Core. It is compliant with 3GPP Release-16 and includes features such as network slicing, QoS capabilities, multiple Packet Data Unit (PDU) session, handover, VoLTE (Voice over LTE) and VoNR (Voice over NR), and 5G roaming. The configuration of its functioning is carried out mainly through configuration files (for example, the AMF is defined in amf.yaml). In this case, a common test Public Land Mobile Network (PLMN) is used, composed of the Mobile Country Code (MCC) 999 and the Mobile Network Code (MNC) 99.

Option #2 - Keysight UXM 5G

The E7515B **UXM 5G** Wireless Test Platform [222] is a compact all-in-one unit. The solution integrates the 5GC, RAN and the 5G UE, which is inside an anechoic chamber, allowing performance tests under fully customized channel conditions (even under ideal conditions without interference) coupled with the 5GC. It supports both 5G deployment modes, NSA and SA, and frequency bands FR1 (sub-6 GHz) and FR2 (28 GHz).

UNIVERSIDAD DE MÁLAGA

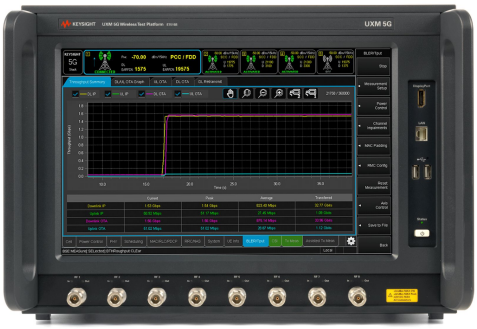


Figure 3.5: Keysight E7515B UXM 5G wireless test solution

3.1.3 TSN over 5G network components

Time-Sensitive Networking Translators

A prototype of the **TSN Translators**, *DS-TT and NW-TT*, has been developed using the programming language P4 [223]. Currently, they are running on a workstation on top of a P4 software switch (BMv2) [224]. They are in charge of some necessary procedures to interconnect the TSN and 5G domains, for example, removing the 802.1Q packet header to be sent across the 5G network and adding it again at the other end of the 5G network. The P4 program specifies how packets are processed by the network pipeline of the switch, including the packet headers definition (e.g. Ethernet, 802.1Q and IPv4), the instructions to parse/deparse packets, and the definition of tables and available actions for the packets. It is based on match-action tables, that is, each table applies specific actions for the packet matched with certain characteristics (for example, headers or fields). The P4 code of the TSN translators and the files necessary for their operation on the testbed described above are available in the AutomAdapt repository [225].

For example, the listing 3.1 defines the table and actions for the management of the 802.1Q (also known as dot1Q) header. Depending on the current domain of the packet, the dot1Q header must be added (TSN network) or removed (5G network) to be processed properly. In addition, the table is in charge of applying the corresponding action by checking each of the packets through the key, which in this case checks the ingress port and the IP destination address (the key could be any of the fields in the package). Note that the P4 program does not specify the specific fields for the control plane. For example, in this case, the ingress port and the destination IP address are not specified, as this would not allow flexibility for real-time adaptation.

```
1    action set_dot1q(bit<3> pri) {
2        if (!hdr.dot1q.isValid()) {
3            hdr.ethernet.etherType = 0x8100;
4            meta.emit_dot1q = 0;
5            meta.pri = pri;
6        }
```


UNIVERSIDAD DE MÁLAGA

```
7     }
8
9     action rm_dot1q() {
10        if (hdr.dot1q.isValid()) {
11            hdr.ethernet.etherType = 0x800;
12            meta.emit_dot1q = 1;
13        }
14    }
15
16    table dot1q {
17      key = {
18          stdmeta.ingress_port:  exact;
19          hdr.ipv4.dstAddr:   ternary;
20      }
21      actions = {
22          set_dot1q;
23          rm_dot1q;
24          none;
25      }
26    }
```

Listing 3.1: Translator P4 code - table and actions for 802.1Q header management

Time-Sensitive Networking Application Function

The **TSN Application Function** is a *software entity* in charge of the control and configuration of the 5GS bridge. It exchanges messages through the CNC to enable the connection between both domains, TSN and 5G. In addition, it stores information related to the TSN translators (e.g. ports) and Packet Data Unit (PDU) sessions (i.e. end-to-end connections) such as time synchronization and QoS mapping. Furthermore, it configures the 5G network based on the traffic pattern declarations of the TSN endpoints. Due to its complexity, its development is one of the main contributions of this thesis. Therefore, it is explained in depth in Chapters 4 and 5.

3.2 Challenges

The integration of TSN and 5G technologies brings with it a number of challenges that need to be resolved. Figure 3.6 represents the two main challenges that have been considered for the construction of the testbed, which are **time synchronization** and **traffic translation** between the TSN and 5G domains. Consequently, these challenges must be resolved in the prototype of the TSN translators (DS-TT and NW-TT) presented above.

3.2.1 Time synchronization

Time synchronization is fundamental in TSN systems to ensure a correct function. There exist protocols and standards that play a key role in time synchronization, such as generalized Precision Time Protocol (gPTP) [75] and IEEE 802.1AS [10]. However, the

UNIVERSIDAD DE MÁLAGA

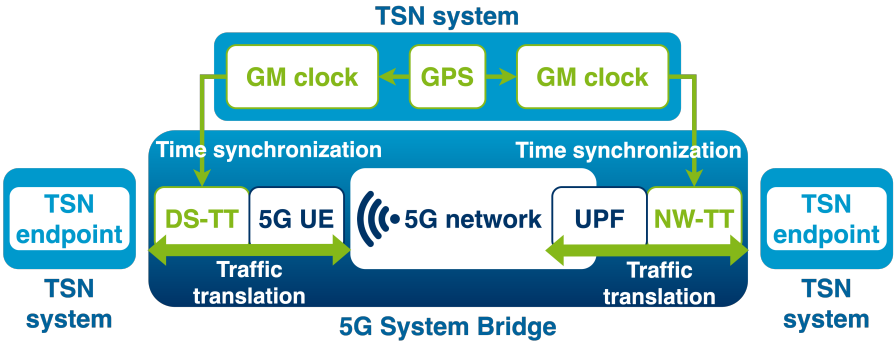


Figure 3.6: Proposed solution for time synchronization and traffic translation

mechanisms and procedures for the interconnection of TSN and the 5G network were introduced in Release 16 [28], where it is defined that the entire end-to-end 5G system is contemplated as time-aware system (IEEE 802.1AS) and only TSN translators located at the edge of the 5G system need to support these procedures. Therefore, TSN translators are responsible for rectifying time synchronization packets based on the residence time between them in the 5G system. For this purpose, the 5G network must be compatible with the transport of PTP packets, for example, using Ethernet PDU sessions.

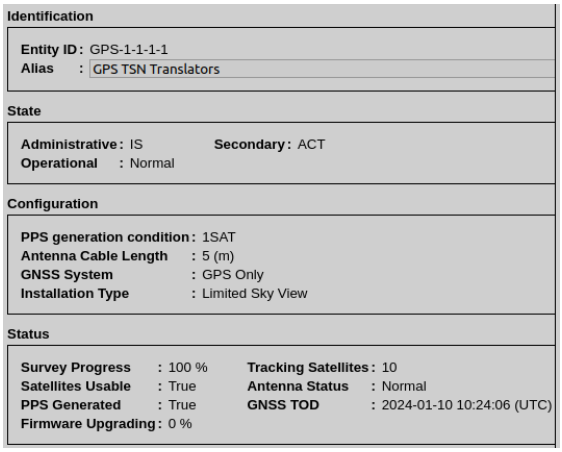
Due to the fact that the available testbed is compliant with Release 15+ and does not support the time synchronization procedures defined in Release 16, an alternative solution has been proposed. Figure 3.6 shows how the system is synchronized; specifically, the TSN translators receive the synchronization packets from two GM clocks that use a single GPS signal. Then the ptp4l [226] and phc2sys [227] programs are used to synchronize the device system clocks. Note that the synchronization difference that may occur between the two GM clocks was measured with an oscilloscope and is considered negligible. Thus, it is assumed that both TSN translators have the same time synchronization signal.

Figures 3.7a and 3.7b display information related to GPS and PTP, respectively. In this case, the GPS uses 10 satellites, and there are two PTP ports configured to create the PTP flows necessary to synchronize the TSN translators. Note that in this case, both are configured as master only, since the PTP packets are not sent through the 5G network, and each TSN translator only receives a single PTP flow.

3.2.2 Traffic translation

Apart from system synchronization, it is vital that both the TSN and the 5G domains can exchange messages and packets. Therefore, TSN translators must be able to perform the necessary procedures to make this happen. In the example mentioned above, in which the 802.1Q headers of the packets must be added or removed, it is important to know specific data of each connection (source and destination address) in order to be able to apply the necessary actions in each case. Since these fields may change for different connections, it is essential to modify the rules defined in the match-action tables. For

UNIVERSIDAD DE MÁLAGA



(a) GPS configuration

EID	Admin State	Alias	PTP FP	Port Identity	Master Only	Port State
PTP PORT-1-1-1	IS	DeviceSideTSNTranslator	ACC PTP FLOW PT-1-1-1-4-1	0080eac42c8000010001	True	Master
PTP PORT-1-1-2	IS	NetworkSideTSNTranslator	ACC PTP FLOW PT-1-1-1-5-1	0080eac42c8000010002	True	Master

(b) PTP ports configuration

Figure 3.7: ADVA FSP 150 EMS configurations

this purpose, P4Runtime [228] allows the addition, deletion, and modification of such rules in real time, allowing the necessary flexibility to apply different actions at each required instant. This also enables the system to be automated by applying procedures without human intervention. Listing 3.2 presents the necessary rules in DS-TT, where the Media Access Control (MAC) and IP destination addresses of the 5G UE are checked to add the dot1q header (since it is removed before in NW-TT to be sent across the 5G network) with the specific priority, which is known for each connection in advance (note that each TSN endpoint sends this information together with the traffic requirements before starting the connection).

```
1  te = sh.TableEntry('ingressImpl.mac_da')(action='ingressImpl.
       set_bd_dmac_intf')
2  te.match['stdmeta.ingress_port'] = ('1')
3  te.match['hdr.ipv4.dstAddr'] =('10.45.0.50')
4  te.action['dmac'] = ('b8:27:eb:13:6f:48')
5  te.action['intf'] = ('2')
6  te.priority = 1
7  te.insert()
8
9  te = sh.TableEntry('ingressImpl.dot1q')(action='ingressImpl.set_dot1q')
10 te.match['stdmeta.ingress_port'] = ('1')
11 te.action['pri'] = ('7')
12 te.priority = 1
13 te.insert()
```

Listing 3.2: Translator P4Runtime code - table rules for 802.1Q header management

UNIVERSIDAD DE MÁLAGA

Consequently, the combination of P4 and P4Runtime enables the definition of mandatory and option procedures specified in standards such as IEEE 802.1Qci [19], specifically, by allowing features such as frame counting, filtering, or service class selection.

3.3 Relevant 5G Network Parameters

The components for implementing TSN over 5G on the testbed, together with their details and challenges, have been presented in this chapter. Due to the fact that the bulk of the work in this thesis focuses on the configuration of specific parameters of these components to meet traffic requirements in the context of a private 5G network in industry, this section presents the relevant 5G network parameters that can be adjusted by the network operator, who has sufficient knowledge of TSN and 5G networks. In particular, the objective is to configure those parameters associated with the Packet Data Unit (PDU) sessions of each 5G UE connected to the network.

3.3.1 5G Core Network parameters

The 3GPP standards [26] define several parameters within the 5G core network that can be configured to improve PDU sessions and optimize Key Performance Indicator (KPI). An end-to-end PDU session, that is, the entire path between the 5G UE and the Data Network (DN) can be composed of one or more QoS flows associated with a Data Radio Bearer (DRB) (between the 5G UE and the 5G RAN) and a Next Generation User Plane tunnel (NG-U) (between the 5G RAN and the 5GC), as depicted in Figure 3.8. Note that an NG-U tunnel is created for each PDU session. In this case, two PDU sessions are created; one to provide internet connection and another PDU session, which contains several QoS flows necessary to provide IP Multimedia Subsystem (IMS) services (e.g. Session Initiation Protocol (SIP), voice, and video). Some of these parameters that are decisive in meeting the requirements of time-critical traffic over the 5G network are as follows.

- **QoS parameters**. 5G Quality of Service Identifier (5QI) in 5G is the equivalent of the QoS Class Identifier (QCI) in 4G networks. It is used to apply individual QoS rules to a QoS flow within a PDU session. In addition, QoS Flow Identifier (QFI) is applied to identify different QoS flows within a PDU session and Allocation and Retention Priority (ARP) determines the priority level assigned to each QoS flow. Accordingly, these QoS parameters are vital to maintaining the same QoS and priority level of the TSN domain within the 5G network. Table 3.1 presents some service examples in addition to their linked values such as 5QI and resource type.
- **Bearer parameters**. A bearer is a communication channel that establishes end-to-end connectivity between the UE and the 5GC. There are two main types of bearers in 5G. First, the default bearer, used as the initial communication that provides IP connectivity for general purposes. Second, the dedicated bearer, which is established on top of the default bearer for specific traffic (for example, real-time

UNIVERSIDAD DE MÁLAGA

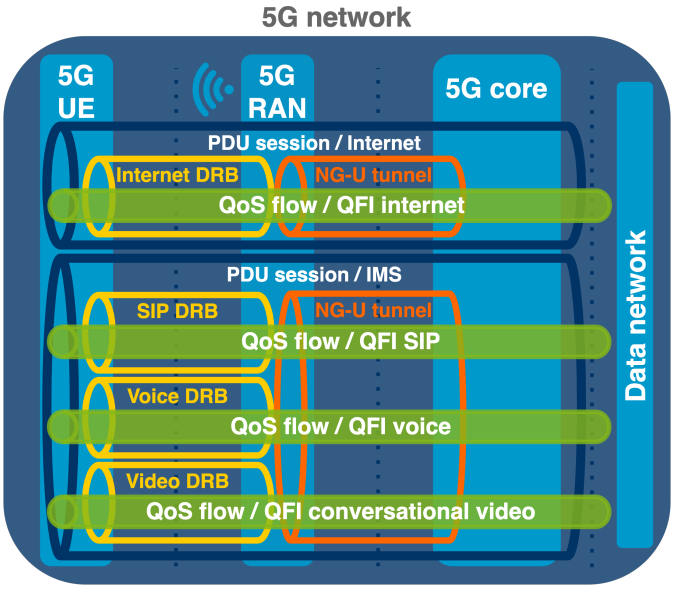


Figure 3.8: 5G Quality of Service Architecture

communication application). Additionally, parameters such as Maximum Bit Rate (MBR) to indicate the highest bitrate that the 5G network can provide to a UE and Guaranteed Bit Rate (GBR) indicating the minimum bitrate guaranteed by the network to a UE.

- **Traffic steering parameters**. The PDU session type requested by the UE (e.g. IPv4, Ethernet, or Unstructured). Note that it is unique for each PDU session and cannot be modified during its lifetime. The UPF plays a key role for the data plane by encapsulating and decapsulating the General Packet Radio Service (GPRS) Tunneling Protocol User Plane (GTP-U) and interconnecting the 5G network and Data Network (DN). Therefore, optimal placement (for example, at the edge of the network close to the UE) and dynamic allocation of the UPF can influence traffic KPIs.
- **Mobility parameters**. Mobility is a critical aspect in mobile networks. A handover allows users to switch connectivity between different base stations. It contributes to the continuity and reliability of the end-to-end connection. Consequently, the handover parameters have an impact on the performance of the network. For example, there are several kinds of handover depending on the scenario, such as intra-cell, inter-cell or soft handover.
- **Timer parameters**. Timer values are vital for UE initial procedures such as *Attach request* (T3410) and *Attach complete* (T3450), and AMF operation to handle paging (T3513) and UE status like deregistered (T3522) or registered (T3555).

In summary, these parameters are crucial for configuring the 5G core network to meet

UNIVERSIDAD DE MÁLAGA

Service Examples	*5QI value*	*Resource type*	*Priority level*	*Delay budget*	*Error rate*
Conversational voice	1	GBR	20	100 ms	10^{-2}
Real-time gaming	3	GBR	40	50 ms	10^{-3}
IMS signalling	5	Non-GBR	10	100 ms	10^{-6}
Video streaming	6	Non-GBR	60	300 ms	10^{-6}
TCP-based (chat, email)	9	Non-GBR	90	300 ms	10^{-6}
Augmented reality	80	Non-GBR	68	10 ms	10^{-6}
Discrete automation	82	Delay-critical GBR	19	10 ms	10^{-4}
Intelligent transport systems	84	Delay-critical GBR	24	30 ms	10^{-5}

Table 3.1: 5G Quality of Service values for common service example [26]

specific service requirements, improve performance, and optimize network resource utilization for end-to-end PDU sessions. Network operators can adjust these parameters based on network architecture and traffic requirements to achieve the expected KPIs.

3.3.2 5G Radio Network parameters

Similarly to the 5G core network, the 3GPP standards [229] define several parameters within the 5G RAN that can be adjusted to improve traffic performance. This section introduces such parameters that can affect the relevant KPIs of time-critical applications. Note that some of the aforementioned parameters are linked with an end-to-end connection and therefore must also be configured in the RAN. For example, the 5QI value is defined in a DRB. Consequently, this section focuses on the different layers of the 5G protocol stack defined in [230] and displayed in Figure 3.9, especially focusing on the user plane, which is the most relevant for the purpose of the thesis work. These layers are connected through communication paths called channels, which are grouped mainly into physical, transport, and logical channels. The main capabilities and procedures of these layers, along with the most significant parameters of each layer, are described below. Note that some parameters (marked as online parameter[1]) allow their modification while the end-to-end traffic session is active, that is, the information is being transmitted through the 5G network. Additionally, several parameters are available for the Downlink (DL) and Uplink (UL) directions[2]. Finally, there are parameters that force a restart of the system[3].

- **Physical layer (PHY)** [231]–[235]. It is the first layer of the protocol stack and is

[1]Online parameter: this parameter can be modified while data is being transmitted through the 5G network

[2]Downlink and Uplink parameter: this parameter can be configured for DL and UL directions

[3]Restart needed parameter: this parameter forces a restart of the system

UNIVERSIDAD DE MÁLAGA

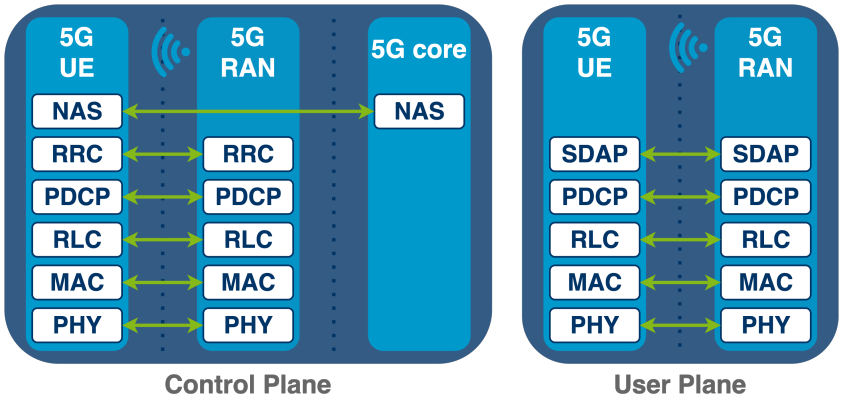


Figure 3.9: 5G Network Protocol Stack - Control Plane and User Plane

responsible for the transmission and reception of raw data over the physical medium (i.e. the air interface). It is involved in the modulation and demodulation of physical channels, the encoding and decoding of the transport channel, radio measurements, multiplexing, and scrambling to ensure reliable wireless communication. Some important parameters are:

  - *MCSTable*[23]: Modulation and Coding Scheme (MCS) table to be applied. For example, 64 Quadrature Amplitude Modulation (QAM) and 256 QAM.
  - *MCS*[2]: MCS index (0 to 28) to be applied to the slots of a subframe. It can only be adjusted manually on the Keysight UXM 5G, on the Nokia RAN it is managed by a proprietary algorithm automatically.
  - *HARQProcesses*[12]: number of Hybrid Automatic Repeat reQuest (HARQ) processes that are active (e.g. 1, 2 and 16).
  - *HARQPolicy*[12]: selection behavior of the HARQ process. For example, *HARQ_PRIO_RETX* gives higher priority to HARQ processes with pending retransmissions.
  - *MaxHARQTransmissions*[12]: maximum number of HARQ transmissions for a HARQ process (e.g. 1,8 and 28).

- **Medium Access Control (MAC)** [234]–[238]. It is responsible for handling shared radio resources, resource allocation, data transmission scheduling, and traffic prioritization. The main parameters are:

  - *delayTarget*[12]: maximum packet delay used by the scheduling algorithm of gNB MAC for GBR bearers only.
  - *prioritisedBitRate*[1]: prioritized bit rate for non-GBR bearers used by the UE for the UL scheduling algorithm.
  - *maxHarqTxDrb*[12]: maximum number of DL/UL HARQ transmissions for an

UNIVERSIDAD DE MÁLAGA

individual DRB profile.

  - *schedulBSD*[1]: Bucket Size Duration (BSD) of the DL/UL scheduler, which is sent to the UE to calculate the bucket size in the logical channel.
  - *schedulPrio*[1]: logical channel priority for the UL scheduler. It should be configured according to the 5QI value to be consistent with the priorities.

- **Radio Link Control (RLC)** [231], [239]. RLC has three modes of functioning: Acknowledged Mode (AM), Unacknowledged Mode (UM), and Transparent Mode (TM). It is responsible for the detection of protocol error (AM) and duplicate (UM and AM), error correction through ARQ (AM), segmentation (UM and AM) and reassembly (AM) of packets, and flow control, providing efficient and reliable data transmission between the MAC and upper layers.
  - RLC Acknowledged Mode (AM) mode. The sender expects acknowledgments from the receiver for each PDU sent. If an acknowledgment is not received within a specified time frame, the sender retransmits the PDU. This mode ensures reliable data transmission, but introduces additional overhead due to acknowledgment messages and potential retransmissions.
    * *maxRetxThreshold*[12]: maximum number of ARQ retransmissions allowed.
    * *pollByte*[12]: interval between polls relative to the number of bytes transmitted.
    * *pollPDU*[12]: interval between polls in PDU.
    * *tPollRetr*[12]: value of the timer to retransmit a poll.
    * *tStatusProhibit*[12]: timer to prohibit the transmission of acknowledgment status reports.
    * *tReassembly*[12]: value of the timer for reassembly.
    * *tReassemblyIBCA*[12]: value of the timer for reassembling used for inter-band CA UE
  - RLC Unacknowledged Mode (UM) mode. In contrast to RLC AM, the sender does not expect acknowledgments from the receiver for each PDU sent. There is no mechanism for retransmission based on acknowledgment. This mode is more lightweight compared to RLC AM but does not guarantee reliable data transmission.
    * *tReassemblyUm*[12]: value of the time for reassembly
    * *rlcUMDrbSNLength*[1]: length of the RLC sequence number to be configured in DRB.
  - RLC Transparent Mode (TM) mode. The RLC header is not added in this

UNIVERSIDAD DE MÁLAGA

mode, and almost no data processing happens (no reassembly, no reordering, no segmentation). Only the transmission side buffers the data, although there is no difference between RLC input and output.

- **Packet Data Convergence Control (PDCP)** [231], [240], [241]. It handles header compression and decompression, ciphering, deciphering, and integrity protection. Additionally, it transfers user and control plane data.
  - *pdcpMaxNumRetxPdu*[1]: maximum number of PDCP PDUs allowed to be retransmitted. Note that this only applies to the DRB that are still pending to be delivered to UE.
  - *tReordering*[12]: timer t-Reordering for DL/UL used to identify loss of PDCP data PDUs in the UE/gNB, respectively.
  - *pdcpBufDiscardEnabled*[1]: enables/disables PDCP buffer-based discarding.
  - *pdcpBufDiscardThreshold*[1]: the quantity of buffered PDCP SDUs and the PDCP PDUs awaiting acknowledgment before discarding new incoming PDCP SDUs.
  - *outOfOrderDelivery*[12]: enables/disables PDCP out-of-order delivery for DL/UL.
  - *pdcpDelayThreshRetxMode*[1]: delay threshold-based retransmission mode for PDCP PDU retransmission, to avoid the expiration of the t-Reordering timer in the UE.
  - *pdcpPduRetxDelayThresh1*[1]: delay threshold 1 for the retransmission of PDCP PDU applied to monitor the delay in the transmission of the PDCP PDU, probably due to the delay or loss of PDCP PDU.
  - *pdcpPduRetxDelayThresh2*[1]: delay threshold 2 for the PDCP PDU retransmission applied to monitor the delay in delivery of the PDCP PDU, probably caused by RLC retransmissions or air link deterioration, contributing to the expiration of the t-Reordering timer in the UE.
- **Service Data Adaptation Protocol (SDAP)** [242]. It deals with QoS flow management by mapping a specific QoS flow within a PDU session to a corresponding DRB (already established with the proper QoS) and also properly marking the QFI of the packets. An SDAP entity exists per PDU session, ensuring that the network meets the service requirements of different applications and traffic types.
- **Radio Resource Control (RRC)** [231]. It manages radio resources (i.e. establishment, release, configuration, and maintenance of point-point radio bearers), mobility functions and handover procedures, and UE measurement and reporting. It plays a crucial role in controlling the radio link and maintaining the overall connection according to the status of the network.
  - *timerRejectWait*[1]: time (s) that the UE waits before sending a consecutive

UNIVERSIDAD DE MÁLAGA

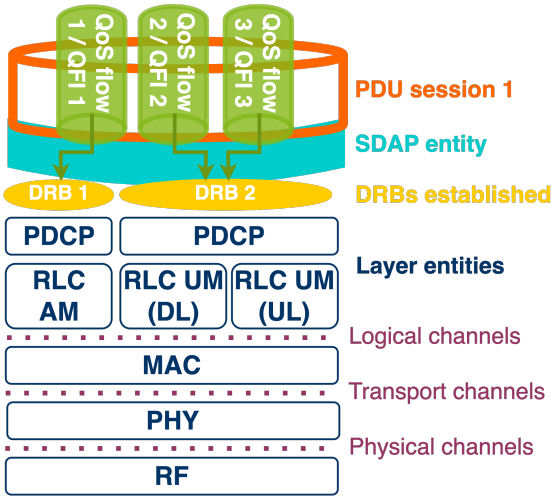


Figure 3.10: QoS Flow mapping within a PDU session on DRBs

RRC setup request message.

- *timerRRCGuard*[1]: time (ms) to wait before aborting the following procedures: RRC message for the setup, modification, or release of a radio bearer sent to the UE.
- *actConnRrcPooling*[3]: activates/deactivates baseband pooling of RRC users connected to the network.

- **Non-Access Stratum (NAS)** [243]. It is responsible for the establishment, maintenance and release of sessions between the UE and 5GC. It also manages the procedures related to authentication, security, and registration. Furthermore, MM (Mobility Management) is implicated in the NAS registration and is mainly in charge of the UE mobility in tasks such as paging, handovers, and location updates. 5G MM defines several states for the network and the UE, which are commonly represented in a state machine.

3.4 Key Findings

In the particular case of the testbed presented above, there are two options for the 5G network that have been used during the thesis. During the first iteration of the methodology, the Keysight UXM 5G (Option #2) was used, as it allows for finer manual operation in a controlled environment such as the anechoic chamber. This also allows for more direct testing of the effect of 5G network parameters on the observed KPIs, where mainly tests were performed using the physical layer parameters such as transmission power, Physical Resource Block (PRB) allocation and MCS index. These parameters have a direct impact on the results obtained. However, their modification in a real environment is not possible or implies a partial or total restart of the system. For this

UNIVERSIDAD DE MÁLAGA

reason, in the second iteration of the methodology, in which the testing was focused on a real environment using the Nokia RAN together with the Open5GS network core (Option #1), it was decided to conduct a more in-depth study on the effect of the different parameters. In conclusion, due to an unmanageable number of possible combinations using the applicable values, it is necessary to focus on a reduced set to study in depth how they affect the most significant KPIs such as delay and jitter in the context of TSN over 5G networks. Specifically, since the objective of this thesis is to configure these parameters automatically while TSN traffic is sent through the 5G network, it is required that the parameter allows for this type of modification (marked as online parameter[1]). Since these kinds of parameter can be configured in most layers [171], the most applicable parameters have to be selected in the case of TSN over 5G, lying in the level of abstraction and control over data transmission.

On the one hand, the physical and MAC layers focus on physical management and access to the environment. Although their respective parameters may influence the results of the KPIs, the physical layer already has algorithms to modify values of these parameters automatically (e.g. MCS index) and the MAC layer has parameters associated with certain types of carriers (e.g. GBR and non-GBR). Therefore, these parameters are not considered to have sufficient potential. On the contrary, the RLC and PDCP layers are at a higher level in the protocol stack, offering a higher level of abstraction. In addition, their parameters can apply more adaptive control over data transmission. The RLC layer is responsible for a number of essential functions to provide efficient and reliable data transmission (e.g. error and duplicate detection, segmentation, reassembly, and flow control), whereas the PDCP plays an important role in data management and data security (e.g. ciphering and deciphering).

Consequently, the characteristics of the Radio Link Control (RLC) layer, make it more suitable to pursue the goal of dynamically reconfiguring the 5G network to meet the requirements of real-time TSN traffic. Due to the relevance of these parameters, in particular the RLC AM mode that allows one to increase the reliability without excessively affecting the latency or jitter, they have been used as the basis for most of the thesis work (see Section 5.2). Furthermore, Table 3.2 briefly presents the impact of such parameters on relevant network KPIs such as end-to-end delay and jitter.

However, the configuration of these parameters must be taken into account through an overall view of the system. For this purpose, this thesis uses model learning as the main approach, which is introduced in Chapter 4, to achieve a global perspective of the network and apply the Zero-Touch approach. In addition, how to dynamically apply configuration changes for the two available 5G network options is detailed in Section 5.1.3.

UNIVERSIDAD DE MÁLAGA

RLC AM parameter	*Impact on delay*	*Impact on jitter*
$MaxRetx_{Threshold}$	Potential increase (more retransmissions)	Potential increase (more variability)
$Poll_{Byte}$	Reduction (less requests)	Potential reduction (even transmissions)
$Poll_{PDU}$	Reduction (less requests)	Potential reduction (even transmissions)
$t_{PollRetr}$	Potential increase (awaiting report)	Potential reduction (predictable transmission)
$t_{StatusProhibit}$	Reduction (less requests)	Potential reduction (even transmissions)
$t_{Reassembly}$	Reduction (quicker discarding)	Potential increase (frequent bursts)

Table 3.2: Impact of RLC parameters on end-to-end delay and jitter

UNIVERSIDAD DE MÁLAGA

Chapter 4

Model Learning Approach

Contents

4.1 Overall Approach 68
4.2 Phase 1. Passive Automata Learning Approach 69
4.2.1 Capturing Network Traces 69
4.2.2 Construction of Learned Automaton 70
4.3 Phase 2. Traffic-Oriented Input/Output Automata 74
4.3.1 Real-time Network Traces 74
4.3.2 Traffic-Oriented Input/Output Automata 76
4.3.3 Construction of Learned Automata 80
4.3.4 Automata for Prediction and Reconfiguration 84

Summary: This chapter presents the model learning (or automata learning) approach to implement automatic network reconfiguration, which consists of two main iterations. First, follow a passive automata learning approach. Then, it provides the notion of Traffic Oriented input/output Automata (TOA), an extension able to enrich the learned information in relation to time.

UNIVERSIDAD DE MÁLAGA

4.1 Overall Approach

As introduced in Section 1.1, the objective of this thesis is to apply automatic reconfigurations according to the requirements imposed by the traffic sent through the TSN over the 5G network. To do this, a 5G network model is built using model learning [183], [184], also called model learning, where the network behavior (i.e. the resulting KPIs) is learned when different traffic profiles are used with a variety of 5G network configuration. Figure 4.1 defines an approach consisting of different modules working in a closed loop.

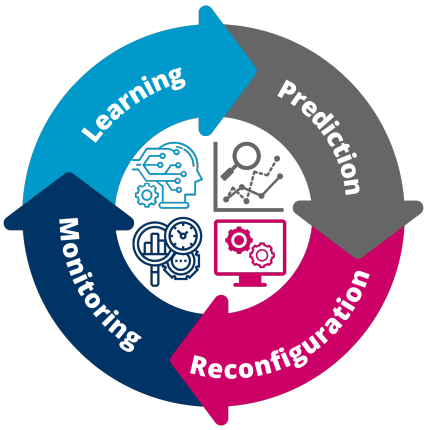


Figure 4.1: Overall approach

The work on this approach has been carried out mainly in the two iterations of the methodology presented in Section 1.2.2. First, the passive automata learning technique is followed (see Section 2.3), where an automaton is inferred from a given data set, namely, from network traffic traces consisting of a sequence of observed states. Second, the passive automata learning method is extended to be time-oriented, which enriches the information of the learning by allowing to represent the evolution of the traffic KPIs in relation to time, together with the information of the network configuration. Furthermore, the automaton can partially interact with the 5G network. A brief description of each module of the approach represented in Figure 4.1 is presented below.

- **Monitoring**. This module monitors the 5G network, including traffic, end-to-end KPIs (e.g. delay and jitter), network configuration (e.g. RAN parameters), and events such as new device connections together with their requirements. The data are processed to extract useful information only from packets corresponding to active connections (for example, packet length).
- **Learning**. This module implements the algorithm to build the automaton, either from network traces captured in advance or directly from the 5G network (real-time). The objective is to build a 5G network model that represents its behavior against

UNIVERSIDAD DE MÁLAGA

different combinations of network configurations and traffic profiles.

- **Prediction**. This module aims to predict possible deviations of KPIs from traffic requirements using the learning automaton. Note that by the nature of the learning used, the system is capable of predicting situations already learned, and it is not possible to predict an unknown situation.
- **Reconfiguration**. This module is responsible for applying a network reconfiguration when necessary. For example, if the prediction module detects a possible KPI deviation in the near future, a new configuration that matches the traffic requirements is applied. Note that, like the prediction module, it is only possible to apply a previously learned network configuration, and it is not possible to generate new network configurations.

4.2 Phase 1. Passive Automata Learning Approach

This phase corresponds to the first iteration of the thesis work, which focuses mainly on monitoring to capture the network traces to be learned and create the automata. However, due to the nature of passive automata learning, i.e. learning from previously collected network traces, the prediction and reconfiguration modules are not significant since there is no active interface to act on the system in real time.

4.2.1 Capturing Network Traces

The first step consists of obtaining traces using the testbed presented in the previous chapter. For this purpose, TSN traffic associated with specific requirements is generated, while the 5G network configuration is changed pseudorandomly several times during a given period of time. Additionally, all relevant events related to traffic and network configuration must be identified in the trace. Therefore, the following events are defined:

- *join*(*intd*)?. The *join* event appears when a TSN endpoint initiates a new connection associated with the requirements (that is, the intended KPIs) given by *intd*.

  The *intended (intd)* KPI is a particular requirement related to traffic sent through the network. Consequently, *intd_throughput* is the minimum value of the amount of data transmitted that the network should guarantee. In the case of *intd_jitter*, it is the maximum value that the traffic can tolerate.

- *config*(*conf*)?. The *config* event occurs when the network changes the configuration to *conf*.
- *leave*?. The *leave* event is applied to close the connection instantiated earlier.

In addition to the events mentioned, the *observed* (*obsd*) KPI is the measured result of the given KPI. Subsequently, *obsd_throughput* and *obsd_jitter*, indicate the volume of traffic sent and the variation between consecutive delays at a given time, respectively. The details of how these events and KPIs are captured are described in Section 5.1.1.

UNIVERSIDAD DE MÁLAGA

4.2.2 Construction of Learned Automaton

This section describes the algorithm for building an automaton from a fixed collection of traces $\mathcal{T} = \{\pi^1, \cdots, \pi^k\}$ observed during k different TSN traffic sessions in the testbed, called $\mathcal{S}$ in the following explanation.

Each trace $\pi = s_0 \cdot s_1 \cdot \cdots \cdot s_{n-1} \in \mathcal{T}$ is a finite sequence of *observed states* s_i (called $\mathcal{N}$-states) over a given period of time. An observed $\mathcal{N}$-state is a tuple of the form $\langle ts, conf, intd, obsd \rangle$, where $ts \in \mathbb{R}$ is the timestamp at which the state was observed, and *conf* is the configuration of the 5G network. The last two components are related to the KPIs, where *intd* represents the intent, that is, the traffic requirements in terms of the KPIs expressed using a real interval denoted by $\mathcal{I}$ between 2 values; and *obsd* symbolizes the observed KPIs, as defined below.

Considering $\mathcal{M}$ as the set of measurements to be monitored that includes all KPIs, for all $m \in \mathcal{M}$, $intd(m) \in \mathcal{I}$ is the interval within m that should remain and $obsd(m) \in \mathbb{R}$ is the observed value at ts. In addition, the set of real intervals $\mathcal{I}$ is of the form (a, b), where $a, b \in \mathbb{R}$ and $a < b$; or (a, ∞); or $(-\infty, b)$; or $(-\infty, \infty)$.

The algorithm Learn, which is the algorithm for building automata, learns automata of the form $\langle Q, q_0, E, f_{\mathcal{A}} \rangle$, where Q is the set of all automaton states, with $q_0 \in Q$ as the initial state; E consists of transitions of the form of (q, ev, q'), where q/q' represent the source and target states of the transition, and ev denotes the internal or external event that triggered the transition.

The function $f_{\mathcal{A}}$ plays a specific role in our definition of automata. It serves the purpose of streamlining the state representation by linking each $q \in Q$ with an *abstraction* $f_{\mathcal{A}}(q)$ of the actual state of the system, which is constructed as described below.

For every $q \in Q$, $f_{\mathcal{A}}(q)$ is a 3-tuple in the format $(conf, intd, obsd^{\alpha})$, with the configuration, the intended KPIs, and the deviation in the observed KPIs, $obsd^{\alpha} : \mathcal{M} \to \mathbb{Z}$, calculated as follows. Given an observed $\mathcal{N}$-state $s = \langle ts, conf, intd, obsd \rangle$ of the system $\mathcal{S}$, a KPI $m \in \mathcal{M}$ and $o = obsd(m)$:

$$obsd^{\alpha}(m) = \begin{cases} -1 & \textit{iff } o < x \ \ \forall x \in intd(m) \\ 0 & \textit{iff } o \in intd(m) \\ 1 & \textit{iff } o > x \ \ \forall x \in intd(m) \end{cases}$$

Note that the value of $obsd^{\alpha}(m)$ establishes a connection between the actual observed values and the intended values in an abstract way. A value of 0 indicates that the observed values align with the intended parameters, while values of 1 and -1 indicate that the observed values exceed or fall short of the intended values, respectively. To define the default configuration $conf_d$ that is employed during system initialization, we use $\bar{0}$ and $\bar{\iota}$ to represent functions that assign a value of 0 and interval $(-\infty, +\infty)$ to each $m \in \mathcal{M}$,

UNIVERSIDAD DE MÁLAGA

respectively. Therefore, we specify $f_{\mathcal{A}}(q_0) = (conf_d, \bar{\iota}, \bar{0})$ as the configuration that allows each $m \in \mathcal{M}$ to take any value.

The algorithm Learn receives a set $\mathcal{T} = \{\pi^1, \cdots, \pi^k\}$ of traces obtained from observation $\mathcal{S}$ as input and produces an automaton $\mathcal{A}_k$ capable of recognizing all behaviors in $\mathcal{T}$. Initially, it generates an initial automaton $\mathcal{A}_0$, which consists of a single initial state q_0 without any transitions. Subsequently, it utilizes the function *Compose* to progressively enhance this automaton by integrating the behavior displayed by each trace $\pi = s_0 \cdot \cdots \cdot s_{n-1} \in \mathcal{T}$. The symbol rd is employed to denote a modification in the automaton state caused by a relevant deviation in the observed values compared to the intended ones.

We use the following symbols to simplify the explanation of the function *Compose*. The variables denoted by q and *conf* correspond to the states and configurations of an automaton, respectively. The procedure involves navigating through the input automaton and introducing new states when needed. Consequently, within the function *Compose*, the variable q_c consistently references the current state of the automaton, and $conf_c$ represents the current configuration in q_c. The function $get(conf, intd, obsd^{\alpha})$ (for example, in line 13) retrieves a state $q \in Q$ if it exists such that $f_{\mathcal{A}}(q) = (conf, intd, obsd^{\alpha})$. Otherwise, it returns $None$. The function $newState(conf, intd, obsd^{\alpha})$ (as seen in line 15) creates a new state q, adds it to Q, and expands $f_{\mathcal{A}}$ to map q to the 3-tuple $(conf, intd, obsd^{\alpha})$. Subsequently, the function $addTran(q, e, q')$ (for instance, in line 16) establishes a new transition from state q to q' labeled by e and adds it to E. Given that E is a set, adding a transition that already exists in E has no impact. Lastly, $deviation(intd, obsd)$ (as shown in line 29) computes the abstract representation of the observed values using the function $obsd^{\alpha}$ defined earlier.

The function *Compose*, as illustrated in the algorithm *Learn*, operates as follows. Initially, it sets the current state q_c to q_0. Then it loops through all trace $\mathcal{N}$-states s_i and expands the current automaton with new states and transistions if it does not capture the behavior exhibited by s_i. The statement *switch* (line 10) evaluates the four potential situations, including external/internal events triggered in s_i. First, lines 11-17 handle the scenario in which a request is made to update the configuration to $conf_n$. Second, lines 18-24 address the situation in which a message is sent to start a new session. Third, the case of a request to terminate the session (lines 25-27) is more straightforward, as it simply resets the current state to the initial state q_0. The final case occurs when no event is triggered in s_i, and therefore a tuple that represents an observed state during a session is processed (lines 28-35). Here, it is assumed that the network configuration aligns with the current one. After traversing all trace states, the updated automaton is returned in line 36.

With the exception of the *leave* event, the algorithm operates uniformly in all other scenarios. It uses the function *get* to verify if the automaton includes a state q' that corresponds to the abstraction of s_i. If such a state is found, a new transition is established from the present state q_c to q'. Alternatively, if no matching state is identified, the

UNIVERSIDAD DE MÁLAGA

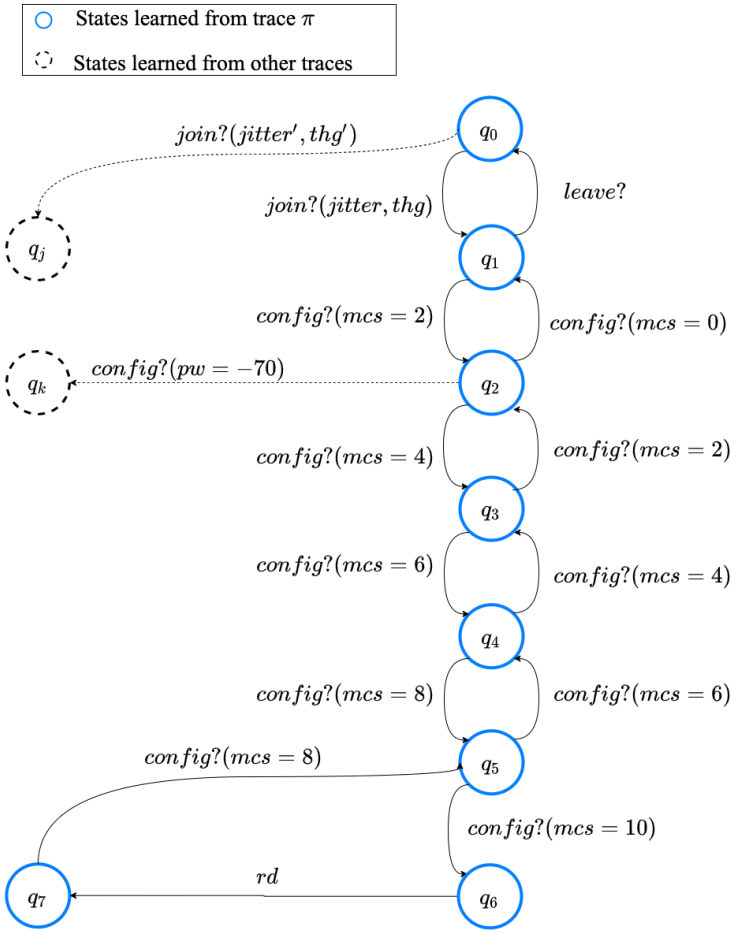


Figure 4.2: Example of passive learned automaton

algorithm generates a new state (also called q') to ensure acceptance of s_i, along with a transition from q_c to q'. In both situations, q' is then designated as the new current state.

The convergence of the Learn algorithm is based on the number of states of the automaton created from the set $\mathcal{T}$, which serves as its input. If we consider $\mathcal{T}$ to be finite and each trace in $\mathcal{T}$ to have a finite number of states, then the termination of the algorithm Learn is straightforward. However, when we assume that the automaton is continuously updated with new traces to capture new behaviors, achieving convergence is not as straightforward. The size of the constructed automaton is heavily influenced by the various configurations applicable to the system $\mathcal{S}$ and the potential intervals that define the target values for each monitored metric. It is important to note that the automaton abstracts the observed values, hence they do not impact the algorithm termination. In addition, it can be assumed that the number of realistic configuration values (referring to [26]) and intent intervals established for new sessions is limited. Consequently, the automaton generated by the Learn algorithm is finite.

UNIVERSIDAD DE MÁLAGA

Algorithm Learn: Construction of the automaton $\mathcal{A}_k$ accepting a set of traces $\mathcal{T}$

```
1  Algorithm Learn(↓ T = {π^1, ··· , π^k}, ↑ A_k):
2      A_0 := ⟨{q_0}, q_0, ∅, {q_0 → (conf_d, ī, 0̄)}⟩;
3      for i := 1 ... m do
4          A_i := Compose(A_{i−1}, π^i) ;
5      return A_k;
6  Function Compose(↕ A, ↓ π = s_0 · ··· · s_{n−1}):
7      q_c := q_0;
8      for i := 0 ... n − 1 do
9          (conf_c, it, ob^α) := f_A(q_c);
10         switch s_i do
11             case config(conf_n)? do
12                 if conf_n ≠ conf_c then
13                     q' := A.get(conf_n, it, 0̄);
14                     if q' = None then
15                         q' := A.newState(conf_n, it, 0̄);
16                     A.addTran(q_c, config(conf_n)?, q');
17                     q_c := q';
18             case join(it')? do
19                 q' := A.get(conf_c, it', 0̄);
20                 if q' = None then
21                     q' = A.newState(conf_c, it', 0̄);
22                 if q' ≠ q_c then
23                     A.addTran(q_c, join(it')?, q');
24                     q_c = q';
25             case leave? do
26                 A.addTran(q_c, leave?, q_0);
27                 q_c := q_0;
28             case ⟨ts, conf_c, it', ob'⟩ do
29                 o^α = deviation(it', ob');
30                 q' := A.get(conf_c, it', o^α);
31                 if q' = None then
32                     q' := A.newState(conf_c, it', o^α);
33                 if  q' ≠ q_c then
34                     A.addTran(q_c, rd, q');
35                     q_c := q';
36     return A;
```


UNIVERSIDAD DE MÁLAGA

In terms of implementation performance, the monitoring and learning modules (see Section 5.1 should not exhibit any delay between them, even running these components for long periods on a real network. In conclusion, the learning module must handle each network capture efficiently, and the complexity (i.e. size) of the learned automaton is limited mainly by the number of network configurations. Note that although this first phase is oriented to passive automata learning and the algorithm does not allow us to represent the evolution of the network, the design of the algorithm is not only limited to build the automaton from previously captured traces, but also allows us to do it directly on the network.

4.3 Phase 2. Traffic-Oriented Input/Output Automata

This second phase is a continuation of the work done in the previous phase. The most substantial changes are as follows: First, we define a new type of automaton, called Traffic Oriented input/output Automata (TOA), which represents the evolution of the TSN session in relation to time. Second, we improve the previously defined learning algorithm. Third, the new automaton (TOA) makes it possible the development of an algorithm for prediction and reconfiguration. All these changes are presented in detail below. In addition, the approach of this new phase that allows representing the network trace information with respect to time is consistent with the possibility to interact with the system in real time, e.g. by applying network reconfigurations when predictions of possible deviations from expected KPIs are made. Therefore, the four modules are operated and vital in this phase to enable the system to autonomously implement the necessary adjustments in the 5G QoS flow setup. In this way, a closed-loop mechanism is defined consisting of several steps using the four modules shown in Figure 4.1.

The entire monitoring, prediction, and reconfiguration procedure must be performed simultaneously with the live TSN session, without adversely affecting the KPIs. The learning process to build the TOA can be approached in different ways. An approach involves conducting multiple trials to generate real session traces to construct the TOA, which can then be used in the closed-loop process. Alternatively, another approach is to create or expand the TOA while operating within the loop.

4.3.1 Real-time Network Traces

Similarly to the previous phase, the first stage of the second phase involves the capture of data that contain diverse TSN traffic (with different requirements) while the 5G network configuration is changed. Although TOA can be constructed from previously captured network traces, the potential of the new time-oriented approach could not be exploited. Therefore, in this phase, all events (e.g. configuration change request and change of the observed KPIs quality) together with traffic measurements are obtained mainly by directly interacting with the system. Consequently, the monitoring module can capture snapshots of the system that include the 5G network configuration and real-time KPIs as *concrete network states* (called $\mathcal{N}$-states). These states correspond to the network

UNIVERSIDAD DE MÁLAGA

traces utilized as input to the learning module to generate an automaton that represents the behavior of the network in an abstract manner. The specific network states must be transformed into the so-called *abstract states* in order to be included in the TOA. Abstracting concrete states is crucial to creating finite and manageable TOA that can be analyzed during the prediction phase. Therefore, the approach to abstract concrete states needs to be considered when constructing the TOA.

On the one hand, since the number of distinct 5G configurations specified by 3GPP standards is limited, they can be directly incorporated into the abstract states. However, the actual KPI values can make the TOA unmanageable without applying some kind of abstraction to them. The function *qty*, which is explained in detail in the following, serves to condense the ranges of values of all monitored KPIs into individual ones for an efficient comparison with the expected ranges of KPIs during the TSN session. In a realistic scenario, this function is defined by the network operator by assigning quality rankings to the observed KPIs based on their criteria (e.g. define a larger number of ranges to obtain finer grain information). For example, Table 4.1 shows a particular definition of *qty*, where several ranges and the correspondence with the observed KPIs are exposed. In this case, when the *qty* value is positive, it means that the operator has established it as a valid threshold for a specific traffic pattern (see Table 1.2). In contrast, negative values indicate the presence of at least one invalid threshold for the observed KPIs in this traffic pattern. Note that a *qty* function is characterized for each traffic pattern presented in the network.

qty	*delay* (ms)	*jitter* (ms)
-18	[15-20)	[15-20)
-8	[15-20)	[0-5)
-1	[0-5)	[15-20)
1	[0-5)	[0-1)
2	[0-5)	[1-5)
12	[5-15)	[0-1)
13	[5-15)	[1-15)

Table 4.1: Example of *qty* function values for a given traffic pattern

The following sections provide a formal description of the entire process of learning TOA from network traces together with the prediction module, which uses the learned TOA to predict potential deviations in the quality of the KPIs and recommends configuration adjustments when feasible. In cases where no configuration changes can be proposed based on existing data, the network operator will receive an alert to address the issues manually. Note that the impact of such manual changes on network configuration and its outcomes in terms of KPIs can be included in the TOA for future decision making.

UNIVERSIDAD DE MÁLAGA

4.3.2 Traffic-Oriented Input/Output Automata

In this section, the concept of Traffic Oriented input/output Automata (TOA) is presented as the particular formalism that describes the behavior of the network $\mathcal{N}$ that needs to be learned.

Let $\mathcal{C}$ represent the collection of all potential network configurations, with $conf_d \in \mathcal{C}$ denoting the initial configuration by default. It is assumed that $KPI = \{kpi_1, \cdots, kpi_k\}$ constitutes the set of k network KPIs that will be monitored, and these are represented as vectors $\bar{o} = \langle o_1, \cdots, o_k \rangle \in \mathbb{R}^k$ where each o_i corresponds to the measurement of the KPI kpi_i. The symbol $\perp$ is utilized to denote unknown KPI values.

Definition 1. *Network states* (referred to as $\mathcal{N}$-states) are characterized as pairs of the form of $(conf, \bar{o}) \in \mathcal{C} \times (\mathbb{R}^k \cup \{\perp\})$, where *conf* represents a network configuration and $\bar{o}$ denotes a vector containing the values of KPIs (or $\perp$ if the values are unknown). A *network trace* (or $\mathcal{N}$-trace) is described as a finite sequence of $\mathcal{N}$-states, such as $(conf_0, \bar{o}_0) \cdots (conf_n, \bar{o}_n)$.

During a network session, the network states *observed* are coupled by a supplementary timestamp that indicates the exact time at which the state was recorded.

Definition 2. An *observed network state* is described as 3-tuples $\langle ts, conf, \bar{o} \rangle$, where $ts \in \mathbb{R}^{\geq 0}$ represents the timestamp and $(conf, \bar{o}) \in \mathcal{C} \times (\mathbb{R}^k \cup \{\perp\})$ denotes an $\mathcal{N}$-state. An *observed network trace* (or observed $\mathcal{N}$-trace) $\langle ts_0, conf_0, \bar{o}_0 \rangle \cdots \langle ts_n, conf_n, \bar{o}_n \rangle$ is a finite sequence of observed $\mathcal{N}$-states ensuring that $\forall 0 \leq i < n.\ ts_i < ts_{i+1}$.

As previously mentioned, a quality function for KPIs is utilized to abstract the $\mathcal{N}$-states, enabling them to be stored in a *finite* manner by TOA.

Definition 3. A function $qty : \mathbb{R}^k \cup \{\perp\} \to \mathbb{Z}$ is considered a *valid quality function* if the range of qty ($img(qty) \subseteq \mathbb{Z}$) is finite and $qty(\perp) = 0$.

Definition 4. An *abstract $\mathcal{N}$-state* is defined as a pair $(conf, v) \in \mathcal{C} \times \mathbb{Z}$. Considering a valid quality function qty and a $\mathcal{N}$-state $(conf, \bar{o}) \in \mathcal{C} \times (\mathbb{R}^k \times \{\perp\})$, the abstraction of $(conf, \bar{o})$ is described adopting qty as $qty(conf, \bar{o}) = (conf, qty(\bar{o}))$.

TOA, as defined in the following, is constructed using abstract $\mathcal{N}$-states. In the sequel, qty represents the quality function that establishes a connection between the $\mathcal{N}$-states and the abstract $\mathcal{N}$-states.

Definition 5. A Traffic Oriented input/output Automata (TOA) is defined as $\mathcal{A} = \langle Q, q_0, E, f_{\mathcal{A}} \rangle$ where Q is a finite set of states of the automaton, with $q_0 \in Q$ being the initial state. The function $f_{\mathcal{A}}$ is *injective* and maps each automaton state $q \in Q$ to an abstract $\mathcal{N}$-state $(conf, v) \in \mathcal{C} \times \mathbb{Z}$, where $f_{\mathcal{A}}(q_0) = (conf_d, 0)$. The set E contains the edges of the TOA of the form $(q, q') \mapsto label$ with $q \neq q'$. Here, q and q' are the source and target states of the edge, respectively, and *label* can be one of two actions:

- **cf**(*conf*) with $conf \in \mathcal{C}$, representing a transition caused by a request for a change in the network configuration;

UNIVERSIDAD DE MÁLAGA

- $\mathbf{rd}(n)$ with $n \in \mathbb{N}^+$, representing a transition triggered by a change in the quality of observed KPIs, where n serves as a guard to determine the time instant at which the transition is fired.

Definition 6. For a given $\mathcal{A} = \langle Q, q_0, E, f_\mathcal{A} \rangle$, the concretization function $\gamma : Q \to \mathcal{C} \times (\mathbb{R}^k \cup \{\bot\})$ is specified as $\gamma(q) = \{(conf, \bar{o}) | f_\mathcal{A}(q) = qty(conf, \bar{o})\}$.

Intuitively, each state $q \in Q$ of the TOA serves as a label for the abstract state $f_\mathcal{A}(q) = (conf, v)$ designed to represent the set of $\mathcal{N}$-states $\gamma(q)$ in the automaton in a finite manner. Furthermore, a transition in the TOA $(q, q') \mapsto label$ indicates a network evolution resulting from a change in configuration (labeled $\mathbf{cf}(conf)$) to *conf*, or a change in the quality of observed KPIs (labeled $\mathbf{rd}(n)$) when more than n time units have elapsed. Each state $q \in Q$ could have multiple outgoing **cf**-transitions, as the network configuration can always be replaced by a different one. However, under the assumption of deterministic network behavior, there can be at most one potential change in the quality of observed KPIs due to the passing of time, as specified in the following condition.

Deviation Condition. Every TOA $\mathcal{A} = \langle Q, q_0, E, f_\mathcal{A} \rangle$ is required to ensure that for each state $q \in Q$, there can be *at most* one outgoing edge of the type $(q, -) \mapsto \mathbf{rd}(-)$ in E.

Example 1. Figure 4.3 illustrates a TOA sample that shows the performance of a basic network with the RLC AM profile p_{AM} and delay δ as the only adjustable parameter and the observed KPI, respectively. The parameter p_{AM} is presumed to be $p1$ (the initial default value) or $p2$. The delay δ (measured in milliseconds) is represented in an abstract manner using the function $qty : \mathbb{R} \cup \{\bot\} \to \mathbb{Z}$ with the following definition: $qty(\bot) = 0$ and

$$qty(\delta) = \begin{cases} 1 & \textit{iff } \delta \in (0, 5] \\ 2 & \textit{iff } \delta \in (5, 10] \\ -1 & \textit{otherwise} \end{cases}$$

In Figure 4.3, each state q of the automaton also displays the value of $f_\mathcal{A}(q)$, where the first part represents the configured RLC AM profile and the second part the abstraction of the delay. It is important to note that in the initial state q_0, the RLC profile is set to $p1$ while the delay remains unknown. Transitions marked with $\mathbf{rd}(-)$ (for example, $(q_0, q_1) \mapsto \mathbf{rd}(4)$, $(q_1, q_2) \mapsto \mathbf{rd}(45)$, $(q_3, q_4) \mapsto \mathbf{rd}(48)$, and $(q_5, q_3) \mapsto \mathbf{rd}(8)$) indicate a change in the observed delay quality after time intervals of 4, 45, 48, and 8 units, respectively. Furthermore, transitions labeled with $\mathbf{cf}(-)$ (such as $(q_1, q_3) \mapsto \mathbf{cf}(p_2), (q_3, q_1) \mapsto \mathbf{cf}(p_1)$, and $(q_2, q_5) \mapsto \mathbf{cf}(p_2)$) are triggered by modifications in the RLC AM profile.

The objective of the remaining part of this section is to establish a formal connection between each TOA and the specific subset of $\mathcal{N}$-traces it denotes. To achieve this, Definition 8 introduces the trace-based semantics of TOA, which is based on a transition system that generates a collection of $\mathcal{N}$-traces. This collection is formed by using the

UNIVERSIDAD DE MÁLAGA

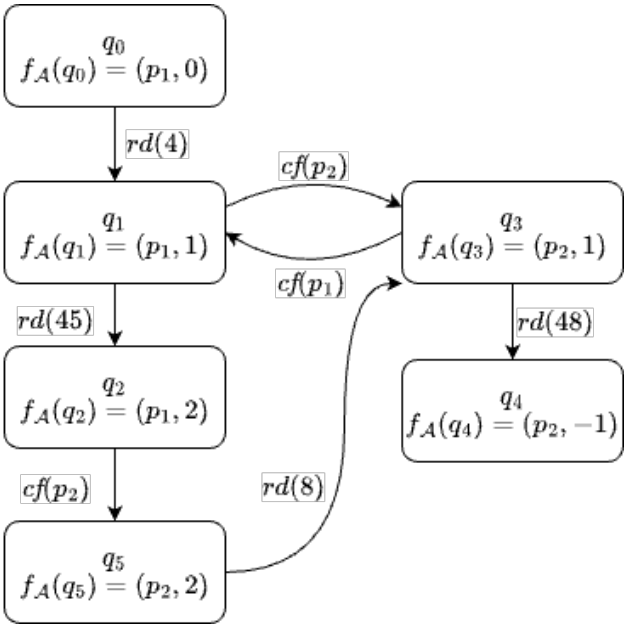


Figure 4.3: Example of learned TOA

function γ, as defined in Definition 6, which links each TOA state to the set of $\mathcal{N}$-states it represents. Likewise, the subsequent definition outlines how TOA can also abstract successive $\mathcal{N}$-states of the real network.

Definition 7. Let $\mathcal{A} = \langle Q, q_0, E, f_A \rangle$ represent a TOA. The semantics of $\mathcal{A}$ is established through the transition system $(S, s_0, \overline{\rightarrow})$, where $S \subset Q \times \mathbb{R}^{\geq 0} \times \mathbb{R}^{\geq 0} \times \mathcal{C} \times (\mathbb{R}^k \cup \{\perp\})$. Each state $s = (q, c, ts, conf, \bar{o}) \in S$ includes a TOA state $q \in Q$, a clock value $c \in \mathbb{R}^{\geq 0}$, and an observed $\mathcal{N}$-state $\langle ts, conf, \bar{o} \rangle$. The initial state is $s_0 = (q_0, 0, 0, conf_d, \perp) \in S$. For every state $(q, c, ts, conf, \bar{o}) \in S$, it is required that $(conf, \bar{o}) \in \gamma(q)$.

The transition relation $\overline{\rightarrow} \subset S \times \{\mathbf{cf}, \mathbf{rd}, \mathbf{dl}\} \times S$ is established by rules (1), (2), and (3) described below. These rules define the behavior of the system, where variable ts stores the global timestamp, and the clock value c_j of the target state $s_j = (q_j, c_j, ts_j, conf_j, \bar{o}_j)$ is calculated to record the time elapsed in the state q_j.

1. *Transition generated by a TOA edge that is marked by a variation in the quality of the KPI observed.* This transition occurs for $s_i = (q_i, c_i, ts_i, conf_i, \bar{o}_i)$ if there is a TOA edge of the form of $(q_i, -) \mapsto \mathbf{rd}(n)$ and if more than n time units have elapsed at state q_i.

   $(q_i, c_i, ts_i, conf, \bar{o}_i) \xrightarrow{\mathbf{rd}} (q_j, 0, ts_j, conf, \bar{o}_j)$,

   if $\exists n > 0$

   (a) $(q_i, q_j) \mapsto \mathbf{rd}(n) \in E$ and

UNIVERSIDAD DE MÁLAGA

(b) $c_i + (ts_j - ts_i) \geq n$

2. *Transition generated by a TOA edge labeled with a change in configuration.* This transition occurs for $s_i = (q_i, c_i, ts_i, conf_i, \bar{o}_i)$ if there is a TOA edge in the format $(q_i, -) \mapsto \mathbf{cf}(-)$ and rule (1) is not applicable:

   $(q_i, c_i, ts_i, conf_i, \bar{o}_i) \xrightarrow{\mathbf{cf}} (q_j, 0, ts_j, conf_j, \bar{o}_j)$, if

   (a) $(q_i, q_j) \mapsto \mathbf{cf}(conf_j) \in E$ and

   (b) if there exist $q_k \in Q$ and $n \in \mathbb{N}$ such that $(q_i, q_k) \mapsto \mathbf{rd}(n) \in E$, then $c_i + (ts_j - ts_i) < n$

3. *Transition due to time elapse.* It can be triggered when rule (1) is not applicable:

   $(q, c_i, ts_i, conf, \bar{o}_i) \xrightarrow{\mathbf{dl}} (q, c_j, ts_j, conf, \bar{o}_j)$ if

   (a) $c_j = c_i + (ts_j - ts_i)$ and

   (b) if there exist $q_k \in Q$ and $n \in \mathbb{N}$ such that $(q, q_k) \mapsto \mathbf{rd}(n) \in E$, then $c_j < n$

The condition of **deviation** mentioned earlier ensures that there is no ambiguity when implementing rule (1), as each state q_i can have only one **rd**-transition going out. It is important to note that the **rd**-transition specified in rule (1) must be followed if conditions (1.a) and (1.b) are satisfied. Otherwise, rules (2) and (3) may be applied in a non-deterministic way, if feasible.

Example 2. The TOA transition system is clarified by showing two sequences of states generated by the automaton depicted in Figure 4.3:

$$\begin{aligned} seq_1 = &(q_0, 0, 0, p_1, \perp) \xrightarrow{\mathbf{rd}} (q_1, 0, 4, p_1, 2ms) \xrightarrow{\mathbf{dl}} \\ &(q_1, 4, 8, p_1, 1ms) \xrightarrow{\mathbf{dl}} (q_1, 8, 12, p_1, 4ms) \xrightarrow{\mathbf{rd}} \\ &(q_2, 0, 45, p_1, 7ms) \cdots \end{aligned}$$

The transition from $(q_0, 0, 0, p_1, \perp)$ to $(q_1, 0, 4, p_1, 2ms)$ occurs due to rule (1) after a duration of 4 time units. The two successive **dl**-transitions in seq_1 is triggered by rule (3) as long as the time elapsed is less than 45 time units. The final transition is generated by rule (1) and results in a modification of the quality of the monitored KPIs. It is important to note that the sequence has the potential to extend endlessly through the application of further **dl**-transitions.

UNIVERSIDAD DE MÁLAGA

$$
\begin{aligned}
seq_2 = &(q_0, 0, 0, p_1, \perp) \xrightarrow{\mathbf{rd}} (q_1, 0, 4, p_1, 2ms) \xrightarrow{\mathbf{cf}} \\
&(q_3, 0, 8, p_2, 2ms) \xrightarrow{\mathbf{dl}} (q_3, 4, 12, p_2, 4ms) \xrightarrow{\mathbf{cf}} \\
&(q_1, 0, 16, p_1, 3ms) \cdots
\end{aligned}
$$

The sequence seq_2 starts as seq_1 and then continues a **cf**-transition occurs due to rule (2) to change the network configuration to p2. After a **dl**-transition, a **cf**-transition is retriggered to change the network configuration to p1. The sequence may proceed by alternating the network configuration (between p1 and p2) as long as the time elapsed in the respective states is less than 45 or 48 ms.

For a state $s = (q, c, ts, conf, \bar{o}) \in S$ of the transition system $(S, s_0, \bar{\rightarrow})$, the $\mathcal{N}$-state $s_\downarrow$ is created by extracting the last two components of s, resulting in $s_\downarrow = (conf, \bar{o})$.

Definition 8. The trace-based semantics of the transition system $(S, s_0, \bar{\rightarrow})$ is denoted by $\mathcal{O}(S, s_0, \bar{\rightarrow})$ and defined as $\{s_{0\downarrow} \cdot s_{1\downarrow} \cdots s_{n\downarrow} \mid \forall\ 0 \leq i < n.\ s_i \bar{\rightarrow} s_{i+1}\}$.

The set $\mathcal{O}(S, s_0, \bar{\rightarrow})$ encompasses the collection of traces of $\mathcal{N}$ (as defined in Definition 1) that are generated by executing the transition relation $\bar{\rightarrow}$ starting from the initial state s_0 and then extracting the last two components of the resulting states. Observe that $\pi_\downarrow = s_{0\downarrow}$ is always in $\mathcal{O}(S, s_0, \bar{\rightarrow})$.

Definition 9. Assuming that $(S, s_0, \bar{\rightarrow})$ is the transition relation given by $\mathcal{A}$, we say that (1) A TOA $\mathcal{A}$ accepts a $\mathcal{N}$-trace $\pi_\downarrow = s_{0\downarrow} \cdot s_{1\downarrow} \cdots s_{n\downarrow}$ iff $\pi_\downarrow \in \mathcal{O}(S, s_0, \bar{\rightarrow})$; (2) A TOA $\mathcal{A}$ accepts a collection of $\mathcal{N}$-traces $\mathbf{T}$ iff $\mathbf{T} \subseteq \mathcal{O}(S, s_0, \bar{\rightarrow})$.

Example 3. We now show the sequence of projected states extracted from seq_1 and seq_2 of Example 2 to clarify the process of constructing the elements of $\mathcal{O}(S, s_0, \bar{\rightarrow})$.

$seq_{1\downarrow} = (p1, \perp) \cdot (p1, 2ms) \cdot (p1, 1ms) \cdot (p1, 4ms) \cdot (p1, 7ms)$

$seq_{2\downarrow} = (p1, \perp) \cdot (p1, 2ms) \cdot (p2, 2ms) \cdot (p2, 4ms) \cdot (p1, 3ms)$

4.3.3 Construction of Learned Automata

The algorithm *Learn* is now presented as a technique for generating a Traffic Oriented input/output Automata (TOA) from a finite set of *observed network traces* $\mathcal{T} = \{\pi^1, \cdots, \pi^p\}$ obtained from p distinct TSN sessions within a specific 5G network. Each individual trace $\pi^i = z_0^i \cdot z_1^i \cdot \cdots \cdot z_{n-1}^i \in \mathcal{T}$ is a finite sequence of *observed $\mathcal{N}$-states* z_j^i (see Definition 2) that are generated either upon a configuration change event or when a predefined time threshold is reached. The algorithm *Learn* produces a TOA $\mathcal{A}_p$ that can accept all behaviors in $\mathcal{T}$ following Definition 9. In order to construct this automaton, the algorithm utilizes a valid quality function qty as specified in Definition 3.

UNIVERSIDAD DE MÁLAGA

The algorithm *Learn* operates as follows. Initially, it generates an initial TOA $\mathcal{A}_0$, containing only an initial state q_0 without transitions. It also establishes the dictionary lv to link each automaton state with the last visited time instant. Initially, the dictionary only comprises the tuple $q_0 \mapsto 0$ indicating that the state q_0 is reached at time 0. Subsequently, it invokes the function *Compose* to progressively extend the TOA by integrating the behavior exhibited by each network trace from $\mathcal{T}$.

The algorithm operates by traversing the input automaton and adding new states to it as needed. Therefore, within the function *Compose*, variable q_c consistently refers to the current state of the automaton, and $conf_c$ represents the current configuration in q_c. The algorithm uses the following auxiliary functions to interact (consult and update) with the automaton in construction $\mathcal{A}$ or to update lv. The function $qty(\bar{o})$ (e.g. line 13) computes the abstract representation of the observed KPIs (see Definition 3). The function $get(conf, qty(\bar{o}))$ (e.g. line 13) retrieves a state $q \in Q$ such that $f_{\mathcal{A}}(q) = (conf, qty(\bar{o}))$, if it exists; otherwise, it returns *None*. The function $newState(conf, qty(\bar{o}))$ (e.g. line 15) generates a new state q, adds it to Q, and updates $f_{\mathcal{A}}$ to link q to the pair $(conf, qty(\bar{o}))$. The function $addTran(q, lb, q')$ (e.g. line 17) introduces a new transition from state q to q' labeled with lb. If the key (q, q') is not present in E, $addTrans$ simply includes the pair $(q, q') \mapsto lb$ in E. Otherwise, E remains unchanged. This is the case if the label indicates a configuration change such as $\mathbf{cf}(conf)$. However, if the label is of the form $\mathbf{rd}(t)$ and $(q, q') \mapsto \mathbf{rd}(t')$ is the current pair in E, it is replaced by $(q, q') \mapsto \mathbf{rd}(min(t, t'))$, where $min(t, t')$ is the minimum value between t and t'. The function $update(q, ts)$ (e.g. line 18) updates lv so that the pair $q \mapsto ts$ is present in the dictionary. Finally, the function $getValue(q)$ (e.g. line 21) retrieves the timestamp recorded in lv for state q, if q is in the dictionary or returns *None*, otherwise.

The function *Compose*, used in algorithm *Learn*, operates as follows. Initially, it sets the current state q_c to q_0. Then, it goes through all observed $\mathcal{N}$-states $z_i = \langle ts, conf, \bar{o} \rangle$ and modifies the current automaton to include the new observed behaviors when necessary. Function *get* is used to check if the automaton has a state q that matches the configuration and observed quality of z_i. If q is not found, a new state (also denoted q) is created to include observation z_i in the automaton. The function *Compose* handles two scenarios. First, lines 16-19 deal with the case where z_i has a configuration *conf* different from that of the current state $conf_c$. Second, lines 20-24 address the case where z_i has the same configuration *conf* as the current state, but the quality of the observed values $qty(\bar{o})$ changes, resulting in a different automaton state q. In both cases, the algorithm introduces a new transition from the current state q_c to q. In particular, when the transition occurs due to a change in the quality of the observed values (lines 20-24), it is denoted as $\mathbf{rd}(t')$, t' representing the time elapsed in q_c. Subsequently, whenever a new transition is added to the automaton, dictionary of lv must be updated with the mapping $q \mapsto ts$, and q is set as the new current state. Once all trace states have been visited, the updated automaton is returned in line 27.

The following results confirm the correctness of the algorithm by ensuring that all observed $\mathcal{N}$-traces utilized to create the TOA are recognized by the resulting automaton.

UNIVERSIDAD DE MÁLAGA

Algorithm Learn: Construction of the automaton $\mathcal{A}_k$ accepting a set of traces $\mathcal{T}$

Algorithm `Learn`($\downarrow \mathcal{T} = \{\pi^1, \cdots, \pi^p\}$, $\uparrow \mathcal{A}_p$)**:**
  $\mathcal{A}_0 := \langle \{q_0\}, q_0, \emptyset, \{q_0 \rightarrow (conf_d, 0)\} \rangle$;
  $lv = \{(q_0, 0)\}$;
  for $i := 1 \ldots p$ **do**
    $\mathcal{A}_i := Compose(\mathcal{A}_{i-1}, \pi^i, lv)$;
  return $\mathcal{A}_p$;
Function `Compose`($\updownarrow \mathcal{A}$, $\downarrow \pi = z_0 \ \ldots \ z_{n-1}$,$\updownarrow lv$)**:**
  $q_c := q_0$;
  $\mathcal{B}_0 := \mathcal{A}$;
  for $i := 0 \ldots n-1$ **do**
    $(conf_c, _) := f_{\mathcal{A}}(q_c)$;
    $\langle ts, conf, \bar{o} \rangle := z_i$;
    $q := \mathcal{B}_i.get(conf, qty(\bar{o}))$;
    if $q = None$ **then**
      $q := \mathcal{B}_i.newState(conf, qty(\bar{o}))$;
    if $conf \neq conf_c$ **then**
      $\mathcal{B}_i.addTran(q_c, \mathbf{cf}(conf), q)$;
      $lv.update(q, ts)$;
      $q_c := q$;
    else if $q \neq q_c$ **then**
      $t' := ts - lv.getValue(q_c)$;
      $\mathcal{B}_i.addTran(q_c, \mathbf{rd}(t'), q)$;
      $lv.update(q', ts)$;
      $q_c := q$;
    $\mathcal{B}_{i+1} := \mathcal{B}_i$;
  $\mathcal{A} := \mathcal{B}_n$;
  return $\mathcal{A}$;

UNIVERSIDAD DE MÁLAGA

Definition 10. For a given observed $\mathcal{N}$-trace $\pi = \langle ts_0, conf_0, \bar{o}_0\rangle \cdots \langle ts_{n-1}, conf_{n-1}, \bar{o}_{n-1}\rangle$, $\pi_{i\downarrow}$ $(0 \leq i < n)$ is defined as the $\mathcal{N}$-trace $(conf_0, \bar{o}_0) \cdots (conf_i, \bar{o}_i)$ obtained by excluding the last $n - i - 1$ states and eliminating the timestamp. If no states are discarded, $\pi_{\downarrow}$ is written instead of $\pi_{n-1\downarrow}$.

Lemma 1. Let $\mathcal{B}_0 = \mathcal{A}$ be the input TOA of *Compose* and $\pi = \langle ts_0, conf_0, \bar{o}_0\rangle \cdots \langle ts_{n-1}, conf_{n-1}, \bar{o}_{n-1}\rangle$ the observed $\mathcal{N}$-trace to be learned. For all $0 < i \leq n$, if $\mathcal{B}_{i+1}$ is the automaton constructed after the i-th iteration of the loop **for** (line 10), then it accepts $\mathcal{N}$-trace $\pi_{i\downarrow}$.

Proof. By induction on the length of π.

Case $i = 0$: Observe that all $\mathcal{N}$-traces start at $(0, conf_d, \perp)$, and therefore, as commented on in Definition 7, $s_0 = (q_0, 0, 0, conf_d, \perp)$, with $s_{0\downarrow} = (conf_d, \perp)$ is trivially accepted by $\mathcal{B}_1$.

Case $i > 0$: Since the automaton $\mathcal{B}_{i-1}$ has been constructed using the sequence $\langle ts_0, conf_d, \bar{o}_0\rangle \cdots \langle ts_{i-1}, conf_{i-1}, \bar{o}_{i-1}\rangle$, there must exist an automaton state q_{i-1} such that $f_{\mathcal{A}}(q_{i-1}) = (conf_{i-1}, qty(\bar{o}_{i-1})$.

By construction of transition systems, each sequence $\pi = \langle conf_0, \bar{o}_0\rangle \cdots \langle conf_i, \bar{o}_i\rangle$ of $\mathcal{O}(S, s_0, \overline{\longrightarrow})$ is the projection of, at least, a sequence of the form $\langle q_0, 0, 0, conf_d, \bar{o}_0\rangle \cdots \langle q_{i-1}, c_{i-1}, ts_{i-1}conf_{i-1}, \bar{o}_{i-1}\rangle$.

By construction of transition systems, each sequence $\pi = \langle conf_0, \bar{o}_0\rangle \cdots \langle conf_i, \bar{o}_i\rangle$ of $\mathcal{O}_($S, s_0, $\overline{\longrightarrow})$ is the projection of, at least, a sequence of the form $\langle q_0, 0, 0, conf_d, \bar{o}_0\rangle \cdots \langle q_{i-1}, c_{i-1}, ts_{i-1}conf_{i-1}, \bar{o}_{i-1}\rangle$.

Let $\mathcal{O}_i(S, s_0, \overline{\longrightarrow})$ and $\mathcal{O}_{i+1}(S, s_0, \overline{\longrightarrow})$ be the transition systems determined by $\mathcal{B}_{i-1}$ and $\mathcal{B}_i$, respectively.

Now, given the observed $\mathcal{N}$-states $z_{i-1} = (ts_{i-1}, conf_{i-1}, \bar{o}_{i-1})$, $z_i = (ts_i, conf_i, \bar{o}_i)$ in π. The (i-1)-th iteration of the function *Compose* sets q_c to q_{i-1}. Then, the function proceeds as follows:

- if $conf_{i-1} = conf_i$ and $qty(\bar{o}_{i-1}) = qty(\bar{o}_i)$ then $\mathcal{B}_i = \mathcal{B}_{i-1}$.

  Now, there exists a transition in the transition relation determined by $\mathcal{B}_i$ of the form $\langle q_{i-1}, c_{i-1}, ts_{i-1}conf_{i-1}, \bar{o}_{i-1}\rangle \xrightarrow{\mathbf{dl}} \langle q_i, c_i, ts_i conf_i, \bar{o}_i\rangle$, which means that $\langle conf_0, \bar{o}_0\rangle \cdots \langle conf_i, \bar{o}_i\rangle$ is in $\mathcal{O}_i(S, s_0, \overline{\longrightarrow})$, and therefore it is accepted by $\mathcal{B}_i$.

- There are two other possible cases that provoke the extension of $\mathcal{B}_{i-1}$ including a new automata transition, and a new state, if it is necessary. Anyway, the proof for these cases is similar to the previous one.

■

Lemma 2. For every $1 \leq i \leq p$, the automaton $\mathcal{A}_i$ returned by the *Compose* function

UNIVERSIDAD DE MÁLAGA

(line 5) recognizes the collection of $\mathcal{N}$-traces $\{\pi^1_\downarrow, \cdots, \pi^i_\downarrow\}$.

Proof. By Lemma 1, each automaton $\mathcal{A}_i$ accepts the trace $\pi^i_\downarrow$, and since $\mathcal{A}_n$ contains all the states and transitions of all $\mathcal{A}_i$ (with $1 \leq i \leq n-1$), we can easily conclude that $\mathcal{A}_n$ accepts all the traces of $\{\pi^1_\downarrow, \cdots, \pi^i_\downarrow\}$. ■

Theorem 1. Let $\mathcal{A}$ be the TOA obtained from the algorithm Learn based on the observed $\mathcal{N}$-traces $\mathcal{T} = \{\pi^1, \cdots, \pi^p\}$. Then the set of $\mathcal{N}$-traces $\mathbf{T} = \{\pi^1_\downarrow, \cdots, \pi^p_\downarrow\}$ is recognized by $\mathcal{A}$.

Proof. It follows trivially from Lemma . ■

The convergence of the algorithm *Learn* is based on the number of automaton states created from the set $\mathcal{T}$ that serves as its input. If we assume that $\mathcal{T}$ is finite and that each trace in $\mathcal{T}$ has a finite number of states, then the termination of algorithm *Learn* is straightforward. Nonetheless, if we consider that the automaton is continuously updated with new observed traces to capture new behaviors, the convergence is influenced by the number of different feasible network configurations (the size of the set $\mathcal{C}$) and the number of potential abstract KPI values (the size of $img(qty)$). Although it is assumed that $img(qty)$ is finite, it can also be assumed, according to [28], that there exists a finite number of practical configuration values. As a result, the constructed automaton is finite.

It is important to note that the *Learn* function produces an automaton that depicts the behavior of the network during TSN sessions characterized by a particular traffic pattern. Given that TSN defines a limited range of potential traffic patterns [23], and the traffic pattern remains constant within a session, our approach involves creating a series of unique automatons, each representing the behavior of the network under a different traffic pattern. Consequently, during the prediction and reconfiguration stages, the automaton associated with the specified traffic pattern will be exploited.

4.3.4 Automata for Prediction and Reconfiguration

This section details the algorithm *Predict*, which is designed to predict the deviation of the observed KPIs from the expected quality values and to suggest feasible alternative network configurations based on a learned TOA $\mathcal{A}$. The approach involves analyzing transitions of type $\mathbf{rd}(n)$ within $\mathcal{A}$ (i.e. those that indicate a deviation in the quality of the KPIs). Therefore, if the network remains in an automaton state where a degradation in KPIs is detected after n time intervals, the algorithm reports applying a reconfiguration procedure, if feasible.

The algorithm runs concurrently with the network. Both the algorithm and the network interact through channels *cin* and *cout*. The network periodically transmits observed $\mathcal{N}$-states in the format $\langle ts, conf, \bar{o} \rangle$ through *cin*, while the algorithm uses *cout* to transmit the prediction results. The Communicating Sequential Processes (CSP) notation is used to represent the sending (!) and receiving (?) of messages.

UNIVERSIDAD DE MÁLAGA

When a new observed $\mathcal{N}$-state *nextState* is received by *Predict*, it reports via channel *cout* with one of the following responses:

- *Continue*: Indicates that *nextState* meets the expected quality based on its configuration, which does not require further action.
- *Reconf*(*List*(*conf*)): Indicates that the network needs to be reconfigured with any configuration from *List*(*conf*) to prevent the observed KPIs from reaching an undesirable quality level. Determining the optimal configuration from the proposed list is outside the scope of the algorithm.
- *Warning*: Indicates that *nextState* has an undesirable quality level, but the algorithm is unable to recommend any reconfiguration action.
- *Unknown*: Indicates that *nextState* is not recognized by the automaton, leading to the termination of the algorithm in this case.

The algorithm *Predict* utilizes the function $desirable : \mathbb{Z} \to \mathbb{B}$, which takes an integer representing a quality value ($img(qty)$) and determines if this quality level is desirable for the observed KPIs. Furthermore, the algorithm employs the auxiliary functions *getConfigs* and *getRd*. The function *getConfigs* takes the TOA $\mathcal{A}$ and a state q_c of $\mathcal{A}$ as inputs and returns a list containing the configurations that label the outgoing transitions from q_c, leading to automaton states with desirable quality values. Similarly, the function *getRd* also takes TOA $\mathcal{A}$ and state q_c as input. It produces a natural number (greater than 0) associated with the outgoing transition from q_c labeled with $\mathbf{rd}()$, if it exists or 0 otherwise. Lastly, it is assumed that the network reconfiguration process takes a maximum of T_r time units. This constant is utilized in the algorithm to determine when a reconfiguration action should be executed.

As mentioned above, the algorithm *Predict* operates by analyzing the state of the network (line 4) received from the channel *cin* and providing a response through *cout*. To determine the appropriate response, it navigates through TOA $\mathcal{A}$, using the variable q_c (which is set to the initial state of the automaton) and the variable t to record the time when the current state q_c was last accessed. The algorithm initially verifies if the TOA accepts the sequence of network states being monitored (lines 10-19). If the current observed state is not identified by $\mathcal{A}$, the algorithm outputs *Unknown* (line 33). This situation arises specifically in the following situations:

- There is no automaton state that corresponds to *nextState* (line 10).
- There is no transition from q_c to q, where q is the automaton state representing *nextState* (line 14).
- In $\mathcal{A}$, there is a transition of the form $\mathbf{rd}(n)$ from q_c to q, but the network has not yet spent at least n time units at the current state to allow for the transition (line 16).

Otherwise, *nextState* is accepted by $\mathcal{A}$, and variables q_c and t are correctly updated.

UNIVERSIDAD DE MÁLAGA

Lines 20-31 determine the appropriate actions based on the current observed $\mathcal{N}$-state.

- If the observed values do not meet the required quality, a message *Warning* is transmitted using *cout* (line 21).
- If the observed KPIs meet the desired quality and no future **rd**-transition occurs within a time period less than T_r, a *Continue* message is transmitted through *cout* (line 25).
- If the observed values still maintain the desired quality, but there is an upcoming **rd**-transition towards a state with undesirable quality, the algorithm inspects $\mathcal{A}$ to determine whether this can be prevented by performing a network reconfiguration (line 27). If there are no alternative network configurations available, a message *Warning* is sent (line 29); otherwise, a message *Reconf* that contains a list of potential new configurations is sent (line 30).

Example 4. Continuing with the previous examples, let us assume that the TOA $\mathcal{A}$ shown in Figure 4.3 serves as the input to the algorithm *Predict*, with T_r set at 4 time units. The quality values 0 and 1 are deemed desirable for the KPI δ, indicating that *Predict* aims to prevent the occurrence of states q_2, q_4, and q_5. The algorithm is expected to receive the following sequence of observed $\mathcal{N}$-states via *cin*:

$$\langle 0, p_1, \perp\rangle \cdot \langle 4, p_1, 2ms\rangle \cdot \langle 8, p_1, 3ms\rangle \cdot \langle 12, p_1, 2ms\rangle \cdot \langle 16, p_1, 3ms\rangle \cdots$$

The initial $\mathcal{N}$-state observed is associated with q_0 in $\mathcal{A}$, while the second state represents a valid transition $\mathbf{rd}(4)$ from q_0 to q_1. The third and fourth states indicate remaining in state q_1. Up to this point, the algorithm has consistently responded to each observed $\mathcal{N}$-state with the message *Continue* as all states are recognized by $\mathcal{A}$, and the network has only spent 8 time units in q_1, which is below $16 - T_r$ (the time point at which q_5 could be reached). The fifth state also corresponds to staying in q_1 but, at this point, the network has been in this state for 12 time units, and within less than $T_r = 4$ time units, the system could transition to q_5. Consequently, the algorithm invokes the function *getConfigs* and retrieves a list containing a single alternative configuration, indicating the feasibility of switching the RLC AM mode from p_1 to p_2 to prevent undesirable quality. In this scenario, *Predict* sends the message *Reconf*$(\{p_2\})$ proposing the reconfiguration of the network to p_2. Depending on the network response, the subsequent observed $\mathcal{N}$-state may exhibit the configuration change (e.g. transition $\mathbf{cf}(p_2)$ from q_1 to q_3) or the progression of observed KPIs toward undesirable quality (e.g. transition $\mathbf{rd}(16)$ from q_1 to q_2). In the latter case, the subsequent prediction from *Predict* will be *Warning*.

In the context of implementation performance, it is essential that the monitoring, learning, prediction, and reconfiguration modules (refer to Section 5.1) operate seamlessly without any delays, especially when running continuously on the real network. Ultimately, these modules should function effectively in parallel without compromising the performance of the network during their interaction with it.

UNIVERSIDAD DE MÁLAGA

Algorithm Predict: Reconfiguration

```
1  Algorithm Predict(↓ A,↓ cin,↓ cout):
2      q_c := A.q_0;
3      t := 0;
4      cin?nextState;
5      if (nextState = None) then
6          goto line 34 ;
7      (ts, conf, ō) := nextState ;
8      qty := qty(ō);
9      q := A.get(conf, qty);
10     if (q = None) then
11         goto line 33;
12     else if (q ≠ q_c) then
13         label = A.getTrans(q_c, q);
14         if (label = None) then
15             goto line 33;
16         if (label = rd(n) && ts − t < n) then
17             goto line 33;
18         q_c := q;
19         t := ts;
20     if (!desirable(qty)) then
21         cout!Warning
22     else
23         m := getRd(A, q_c);
24         if (m = 0||m − (ts − t) > T_r) then
25             cout!Continue;
26         else
27             lconf := getConfigs(A, q_c);
28             if (lconf.isEmpty()) then
29                 cout!Warning;
30             else
31                 cout!Reconf(lconf);
32     goto line 4 ;
33     cout!Unknown;
34     return;
```


UNIVERSIDAD DE MÁLAGA

Function getConfigs

```
1 Function getConfigs(↓ A, ↓ q_c, ↑ lConf):
2     succ_d := A.getCfSucc(q_c).filter(desirable);
3     for q in succ_d do
4         lConf.add(A.getTrans(q_c, q));
5     return lConf;
```

Function getRd

```
1 Function getRd(↓ A, ↓ q_c, ↑ m):
2     trd := A.getRdTrans(q_c);
3     m := 0;
4     if trd ≠ None then
5         (q_c, q, m) := trd;
6         if desirable(q) then
7             m := 0;
8     return m;
```


UNIVERSIDAD DE MÁLAGA

Chapter 5

Implementation and Evaluation of the Proposal

Contents

5.1 Proposal Implementation 90
5.1.1 Monitoring 90
5.1.2 Learning 96
5.1.3 Prediction and Reconfiguration 99
5.2 Evaluation 105
5.2.1 Evaluation of the *Learn* algorithm 107
5.2.2 Evaluation of the AutomAdapt Initial Learning Phase 110
5.2.3 Evaluation of the Complete AutomAdapt Operation Cycle . . . 111
5.2.4 Example Trace 113
5.2.5 Analysis of Results 116

Summary: This chapter outlines the implementation of the proposal (code available at [225]), including the details of each module. In addition, it includes the evaluation conducted at the University of Malaga testbed, presenting a complete example trace and its analysis of the results obtained.

UNIVERSIDAD DE MÁLAGA

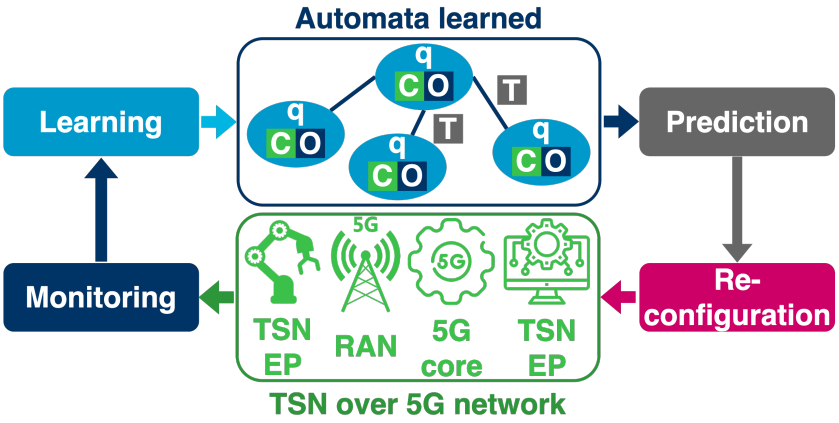


Figure 5.1: Proposal implementation architecture

5.1 Proposal Implementation

This chapter provides details of the implementation of the proposal presented in the previous chapter. The key aspects of AutomAdapt, which is the tool that includes the **monitoring**, **learning**, **prediction** and **reconfiguration** modules, as well as the elements and procedures involved, are presented below. In addition, the code for each module is available in the AutomAdapt repository [225]. Furthermore, this chapter offers an analysis of the complete solution evaluation results, which were performed on the TSN over 5G testbed developed during the thesis, which is described in Section 3.1. This allows real data to be obtained to support the design decisions that have been made during the development of this thesis.

5.1.1 Monitoring

The **monitoring** module is responsible for gathering data from various points on the network. This data includes all pertinent header information for packets sent and received from TSN endpoints, through the RAN, 5G core network, and TSN translators. At present, the monitoring module has access to all of these network components. Nevertheless, the most important thing for monitoring implementation and, in turn, for the learning module is to acquire this information at the TSN endpoints, thus enabling the end-to-end KPIs to be obtained.

Monitoring operation

Figure 5.2 shows the details of the implementation of the monitoring module, which is mainly powered by the open source distributed event streaming platform called *Apache Kafka* [244]. It is used to collect and process information from streaming events or data (for example, TSN endpoint traffic or connection establishment / release) through *Kafka streams*. Additionally, the data are organized into specific topics.

UNIVERSIDAD DE MÁLAGA

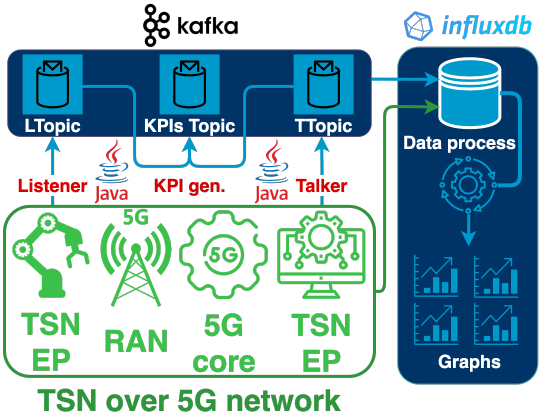


Figure 5.2: Monitoring module implementation

Java programs (called Listener and Talker) are in charge of capturing all packets in the TSN endpoints, processing them to extract all the significant information (e.g. timestamp and packet size) and inserting this information into the analogous Kafka topic (for example, LTopic stores the information related to the listener). Then, another Java program (called KPI generator) uses Kafka streams to handle these data, generate the KPIs (throughput, delay, jitter, and packet loss), and inject the resulting KPIs into another Kafka topic. Eventually, these KPIs are inserted into a high-speed read and write database called Influx Data Base (InfluxDB) [245], where they can be processed using different functions such as sum or average. It is worth mentioning that the 5G network configuration (for example, RAN parameters for PHY and RLC layers) is also read and stored in the data base. It allows us to represent the status of the network (KPIs and configuration) in real time using several types of graph as an additional function for the network operator, although it is not required for automatic operation. Note that we can retrieve fine-grained reports, even differentiating between each packet sent across the network.

Talker and Listener Codes

The Talker and Listener programs capture packets using the Pcap4J library [246] at the specific network interface of the TSN endpoints. Additionally, a filter is set to ensure that relevant packets are captured. In this case, we only capture packets that include the 802.1Q tag, which is used for traffic separation. For example, the use of Priority Code Point (PCP) field to indicate the priority of different classes of traffic.

```
1        final PcapHandle handle_sender = nif.openLive(SNAPLEN,
    PcapNetworkInterface.PromiscuousMode.PROMISCUOUS, READ_TIMEOUT);
2        // Set a filter to only listen for 802.1Q packets
3        String filter = "ether proto 0x8100";
```


UNIVERSIDAD DE MÁLAGA

```
4         handle_sender.setFilter(filter, BpfProgram.BpfCompileMode.
    OPTIMIZE);
```

Listing 5.1: Packet filter

Once the sent or received packet is captured, we obtain the packet timestamp in nanoseconds and the packet length, both identified by the packet id, which is a unique identification number used later for the calculation of the KPIs (e.g. delay). This information is published as events on a Kafka topic. It should be noted that the Pcap4J library has millisecond precision by default and has been recompiled to enable nanosecond precision. Also, to identify each packet, the packet id value is incremented by 1 each time a packet is sent. To do this, 6 bytes of the payload are used, resulting in 256^6 possibilities. Considering that the most restrictive TSN traffic profile [23] sends a packet every 100us, the sequence number would not be repeated for years on a connection between 2 TSN endpoints. This ensures that we have unique identification numbers and therefore a correct calculation of the KPIs.

```
1         PacketListener listener =
2                 packet_sent -> {
3                     int index = packet_sent.getPayload().getPayload().
    getPayload().toString().lastIndexOf(":" ) + 2;
4                     String packet_id = packet_sent.getPayload().
    getPayload().getPayload().toString().substring(index,index + 11);
5                     long packet_timestamp = TimestampToEpochNano(
    handle_sender.getTimestamp());
6                     long packet_length = packet_sent.length();
7                     try {
8                         //Time producer - insert the packet id and
    timestamp
9                         producer.send(new ProducerRecord<>(delay_topic,0,
     packet_id, packet_timestamp)).get();
10                        //Data producer - insert the packet id and packet
     length
11                        producer.send(new ProducerRecord<>(
    throughput_topic, 0, packet_id, packet_length)).get();
12                    } catch (InterruptedException | ExecutionException e)
     {
13                        e.printStackTrace();
14                    }
15                };
```

Listing 5.2: Data extraction from headers

KPIs generator code

This code is responsible for the calculation of the end-to-end KPIs based on the events published in the Kafka topics, that is, the information previously written for the Talker and Listener programs. It consists of a continuous loop that reads from Kafka topics to calculate the KPIs, which are delay, jitter, throughput, and packet loss (note that this calculation is done together with the throughput).

UNIVERSIDAD DE MÁLAGA

```
        while (true) {
            // Poll for records
            ConsumerRecords<String, Long> delayRecords = delayC.poll(
    Duration.ofMillis(1000));
            ConsumerRecords<String, Long> thrSRecords = thrSC.poll(
    Duration.ofMillis(1000));
            ConsumerRecords<String, Long> thrRRecords = thrRC.poll(
    Duration.ofMillis(1000));

            // If records are found, calculate the KPIs
            delayCalculation(delayRecords);
            jitterCalculation(delayRecords);
            throughputCalculation(thrSRecords, thrRRecords);
            }

        }
```

Listing 5.3: KPIs calculation

For example, the delay is calculated by comparing the packet id in one-second windows. In the case of a match, that is, the packet sent is the same as the one received, the timestamps are subtracted to obtain the end-to-end delay and published in the corresponding topic.

```
        // Defition of the processing topology of the Streams application
        final StreamsBuilder builder = new StreamsBuilder();
        final KStream<String, Long> s_packets = builder.stream("
    delay_talker_topic");
        final KStream<String, Long> r_packets = builder.stream("
    delay_listener_topic");

        // Join the two streams using a window of 1 second
        ValueJoiner<Long, Long, Long> valueJoiner = (leftValue,
    rightValue) -> {
                return rightValue - leftValue;
        };

        KStream<String, Long> result_packets = s_packets.join(r_packets,
                valueJoiner,
                JoinWindows.of(Duration.ofSeconds(1)));

        // Send the delay calculation result to the delay_topic
        result_packets.to("delay_topic", Produced.with(stringSerde,
    longSerde));
```

Listing 5.4: Delay calculation

Finally, an example of how the results obtained and stored in the Influx database can be graphically represented is shown in Figure 5.3, specifically the evolution of these KPIs during 24 hours of testing.

UNIVERSIDAD DE MÁLAGA

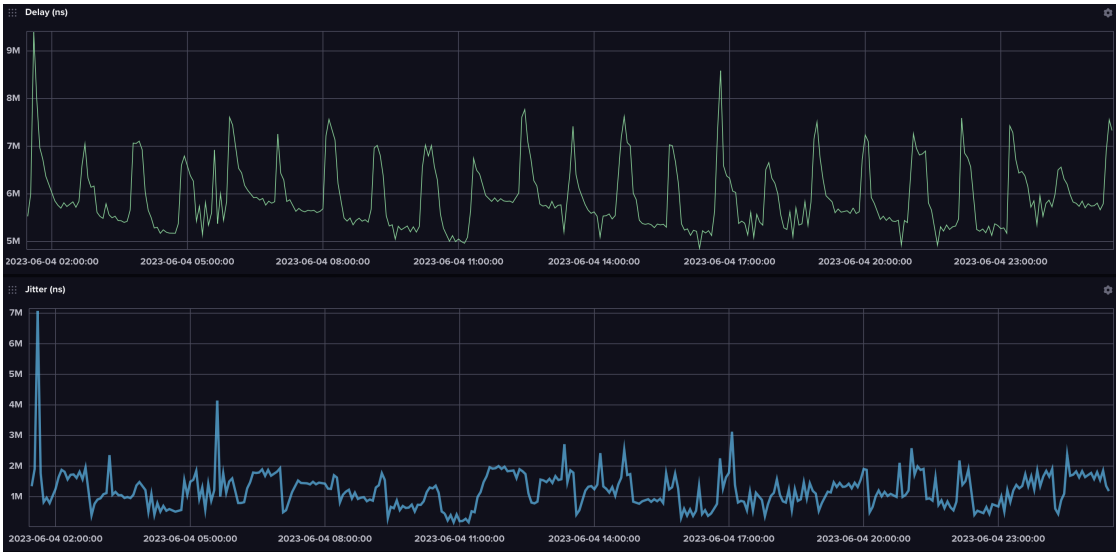


Figure 5.3: Delay and Jitter graphs plotted in InfluxDB

RAN configuration monitor code

As mentioned above, we can also read the RAN parameters, including the different layers of the protocol stack. This is possible for both 5G network options available on the testbed. In the case of Nokia Base Transceiver Station (BTS), the current configuration is downloaded with the Extensible Markup Language (XML) extension using an API provided by Nokia as presented below.

```
# Auxiliary variables
currentdate=$(date +'%Y-%m-%d_%H:%M:%S')
name="micro-config"
extension=".xml"
OUTPUT_FILE="$name-$currentdate$extension"
DATA={\"requestId\":1,\"parameters\":{\"name\":\"generateScf\"}}

# Get the configuration from the BTS
"$ADMIN_CLI" --bts-username="$USERNAME" --bts-password="$PASSWORD" --bts-
    host="$HOST" --bts-port="$PORT" --format="$FORMAT" --output-file="
    $OUTPUT_FILE" --data="$DATA"

# Convert the XML file to a SQL file
python3 xmltodb.py "$OUTPUT_FILE" "$1"
```

Listing 5.5: Reading Nokia radio parameters

Then, this XML RAN configuration file is parsed. After that, the requested objects are obtained. In this case, the RLC AM is acquired, which contains 6 parameters corresponding to the downlink and 6 to the uplink (see Section 3.3 for further details) and are stored in the InfluxDB database.

UNIVERSIDAD DE MÁLAGA

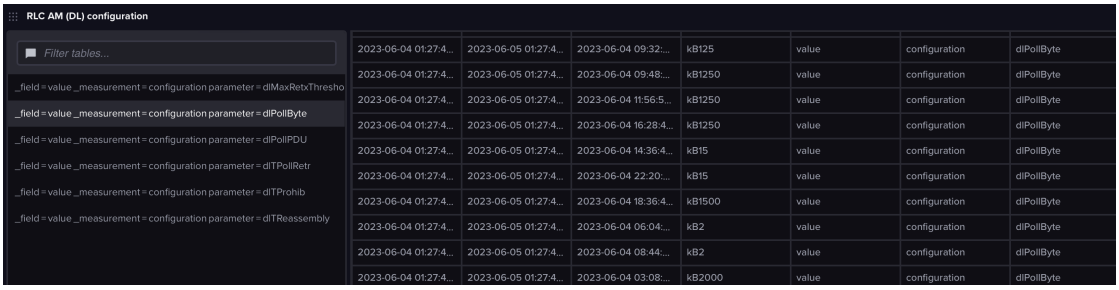


Figure 5.4: RLC AM downlink parameters in InfluxDB

```
# Getting the required RLC AM object
for nokia_object in mon:
    if (nokia_object.attrs['distName'] == 'MRBTS-2/NRBTS-1/NRDRB_RLC_AM-'
    +profile_id):
        rlc_o = nokia_object

# Looping through the RLC AM object and writing the data to the database
for iterator in rlc_o:
    if (iterator != '\n'):
        childname = iterator.attrs['name']
        childvalue = iterator.contents[0]
        p = influxdb_client.Point("configuration").tag("parameter",
    childname).field("value", childvalue)
        write_api.write(bucket=bucket, org=org, record=p)
```

Listing 5.6: Storing Nokia radio parameters in InfluxDB

Similarly, the RAN configuration can be exported and read from Keysight UXM 5G. Then, the configuration file is parsed to fetch the desired parameters, which has the standard command for programmable instruments (scpi) extension in this case. Again, these data can be stored in the InfluxDB database. The following example shows how to extract the MCS table and index values from the physical layer (note that exporting data to InfluxDB is done in the same way).

```
# Read the SCPI file
with open('UXM5G_config.scpi', 'r') as file:
    scpi_content = file.read()
    # Find the line that contains the MCSTable value
    line = re.search(r'BSE:CONFig:NR5G:SCHeduling:QCONFig:DL:MCSTable Q\d
    +', scpi_content).group()
    # Extract the MCSTable value
    mcstable_value = re.search(r'Q(\d+)', line).group(1)
    # Find the line that contains the MCS index
    line = re.search(r'BSE:CONFig:LTE:CELL1:PHY:UL:IMCS:SFRame0 \d+',
    scpi_content).group()
    # Extract the MCS index value
    mcs_index = re.search(r'SFRame0 (\d+)', line).group(1)
```

Listing 5.7: Reading Keysight UXM 5G radio parameters in the database

UNIVERSIDAD DE MÁLAGA

5.1.2 Learning

The **learning** module is in charge of creating *Automata*, with each automaton representing the behavior of the network during TSN sessions with specific traffic patterns. Figure 5.5 presents how the learning module builds automata, including the characterization of the so-called *network states* for both phases related to the algorithm *Learn* detailed extensively in Chapter 4, in addition to the ability to communicate with the monitoring module to retrieve network traces that require learning and then convert them into automata.

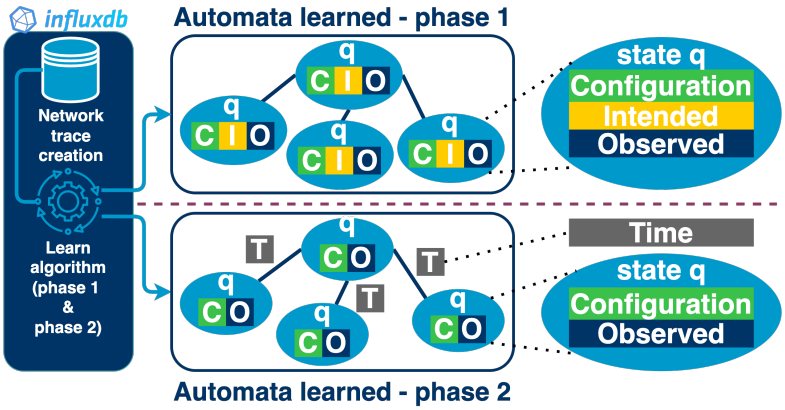


Figure 5.5: Learning module implementation

The implementation of this module has been developed in two main stages. First, by following a passive automata learning approach corresponding to phase 1 as introduced in Section 4.2 and, therefore, capturing network traces in advance to be learned and building the automaton by the algorithm later. Second, by introducing a new kind of automaton corresponding to phase 2 as described in Section 4.3, called Traffic Oriented input/output Automata (TOA) to represent the evolution of the network in relation to time. Note that *intended* values only are included in the phase 1 network states, since each automaton learned in phase 2 corresponds to a single traffic pattern that already includes the specific requirements (such intended values). Since phase 2 is considered an evolution of phase 1, this section focuses on this second phase, where *time* is part of the transitions between states, enriching the information of the automaton. Although algorithm Learn is oriented to the construction of real-time automata (TOA) based on the current behavior of the network, it also allows the use of an automaton built previously during an *initial learning phase* as a base for the *online learning*, that is, all AutomAdapt modules are active. First, the operation of the learning module is explained during this initial learning phase, where the prediction and reconfiguration modules are not involved. Second, when the learning module is considered to have built a convenient automaton for prediction and reconfiguration, these modules are activated to close the complete cycle, enabling the objective of this thesis. Note that the learning module continues to learn after activating the rest of the modules (called online learning), feeding back to

UNIVERSIDAD DE MÁLAGA

the learned automaton (e.g. adding new states or updating states already present in the automaton).

Initial Learning Phase

As mentioned above, the initial learning phase serves to build a source automata (TOA) corresponding to the behavior of the system in a certain number of network configurations and traffic profiles. This is essential for the prediction and reconfiguration modules, as they perform their function with the learned information. Consequently, KPI deviation predictions and network reconfigurations can be performed based on the information available in the automata. For this purpose, different traffic patterns are generated while the 5G network configuration is randomly modified. The monitoring module is in charge of obtaining the observed KPIs, while the learning module builds the automaton. Note that the other modules are not active and therefore no prediction is reported and no reconfigurations are performed through the prediction and reconfiguration modules, respectively.

Depending on the level of detail desired by the network operator to build the automaton, there is a key parameter called *cycle time*, which defines how often information related to observed KPIs is collected. This can lead to the creation of new states more or less frequently (see the action labeled $\mathbf{rd}(n)$ in Section 4.3.3). For example, by defining a short monitoring cycle (i.e. obtaining the observed KPIs every 1s), a higher level of detail is achieved than when defining a longer monitoring cycle (e.g. 10s), since some KPI deviations are hidden when averaging the data. Subsequently, the number of network states depends on the level of detail. Note that the time it takes for the application to create the automaton using network traces previously captured will be longer as the more level of detail is added, since a greater number of operations are needed to be performed. In addition, as expected, the number of states will grow over time as the aforementioned random configuration changes or possible deviations in the observed KPIs occur.

Table 5.1 presents an example of several automata learned at different detail levels, that is, using different values of the cycle time. Each automaton is started to build from the same instant of time for 24 hours to show how this parameter affects the number of states obtained. In this case, the number of states decreases when using a longer cycle time, because the average of the observed KPIs masks some deviations produced during the TSN traffic session. Similarly, the creation time is lower if we use a high cycle time, since the number of computations decreases considerably. Inevitably, there is a trade-off between the time of creation of the automata and its level of detail. Note that the number of configurations is the same regardless of the cycle time, since we operate in the same time period. Therefore, the network operator selects the appropriate cycle time according to the current needs of the network. For example, by selecting a cycle time that allows one to obtain as much information as possible in the time established for the initial learning phase.

It should be noted that the definition of the *qty* function also directly influences the level

UNIVERSIDAD DE MÁLAGA

Automaton ID	*Automaton build time*	*Cycle time*	*Number of states*	*Number of configurations*	*Creation time*
1_1	24 h	1 s	342	90	9212 s
1_5	24 h	5 s	259	90	1866 s
1_{10}	24 h	10 s	234	90	934 s
1_{30}	24 h	30 s	205	90	316 s
1_{60}	24 h	60 s	187	90	146 s
1_{120}	24 h	120 s	170	90	57 s

Table 5.1: Example of a learned automaton with different values of the cycle time.

qty	*delay* (ms)	*jitter* (ms)
-30	[12-2e^6)	[10-2e^6)
-18	[10-12)	[5-10)
-8	[10-12)	[1-5)
-1	[0-7)	[5-10)
1	[0-7)	[0-1)
2	[0-7)	[1-5)
12	[7-10)	[0-1)
13	[7-10)	[1-5)

Table 5.2: *qty* function values used to learn automaton during the initial learning phase for a given traffic pattern

of details of the automaton. In particular, the full range of observed KPIs $(-\infty, +\infty)$ must be defined in the implementation to abstract the values. In this case, the values of *qty* are presented in Table 5.2. An additional value of the *qty* function is defined (in this case, *qty* = -30) to consider any range of values different from those defined by the network operator during the learning process. The results and conclusions obtained in this initial learning phase can be extrapolated for different values of both the cycle time and the qty function.

Online Learning

Once the automata (TOA) of the initial learning phase are ready, the so-called *online learning* is activated. This consists of activating the prediction and reconfiguration modules, which are explained in detail below. Then, the learning module, specifically the algorithm *Learn*, uses the *Compose* function to progressively expand the learned TOA. Note that this real-time learning can also be performed during the initial learning phase of learning, although it is not necessary since the prediction and reconfiguration modules are not in operation. The flow chart that includes the main functions of the learning module is depicted in Figure 5.6. The monitoring module is in charge of providing the

UNIVERSIDAD DE MÁLAGA

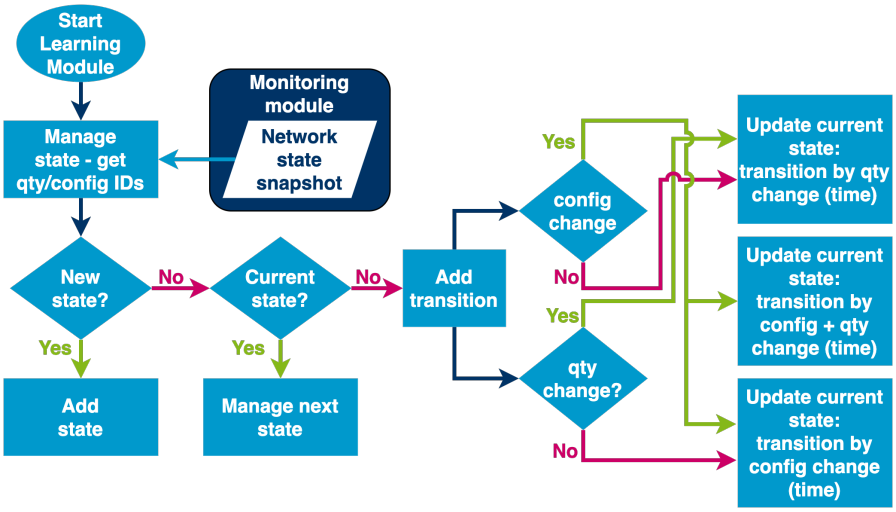


Figure 5.6: Learning module flow chart

network state snapshot as introduced before, and then the learning module manages that state to perform the corresponding action (e.g. add it to the automaton if it is a new state or add the necessary transition to another state). Due to the length of these codes, they are not shown below. However, complete codes for both the initial learning phase and online learning are available in the AutomAdapt repository [225].

5.1.3 Prediction and Reconfiguration

On the one hand, the prediction module is in charge of detecting in advance a possible deviation from the observed KPIs from the requirements demanded by the traffic. If a deviation is identified, the module attempts to obtain a list of suitable 5G network configurations to continue to meet the requirements. It should be noted that the selection of the most suitable configuration goes beyond the capabilities of the Predict algorithm. Subsequently, the entire list would be sent to the reconfiguration module. Depending on the reported message based on the new $\mathcal{N}$-state *nextState* observed by the algorithm (see Section 4.3.4), the situation may vary.

1. *Continue* message. In this case, no further action is required as traffic requirements are being met.
2. $Reconf(List(conf))$ message. The algorithm has detected a possible observed deviation from the KPIs. Then, uses an auxiliary function to get the list of suitable configurations already learned to be sent to the reconfiguration module, which will be in charge of selection a final candidate.
3. *Warning* message. Similarly to the previous message, the algorithm detects a deviation. However, it is not possible to suggest any configuration. Therefore, it is likely that at some point the expected quality will not be met.

UNIVERSIDAD DE MÁLAGA

4. *Unknown* message. This situation leads to the termination of the algorithm, since it is a situation that has not been already learned, that is, it is not included as network state in the automaton.

The case where the prediction and reconfiguration modules are not able to help maintain traffic requirements (messages *Warning* and *Unknown*) has been considered in Section 6.2, where possible complementary and alternative solutions to make the system responsive in all cases are proposed as future work of this thesis.

There exist several methods to sort the list of suitable configurations retrieved from the prediction algorithm. An example is to sort the list according to the time elapsed in the states to which each of the configurations belongs. In other words, the configuration that has been operational for the longest duration would be chosen. Another example can be sorting based on the KPI values observed with each configuration. For example, in the case where current traffic has demanding delay requirements, the configuration associated with the lowest delay would be selected. In addition, the network operator can define complementary parameters such as the following:

- *Configuration goal time.* The minimum time required to consider a suggested configuration for network reconfiguration (e.g. 5 minutes, 1 hour). It should be noted that a longer time value can lead to a more optimal network configuration, avoiding future reconfigurations in a short period of time. However, this parameter must be chosen based on the values of the automaton learned in the initial learning phase because if a restrictive (very high) value is chosen, the number of available configurations may be considerably reduced or even be 0. A possible solution could be to automatically reduce this time in case no configuration in the list meets the time initially set.
- *Reconfiguration number.* Number of times a reconfiguration can be reapplied during the same traffic session (e.g. 0 or 3). For example, a value of 0 does not allow the same configuration to be chosen again, so it would not be considered in the list of possible configurations the next time a prediction is made, even if it met the requirements. Note that this can lead to an empty list of suitable configurations over time, since the automaton cannot suggest new configurations, i.e. it is not able to generate configurations that have not been learned.
- *Stability time.* Time considered necessary for the stabilization of the system after a reconfiguration (e.g. 30 seconds, 5 minutes), during which no reconfiguration is applied again.

For example, the auxiliary function to get the list of suitable configurations is defined below. First, we filter the states of the automaton that math with the qty objective (i.e. the qty value that corresponds to the expected quality). Second, all the stored values corresponding to the elapsed time in that state are summed and sorted in descending order (the learned state with the highest value is the first one). Then, the minimum elapsed time value of each state is checked to ensure that it is greater than the set *configuration goal time.*

UNIVERSIDAD DE MÁLAGA

Note that in this case, we are combining one of the complementary parameters mentioned above (configuration goal time) with one of the criteria for selecting a configuration from the list (longest elapsed time in that state) directly in the function. There is also the possibility to extract the complete list and include one or more complementary parameters together with one of the configuration selection modes at a later time. Finally, the specific configuration parameters for each state are extracted through *getConfigParams* function and returned to the configuration module.

```
def getSuitableConfigurations(qty_objective, goal_time):
    global automaton_df
    suitable_configs = []

    #Filter automaton_df by desired qtyID
    automaton_df_qty_obj = automaton_df[automaton_df['qtyID'] ==
    qty_objective]
    #Sum elapsed_time values for each configID
    automaton_df_qty_obj['sum_elapsed_time'] = automaton_df_qty_obj['
    elapsed_time'].apply(lambda x: sum(x))
    #Sort by sum_elapsed_time in descending order
    automaton_df_qty_obj.sort_values(by=['sum_elapsed_time'], inplace=
    True, ascending=False)

    for i in range(len(automaton_df_qty_obj)):
        #Check if the minimum elapsed_time is greater than the goal_time
        if ((automaton_df_qty_obj.iloc[i]['sum_elapsed_time'].min()) >=
    goal_time):
            suitable_configs.append(getConfigParams(automaton_df_qty_obj.
    iloc[i]['configID']))
    return suitable_configs
```

Listing 5.8: Get the list of suitable configurations

Once the configuration to be applied is selected, the reconfiguration module is responsible for performing these changes in the 5G network automatically. For example, the following code applies the configuration chosen to the Nokia RAN using a subprocess call to another script. Specifically, this script changes 12 RLC AM parameters (6 for DL and 6 for UL). In addition, the applied reconfiguration is added to a list, which can be useful to apply another of the defined complementary parameters, namely the *reconfiguration number*.

```
def networkReconfiguration(selectedConfig):
    global old_config_id
    print("Network reconfiguration started (ETA: 40 seconds)")
    subprocess.call(["/home/schempp/Documents/NokiaAdminCLI/
    setConfigMicroArguments.sh", selectedConfig[0], selectedConfig[1],
    selectedConfig[2], selectedConfig[3], selectedConfig[4],
    selectedConfig[5], selectedConfig[6], selectedConfig[7],
    selectedConfig[8], selectedConfig[9], selectedConfig[10],
    selectedConfig[11]])
    if (len(selectedConfig) == 13):
        addReconfiguration(old_config_id, selectedConfig[12], datetime.
    utcnow().isoformat())
        old_config_id = selectedConfig[12]
```


UNIVERSIDAD DE MÁLAGA

```
        print("Network reconfiguration completed")
        return True
    else:
        old_config_id = getConfigIDhash(selectedConfig)
        print("Network reconfiguration failed")
        return False
```

Listing 5.9: Apply the selected configuration to the Nokia RAN

The configuration methods available in the testbed presented in Section 3.1 are now presented. Specifically, the configuration interfaces for the Nokia RAN and the Keysight UXM 5G, detailing the process for applying automatic reconfigurations that can be triggered by the prediction module and applied using the reconfiguration module.

Nokia RAN

The Nokia RAN can be configured in two main ways. On the one hand, through the dashboard called Web Element Manager (software version SBTS22R3) presented in Figure 5.7, which allows the management of all aspects of the RAN (e.g. checking current status and IP traffic capture) and the configuration of hundreds of radio parameters using a graphical interface. These parameters are structured according to the eXtensible Markup Language (XML) format and were defined in Section 3.3. Although this method is more intuitive and easier to use, it does not allow you to have control over the system in an efficient way.

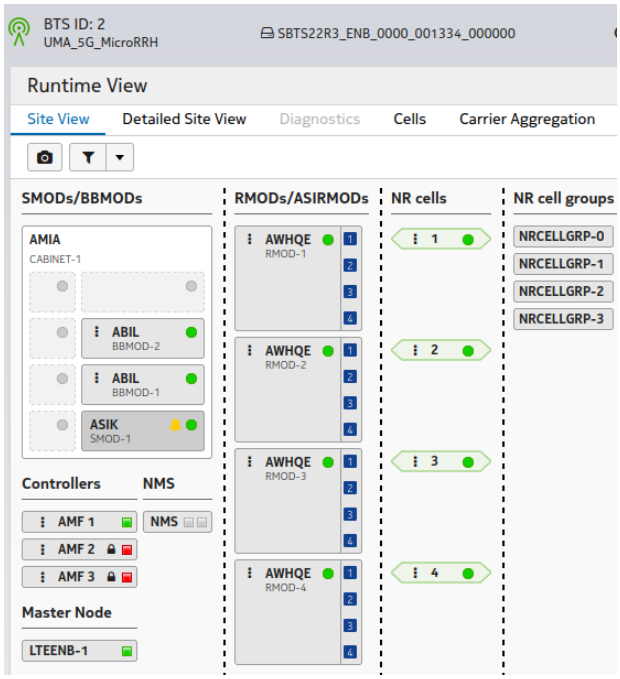


Figure 5.7: Nokia Web Element Manager

UNIVERSIDAD DE MÁLAGA

On the other hand, through the command-line interface (CLI) client, since the BTS has a CLI server to listen to the requests. Unlike the previous method, this one is neither accessible nor user-friendly. However, it enables better control and efficiency over the system, allowing one to automate procedures (e.g. network reconfiguration after a prediction). Consequently, this technique has been used for implementation. The following example shows how to apply a network reconfiguration to the Nokia RAN using CLI. In particular, this script defines the XML path to the node to be modified, the RLC AM in this case (several RLC AM profiles can be created, and therefore the specific value has to be specified). Then, all the parameters that compose the RLC AM profile (12 in total in this example) are assigned and modified in the corresponding XML path using the values of the previously chosen configuration (only the values for TReassembly in DL/UL are shown here). Afterwards, a DRB object is generated that contains the created RLC AM profile. Finally, the configuration is applied to the specific BTS, indicating its credentials (user, password, IP address, and port).

```
#RLC AM profile Xpath
DISTNAME=MRBTS-2/NRBTS-1/NRDRB_RLC_AM-$PROFILE

#Assign the value of the received TReassembly parameter (DL/UL)
dlTReassembly=$6
ulTReassembly=${12}

#Modify the value of the TReassembly parameter (DL/UL)
dlTReassembly_Param={\"parameterName\":\"dlTReassembly\",\"operation\":\
    "update\",\"value\":\"$dlTReassembly\"}
ulTReassembly_Param={\"parameterName\":\"ulTReassembly\",\"operation\":\
    "update\",\"value\":\"$ulTReassembly\"}

#Add all the values of the RLC AM parameters
DRB_AM={\"distName\":\"$DISTNAME\",\"operation\":\"update\",\"
    parameters\":[$dlTPollRetr_Param\,$ulTPollRetr_Param\,
    $dlTReassembly_Param\,$ulTReassembly_Param\,$dlTProhib_Param\,
    $ulTProhib_Param\,$dlMaxRetxThreshold_Param\,$ulMaxRetxThreshold_Param
    \,$dlPollPDU_Param\,$ulPollPDU_Param\,$dlPollByte_Param\,
    $ulPollByte_Param]}

#Add DRB_AM data DRB1
DATA_DRB="{\"requestId\":1,\"parameters\":{\"name\":\"recommission\",\"
    parameters\":{\"planObjects\":[$DRB_AM],\"shouldBeActivated\":true}}}"
#Apply DRB1 configuration
$ADMIN_CLI --bts-username=$USERNAME --bts-password=$PASSWORD --bts-host=
    $HOST --bts-port=$PORT --format=$FORMAT --data=$DATA_DRB
```

Listing 5.10: RLC AM parameters configuration using the Nokia RAN CLI client

Keysight UXM 5G

In the same way, the Keysight UXM 5G has two main methods for configuration. First, the Test Application (TA) displayed in Figure 5.8, which is a graphical interface that

UNIVERSIDAD DE MÁLAGA

allows full control of the system, allowing the configuration of hundreds of 5G network parameters.

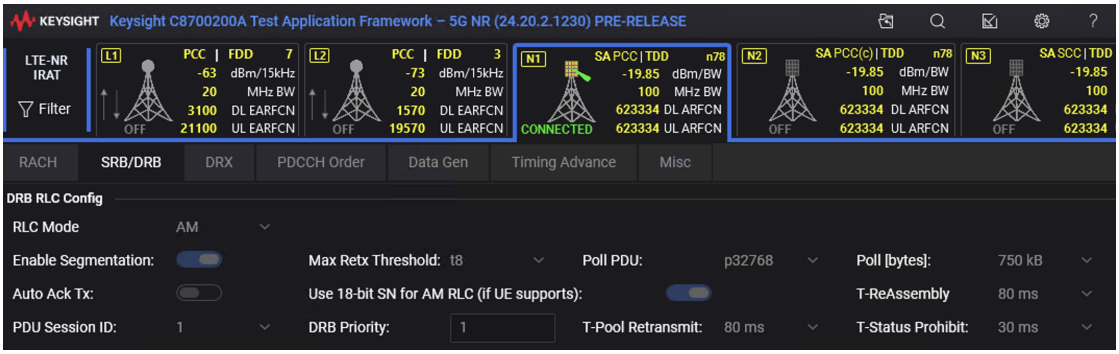


Figure 5.8: Keysight UXM 5G Test Application

Second, the Standard Commands for Programmable Instruments (SCPI) commands, which are sent to the UXM 5G to modify the available parameters. Again, this approach enables the automation of network reconfiguration and is therefore the chosen one for implementation. Figure 5.9 presents an example of an SCPI command to configure the RLC AM *tReassembly* timer to 0 ms on the UXM 5G. Note that, similarly, a script can be used to automatically modify all parameter values of the selected configuration.

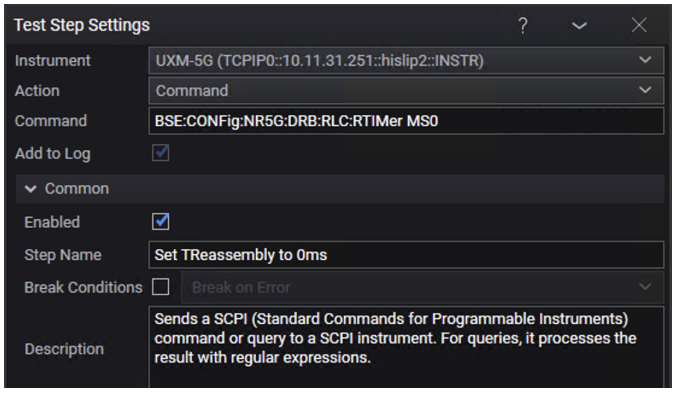


Figure 5.9: Keysight UXM 5G Test Automation Platform

Regardless of where the reconfiguration is performed (Nokia RAN or Keysight UXM 5G), these changes must be reflected in the 5G UE. To do this, an RRC reconfiguration message is sent from the radio to the UE. An example of this procedure is shown in Figure 5.10, where some changes in some parameters of the RLC layer (e.g. $t_{Reassembly}$ and $t_{StatusProhibit}$ for the downlink direction) captured with the Keysight NEMO Handy measurement solution are shown.

UNIVERSIDAD DE MÁLAGA

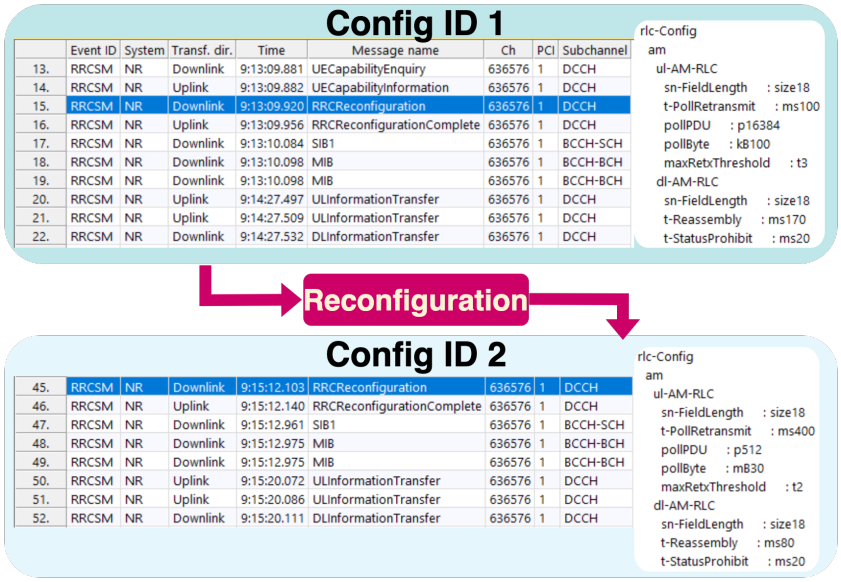


Figure 5.10: RRC message sent to the UE after a network reconfiguration

5.2 Evaluation

Once the details of the AutomAdapt implementation (monitoring, learning, prediction, and reconfiguration modules) have been presented, an evaluation is necessary to know its performance. For this purpose, a series of experiments have been carried out that involve the complete operation of AutomAdapt and therefore of its modules on the TSN over 5G testbed presented in Section 3.1. Each experiment is defined by a specific scenario, traffic pattern, and network configuration used, and the details of each of these elements are presented below. Note that the objective of the evaluation is to verify the correct functioning of AutomAdapt focused on TSN traffic over the 5G network. Subsequently, the setup should allow traffic to be sent that matches the type of TSN traffic over the 5G network between both TSN endpoints, while the 5G network configuration can be adapted to meet the expected requirements.

- *Evaluation scenario.* The scenario includes the hardware and software shown in Figure 3.1. Since there are two options for the 5G network (Nokia RAN + Open5GS or Keysight UXM 5G), it is possible to choose any of them for the experiment. Note that the network configuration parameters that can be modified may vary depending on the option chosen.
- *Traffic pattern.* TSN endpoints can insert various types of traffic according to the industrial traffic profiles presented in Table 1.2. Some of the examples of traffic patterns used for the evaluation are shown in Table 5.3.
- *5G network configuration.* The 5G network (5GC and RAN) supports the configuration of hundreds of parameters that can be relevant to meet certain traffic

UNIVERSIDAD DE MÁLAGA

Traffic pattern	*Traffic type (periodic)*	*Typical period*	*Typical data size (bytes)*
1	Cyclic-Asynchronous	2 ms	100
2	Cyclic-Asynchronous	10 ms	500
3	Cyclic-Asynchronous	10 ms	1000
4	Cyclic-Asynchronous	20 ms	1000
5	Network control	200 ms	100
6	Network control	1 s	500

Table 5.3: Evaluation traffic pattern examples

requirements, which are described in Section 3.3. The possible combinations of the available values of these parameters lead to an unmanageable number of different configurations. Consequently, the evaluation process is focused on the configuration of a reduced set of parameters that may be crucial for TSN traffic requirements in 5G networks. In particular, we selected the parameters of the RLC layer (AM mode) as discussed in Section 3.4.

RLC AM parameter	*Example values*
$MaxRetx_{Threshold}$	t1, t2, t8, t32
$Poll_{Byte}$	kB1, kB6000, mB8, mB40
$Poll_{PDU}$	p4, p1024, p12288, p65536
$t_{PollRetr}$	ms5, ms250, ms1000, ms4000
$t_{StatusProhibit}$	ms0, ms170, ms350, ms2400
$t_{Reassembly}$	ms0, ms65, ms160, ms200

Table 5.4: 5G radio network configuration (RLC AM) examples

In this case, to demonstrate the AutomAdapt behavior in a realistic environment, we use a commercial network scenario: Nokia RAN and Open5GS core, enabling several traffic patterns while several parameters of the 5G network can be modified. Specifically, Table 5.4 presents some example values of the parameters selected corresponding to the Acknowledged Mode (AM) mode of the Radio Link Control (RLC) layer. This scenario can be replicated using the components of the testbed shown in Figure 3.1. Furthermore, AutomAdapt modules and the instructions needed to reproduce the evaluation process are available in the AutomAdapt repository [225].

Once the specific evaluation aspects are identified, to use AutomAdapt, it is necessary to define the function *qty* by indicating the correspondences of the observed KPI ranges with individual values. The *qty* used in the experiments are based on the examples provided in Table 5.2, which defines distinct intervals for the observed delay and jitter. Furthermore, the function *desirable* must be established, which denotes the expected observed KPIs during the TSN session. As introduced in Section 5.1, the learning phase of

UNIVERSIDAD DE MÁLAGA

the AutomAdapt operation cycle involves an initial learning phase to create an automaton that can be used for online learning, as well as for prediction and reconfiguration phases. Given that the *Predict* algorithm relies on the states previously learned by the automaton and the network reconfiguration occurs after a prediction, it is crucial to have an initial phase dedicated to learn a reasonable number of network states. Subsequently, the details of the learning module as well as of these phases during the evaluation process are presented below.

5.2.1 Evaluation of the *Learn* algorithm

Since the learning module, together with the *Learn* algorithm, is one of the main components of AutomAdapt, an analysis is presented prior to the evaluation of the tool operation phases. However, before starting the analysis, it is worth highlighting the two key points of the algorithm (see Section 4.3.3) with respect to performance and complexity. First, the function $get(conf, qty(\bar{o}))$ (e.g. line 13), which retrieves a state $q \in Q$ such that $f_{\mathcal{A}}(q) = (conf, qty(\bar{o}))$. In this case, performance and complexity are directly related to the size of the automaton, as the search carries a higher computational cost. This is not a critical aspect considering that the size of the automata is finite and not too large, as well as the efficiency in data manipulation offered by the structure used to store the information [247]. Second, if the label is of the form $\mathbf{rd}(t)$ and $(q, q') \mapsto \mathbf{rd}(t')$ is the current pair in E, it is replaced by $(q, q') \mapsto \mathbf{rd}(min(t, t'))$, where $min(t, t')$ is the minimum value between t and t'. Therefore, the value of the variable remains the same or is updated with a lower value (until it reaches 0).

Based on the different automata presented in Section 5.1.2 associated with different cycle times, that is, automata with different levels of detail, Figure 5.11 depicts the trade-off between the build time of automata and its level of detail, demonstrating that a longer cycle time can have a significant impact on the reduction of the creation time due to the reduced number of operations to perform.

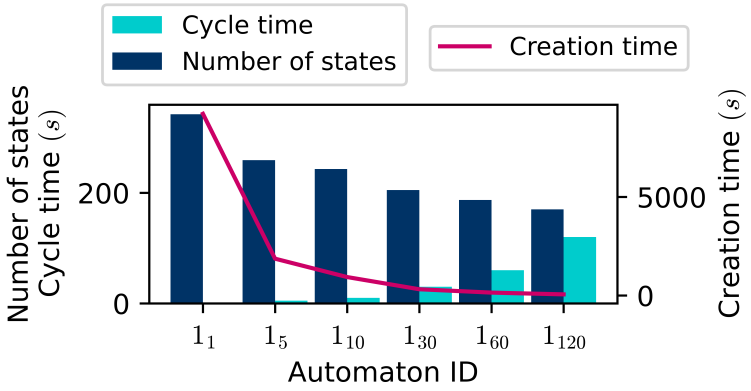
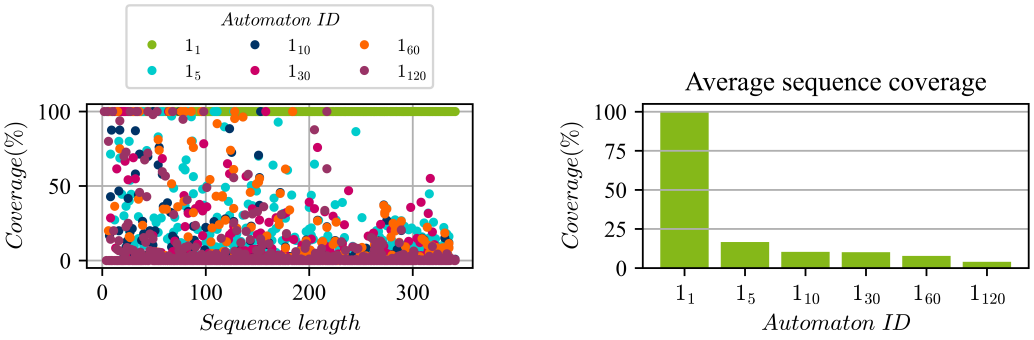


Figure 5.11: Evolution of the number of states and creation time of automata created with different cycle time values.

UNIVERSIDAD DE MÁLAGA

The information related to the sequence coverage of the automata is shown below. This type of coverage evaluates how many sequences from the network traces or the expected set of behaviors are accepted by the automaton, where a high sequence coverage means that the automaton can recognize and correctly process most or all of the input sequences, indicating a good match with the real-world behaviors. For this purpose, possible sequences with a variable length are generated from Automaton ID 1. Subsequently, these sequences are analyzed to check if they can be accepted by the automaton. Firstly, it is tested in a restrictive way, that is, if the automaton is able to accept the sequence that matches both the network states and the transitions (independently of their type). On the one hand, Figure 5.12a shows the percentage of sequence coverage based on sequence length, in which it is found that automata with a higher level of detail are able to accept longer traces, while automata with a lower level of detail are only able to accept shorter traces. On the other hand, Figure 5.12b shows the average sequence coverage of all generated traces, showing that Automaton ID 1 is able to accept all traces (as expected), while the percentage of acceptance is reduced, as the automata have a lower level of detail. Secondly, the sequences are checked in a less restrictive way, that is, checking only transitions due to a qty change, as it is assumed that a network configuration change can be made at any time and this transition could be accepted by the automaton. Similarly, Figures 5.13a and5.13b show the results, where it is demonstrated that the percentage of sequences that can be accepted by automata with lower level of detail increases slightly.

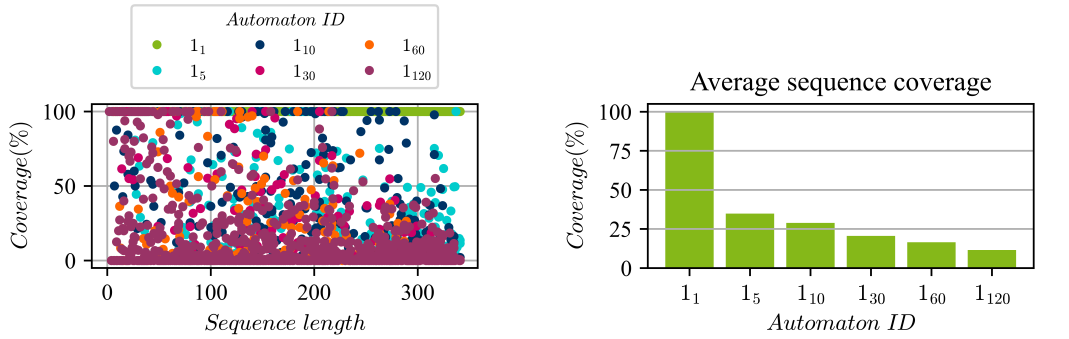


(a) Sequence coverage vs sequence length **(b)** Average sequence coverage of each automaton

Figure 5.12: Sequence coverage - restrictive

In addition, a heatmap showing the percentage of sequences accepted by each automaton as a function of the number of sequences generated in the restrictive mode is shown in Figure 5.14a. percentage of sequences accepted by each automaton based on the number of sequences generated. In general, the results obtained are similar regardless of the number of sequences generated. As mentioned above, the number of sequences accepted by the automaton decreases as the automaton has a lower level of detail. There is also a slight tendency for automata with a lower level of detail (e.g. Automaton ID 4 and 5) to accept fewer sequences as the number of generated traces increases, this may be due to the higher number of cases explored by the different sequences. In the same way, this is

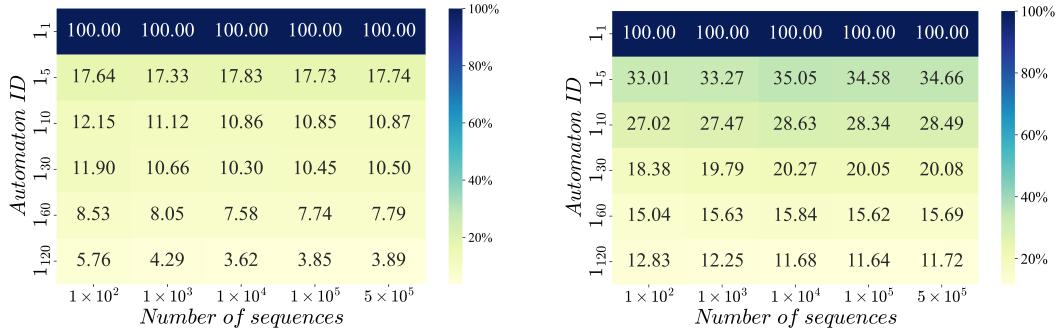
UNIVERSIDAD DE MÁLAGA



(a) Sequence coverage vs sequence length **(b)** Average sequence coverage of each automaton

Figure 5.13: Sequence coverage - less restrictive

true when sequences are tested in the less restrictive mode shown in Figure 5.14b.

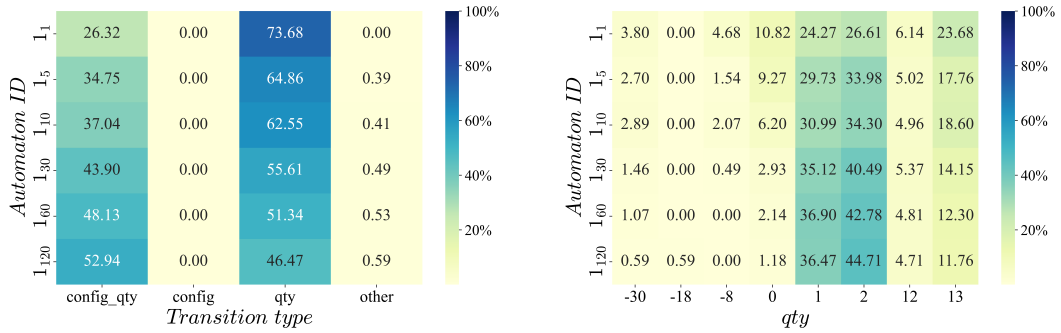


(a) Restrictive **(b)** Less restrictive

Figure 5.14: Sequence coverage heatmap: percentage of sequences accepted by each automaton

Finally, a heat map representing the coverage associated with the network states is displayed. First, Figure 5.15a shows the coverage associated with the possible types of transitions of the different automata, where it is clearly shown how the automata with a higher level of detail have a higher number of transitions due to a change in quality (qty). Note that in this case, all the transitions associated with a change of network configuration have a change in quality associated with them, which is crucial to take into account when configuring parameters such as stability time. Second, Figure 5.15b presents the coverage associated with the different values of qty for the different automata. In this case, automata with a lower level of detail obtain a higher percentage of qty values associated with better quality, confirming that a high cycle time masks a certain number of deviations in quality due to the averaging of the KPI values.

UNIVERSIDAD DE MÁLAGA



(a) Transition type coverage **(b)** *qty* values coverage

Figure 5.15: Transition type and *qty* values coverage heatmap

5.2.2 Evaluation of the AutomAdapt Initial Learning Phase

In this phase, AutomAdapt monitors the 5G network to obtain network traces that are used to generate automata. Specifically, the observed KPIs are obtained from a traffic pattern consisting of sending a 1000 bytes payload packet with a period of 10ms corresponding to traffic pattern 3 in Table 5.3. At the same time, the values of the 5G network parameters selected for evaluation (see Table 5.4) are randomly modified every 10-15 minutes. This results in the creation of an automaton (TOA) by the algorithm Learn with a number of states depending on the number of configurations tested and the definition of the function *qty* according to the definition of the algorithm.

Note that the learned automata will be used in the following phases for prediction and reconfiguration. Therefore, to make a more complete evaluation and check how the size of the automaton (i.e. the number of states) affects the performance of AutomAdapt, different time intervals starting at the same time reference will be used for the initial learning phase. Table 5.5 shows the information about such automata, including the time interval used to build the automaton, the states learned, and the number of configurations applied during that time interval. In particular, after the evaluation of the previous section, we decided on the automaton ID 1_{10} corresponding to a cycle time of 10 seconds (see Table 5.1), which presents values that we assume reasonable for this use case in terms of creation time and number of states. Additionally, we use a configuration goal time of 5 minutes, a reconfiguration number of 0, and a stability time of 3 minutes. As time progresses, the number of network states and configurations is expected to increase. Consequently, a greater number of states learned by the automaton means a greater amount of information that can be used in the following phases, potentially resulting in a higher number of predictions and reconfigurations that can be applied in the future. However, a high number of learned states can be counterproductive, as it leads to longer response times in processing the necessary information. In addition, the task of optimally sorting the list of suitable configurations obtained after a prediction is more complex.

UNIVERSIDAD DE MÁLAGA

Automaton ID	*Automaton build time*	*Number of states*	*Number of configurations*
1	24 h	234	90
2	48 h	473	181
3	72 h	724	273
4	96 h	986	407
5	120 h	1222	497

Table 5.5: Automata learned during initial learning phase

5.2.3 Evaluation of the Complete AutomAdapt Operation Cycle

Once the above-mentioned automata have been generated, AutomAdapt is prepared for prediction and reconfiguration using any of them as basis. Note that AutomAdapt continues to monitor the current state of the network and expand the base automaton (e.g. by adding new transitions or states when necessary) using the *Compose* function of the Learn algorithm. At this point, AutomAdapt can check if the current status does not meet the expected traffic requirements or predict possible deviations from the expected quality. Then, the algorithm Predict gets the network configurations that are suitable for meeting the expected quality (if they exist) utilizing the automaton chosen as the basis (i.e. the one learned during the initial learning phase).

In this case, the function *desirable*, that is, the target network state aims to keep the delay within the range of $[0, 7]$ms and the jitter within $[0, 1]$ms, corresponding to $qty = 1$ (refer to Table 5.2). Since we want to check the correct functioning of AutomAdapt, that is, that it can fulfill the objective of reconfiguring the 5G network to avoid deviations from the expected KPIs automatically, the baseline results are essential to provide a reference point for measuring the progress of improvement. Consequently, the network observed KPIs are monitored for a minimum of 12 hours using a fixed 5G network configuration and the previously chosen traffic pattern. Furthermore, to obtain solid results, four different 5G network configurations have been used, which are presented in Table 5.6 together with the percentage of time elapsed in the desired state ($qty = 1$), that is, the observed KPIs are in line with the expected ones during that interval.

Then AutomAdapt is run repeatedly with each automaton shown in Table 5.5 to obtain the results of the observed KPIs, making use of algorithm Predict for the predictions and implementing the suggested network reconfigurations.

Figure 5.16 displays a comparative analysis of the time elapsed in the desired network state, where the measured KPIs align with traffic requirements. This figure distinctly shows the percentage difference, ideally an enhancement (note that a value $<= 0$ as in the case of Automaton ID 1 versus Baseline configuration ID 4, indicates that AutomAdapt is not effective in terms of improving the elapsed time), in the time spent in the desired network state with AutomAdapt active and its implemented automatic network reconfigurations,

UNIVERSIDAD DE MÁLAGA

Baseline configuration	*ID 1*	*ID 2*	*ID 3*	*ID 4*
$MaxRetx_{Threshold}$	t6	t3	t32	t32
$Poll_{Byte}$	kB8	kB6500	kB8	mB40
$Poll_{PDU}$	p24576	p20480	p1024	p40960
$t_{PollRetr}$	ms500	ms140	ms70	ms40
$t_{StatusProhibit}$	ms800	ms90	ms100	ms40
$t_{Reassembly}$	ms170	ms45	ms10	ms180
Desire network state time (%)	41,99	40,81	45,02	47,22

Table 5.6: 5G network configurations (RLC AM) used to obtain baseline results for comparison

versus the time under a static network configuration with AutomAdapt inactive (i.e. baseline results). In general, the data indicate that the enhancements increase gradually (approximately 0-45%) when using an automaton as a basis for AutomAdapt with more states (see Table 5.5), which makes sense considering the longer initial learning period and the increased variety of network configurations. Thus, AutomAdapt collects more data, improving its ability to predict and adjust the network more effectively. However, as mentioned above, a large number of learned states can be counterproductive. This is the case of Automaton ID 5, which corresponds to the automaton with more learned states, although it shows a lower percentage of improvement than Automaton ID 4. Therefore, it is important to choose precisely the base automata for each specific traffic pattern. Furthermore, it should be noted that the additional parameters introduced above (configuration goal time, reconfiguration number, and stability time) and the way of ordering the suitable configurations suggested to reconfigure the 5G network can affect the final results.

Furthermore, since AutomAdapt continues to expand the automaton used as a basis, Table 5.7 details the results of these automata after the evaluation of the complete operation cycle. In this context, a greater number of new states learned means that the information available in the automaton used as a basis was not sufficient. AutomAdapt is also considered to operate efficiently if better results are achieved with fewer reconfigurations applied. On the one hand, the number of new states is similar, regardless of the number of states of the base automaton. Considering Automaton ID 1 to Automaton ID 4, the number of reconfigurations decreases, and the percentage improvement in terms of time elapsed in the desired state increases. However, with Automaton ID 5 the number of reconfigurations applied increases again, while the results obtained are worse than with Automaton ID 4. Consequently, in this case it can be assumed that the automaton ID 4 is the best choice for traffic pattern 3 (Table 5.3) in order to meet the requirements defined by $qty = 1$ (see Table 5.2).

UNIVERSIDAD DE MÁLAGA

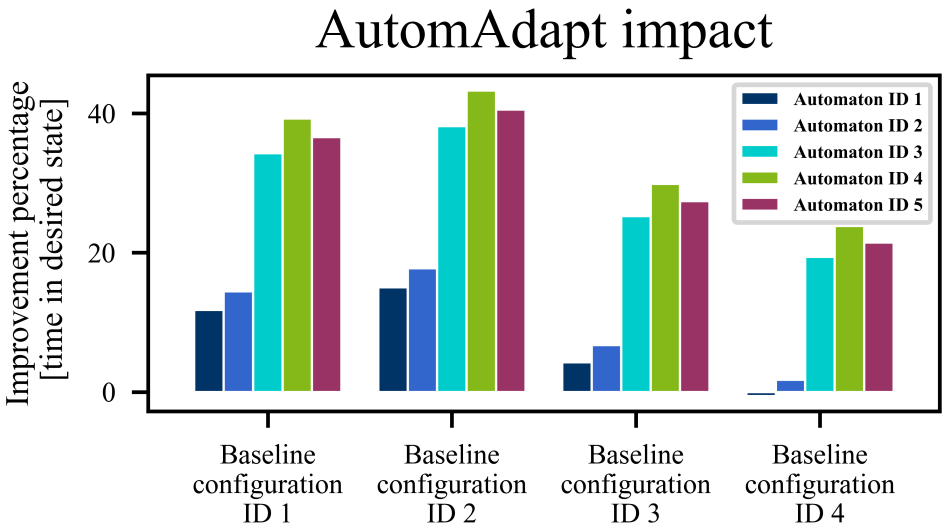


Figure 5.16: AutomAdapt impact: Elapsed time in desired network state

Automaton ID'	*Number of states*	*Number of new states*	*Number of reconfigurations*
1	252	18	28
2	494	21	15
3	747	23	17
4	1004	18	11
5	1243	21	15

Table 5.7: Resulting automata during prediction and reconfiguration phase

5.2.4 Example Trace

In order to show in a clear and visual way the operation cycle of AutomAdapt, an example network trace is presented below. In particular, this example is extracted from the evaluation and therefore uses $qty = 1$ (Table 5.2) and traffic pattern 3 (Table 5.3). Additionally, the specific values of the 5G network configuration parameters (RLC AM) used in this example are displayed in Table 5.8.

Following the structure of the evaluation, AutomAdapt starts monitoring and learning the behavior of the 5G network during a traffic session (initial learning phase). Figure 5.17 shows the evolution of the observed KPIs during this phase, where the delay and jitter results are represented on the left and right y axes, respectively. In this case, AutomAdapt creates an abstract network state $q1$, and learns that network configuration ID 1 satisfies the requirements of this traffic pattern for approximately 230 seconds. Subsequent deviations are also learned by the automaton, which will be crucial for future predictions and suggest a reconfiguration of the network. Thus, when AutomAdapt next observes

UNIVERSIDAD DE MÁLAGA

Configuration	*ID 1*	*ID 2*
$MaxRetx_{Threshold}$	t3	t3
$Poll_{Byte}$	kB75	kB10
$Poll_{PDU}$	p8	p8192
$t_{PollRetr}$	ms50	ms220
$t_{StatusProhibit}$	ms140	ms10
$t_{Reassembly}$	ms5	ms120

Table 5.8: 5G network configuration (RLC AM) examples

and identifies that the network has reverted to a previously learned network state, it can predict such deviations of the observed KPIs and apply the necessary reconfiguration to fulfill the requirements specified for the TSN session.

Once the automaton has enough information to predict and reconfigure, it can be used as a basis. An example of the complete operation cycle is represented in Figure 5.18, where the automaton detects that it is in the same previously learned network state $q1$ (that is, using traffic pattern 3 and the network configuration ID 1), where deviations are expected at about 230 seconds. Consequently, a reconfiguration must be applied in advance. The left-hand side shows a similar (as expected) evolution of the observed KPIs. On the right part, the evolution of such observed KPIs is presented, once AutomAdapt predicts the deviations and applies a learned configuration to continue meeting the requirements of the TSN traffic session. Additionally, AutomAdapt transitions to the network state $q2$, which corresponds to the network configuration ID 2. Putting it all together, AutomAdapt is able to predict deviations and apply a network configuration to avoid them automatically, thus improving the results obtained in terms of elapsed time in the desired state, i.e. meeting the traffic requirements.

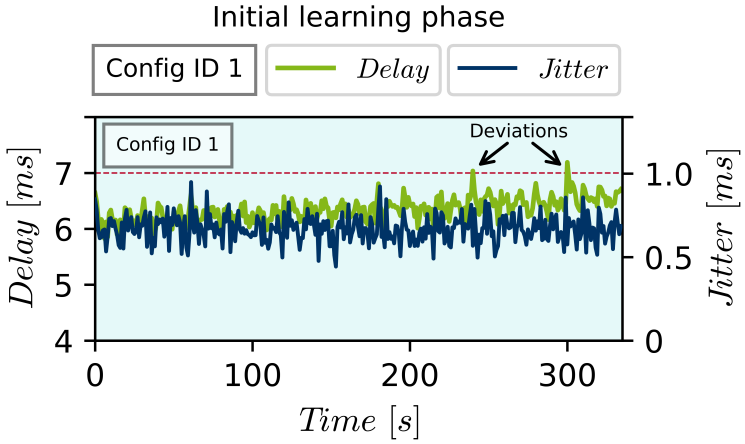


Figure 5.17: Example network trace: Observed KPIs during the initial learning phase

UNIVERSIDAD DE MÁLAGA

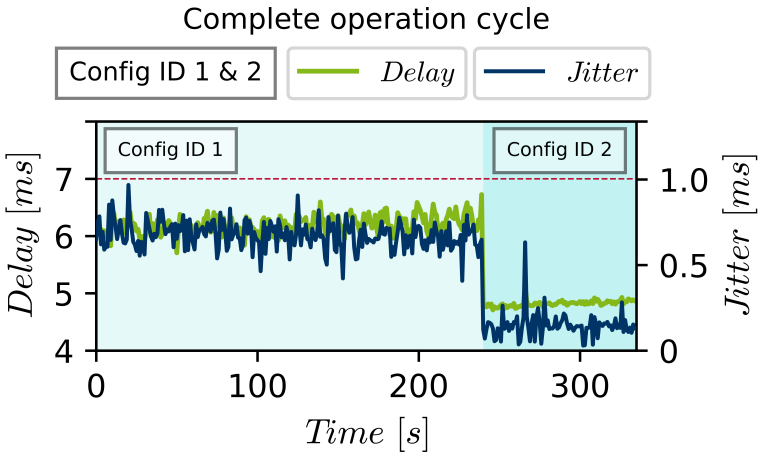


Figure 5.18: Example network trace: Observed KPIs during the complete operation cycle

Taking into account the two network configurations (see Table 5.8 used in this example, the following explains how different values of the parameters of the RLC layer (AM mode) affect the results of the observed KPIs obtained according to Table 3.2. First, increasing the $Poll_{PDU}$ value can decrease control signaling and reduce delays by permitting the transmission of more PDUs before requesting status updates from the UE. In addition, extending the duration of $t_{PollRetr}$ provides the UE with additional time to process and respond with status information, potentially minimizing unnecessary retransmissions and reducing delays. Furthermore, a higher value for the duration of $t_{Reassembly}$ allows complete arrival and reassembly of all PDUs from the RLC data from the upper layers, helping to avoid unnecessary retransmissions. In contrast, reducing the value of $t_{StatusProhibit}$ decreases the interval between PDUs but may increase control signaling. The value of $Poll_{Byte}$ affects in the same way as $Poll_{PDU}$, however, its reduction in this case is minimal and is not reflected. Finally, a large value of $MaxRetx_{Threshold}$ allows more retransmissions, increasing the probability of successful delivery, but it can increase the end-to-end delay and jitter if retransmissions occur frequently, which is not reflected since both configurations use the same value. Furthermore, the pair plot corresponding to the observed KPIs of the configurations used in the example network trace (Config ID 1 and Config ID 2) is presented in Figure 5.19, where the improvement produced by Config ID 2 in terms of delay and jitter is verified.

The entire AutomAdapt data set, which includes automata learned during the initial learning phase, applied network configurations, and reconfigurations performed during the operation cycle, is available at [225], covering all details and files of the evaluation process. In total, the data set consists of a total of **12071** disparate **learned network states** and **7211** different **network configurations**, corresponding to **2310 hours of testing**. Note that this number of states corresponds to a cycle time of 10 seconds; therefore, the total number of states may vary depending on the selected cycle time (more details are provided in Table 5.1), while the number of network configurations and testing

UNIVERSIDAD DE MÁLAGA

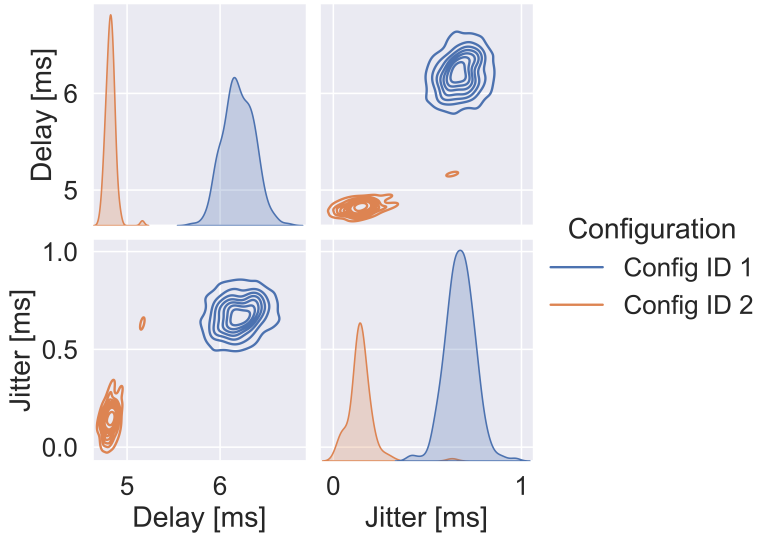


Figure 5.19: Example network trace: Relationship between the observed KPIs of the two configurations used.

hours remain the same. Additional related information related to transition and state coverage can be found in Tables 5.9 and 5.10, respectively. First, the number of transitions according to their type, where *config* corresponds to a transition by configuration change; *qty* to a transition by *qty* value change; *config+qty* a transition in which there is a change of both; and error for transitions not captured. Second, the count of the number of states associated with the values defined for the function *qty* during the evaluation. In addition, Figure 5.20 shows the relationship between the RLC AM parameters of the entire data set, where the main diagonal shows the kernel density estimate plots for each parameter separately, and the rest of the elements represent the relationship between each pair of parameters. This shows that a considerable part of the possible values of the chosen parameters have been covered.

5.2.5 Analysis of Results

To wrap things up, the results obtained are interpreted in order to draw some conclusions.

- The elapsed time in the desired network state (i.e. the observed KPIs are aligned with the traffic requirements of the TSN session) grows considerably from a certain number of network states learned by the automaton. This is evidenced by Automaton ID 1 and 2 (less than 500 states), with which the improvement achieved is low, even becoming negative in the case of Automaton ID 1 compared to the baseline configuration 4. In contrast, Automaton ID 3, 4 and 5 achieve an improvement ranging from 35-45%, significantly enhancing the observed KPIs during the traffic session. Figure 5.16 shows that there is a great improvement between Automaton ID 1 and 2 compared to Automaton ID 3, 4 and 5.

UNIVERSIDAD DE MÁLAGA

Transition type	*Count*
config + qty	7284
qty	4675
config	105
error	7

Table 5.9: Count of the type of transitions of the learned states of the data set

qty	*Count*
-30	100
-18	1
-8	447
-1	524
1	5964
2	2672
12	659
13	1704

Table 5.10: Number of states learned for the *qty* function definition for evaluation

- Due to the fact that AutomAdapt continuously acquires knowledge, not only during the initial learning phase, new states and previously unlearned transitions can be added at any time. Once incorporated, these states can become relevant to the automaton, aiding in future predictions. Consistently, a similar number of new states are added through different experiments carried out in the evaluation (see Table 5.7). In general, this is due to the time it may take for the network to stabilize after a configuration change, since according to Table 5.9 which shows the total number of each type of transition, a large part of the configuration change transitions have associated qty change also due to a brief destabilization. The *stability time* parameter is used to not apply any reconfiguration for the defined time, i.e. the maximum time allowed for the network to be considered stable after reconfiguration. Therefore, choosing a precise value can help to reduce the number of configurations while the time in the desired state is equal or longer. However, a number that is too low or too high may result in the network failing to stabilize or in the time in the desired state decreasing if reconfiguration does not produce the expected results.
- In general, it is logical to assume that as AutomAdapt learns more network states (more information), the efficiency of its operation is improved. This assumes that the network meets traffic requirements for longer, ideally with a lower frequency of necessary reconfigurations. However, it has been found that this is not always the case (e.g. using Automaton ID 5 in the evaluation), as more information has to be processed. For example, if there are more suitable configurations suggested by the algorithm after a prediction, the task of ordering them conveniently for the specific traffic type can influence the results significantly.
- Similarly, the information learned by the automaton can vary using different values of the cycle time, i.e. the time used to average the results of the observed KPIs. Choosing an appropriate value is important in order not to lose useful information

UNIVERSIDAD DE MÁLAGA

(e.g. deviations that can be foreseen in the future). Moreover, in the case of learning from network traces, the automaton construction time can increase or decrease greatly depending on this value as shown in Figure 5.11. On the other hand, it is important to take into account the results based on the definition of the *qty* function, since most of the network states should have a *qty* value associated with them as defined by the network operator. For example, the definition of the function *qty* (see Table 5.2) can be considered sufficient to capture the majority of states (>99%) according to Table 5.10 which shows the number of states learned based on the *qty* function definition.

- It is vital to establish a baseline of observed KPIs to evaluate the performance of AutomAdapt. As seen, the results obtained from this comparison can vary by approximately 15-20% depending on the baseline values used. Therefore, values should be chosen that cover the different possible conditions.
- It is considered necessary (at least) about 7 full days of testing to properly evaluate the performance of AutomAdapt with a traffic profile, of which 4-5 days are used for initial learning and obtaining enough information that can be used for predictions and their respective reconfigurations. At least 2 days are needed to evaluate the complete AutomAdapt operation cycle, in which different values of the parameters (cycle time, stability time, configuration goal time, etc.) are tested to try to obtain the best combination.

UNIVERSIDAD DE MÁLAGA

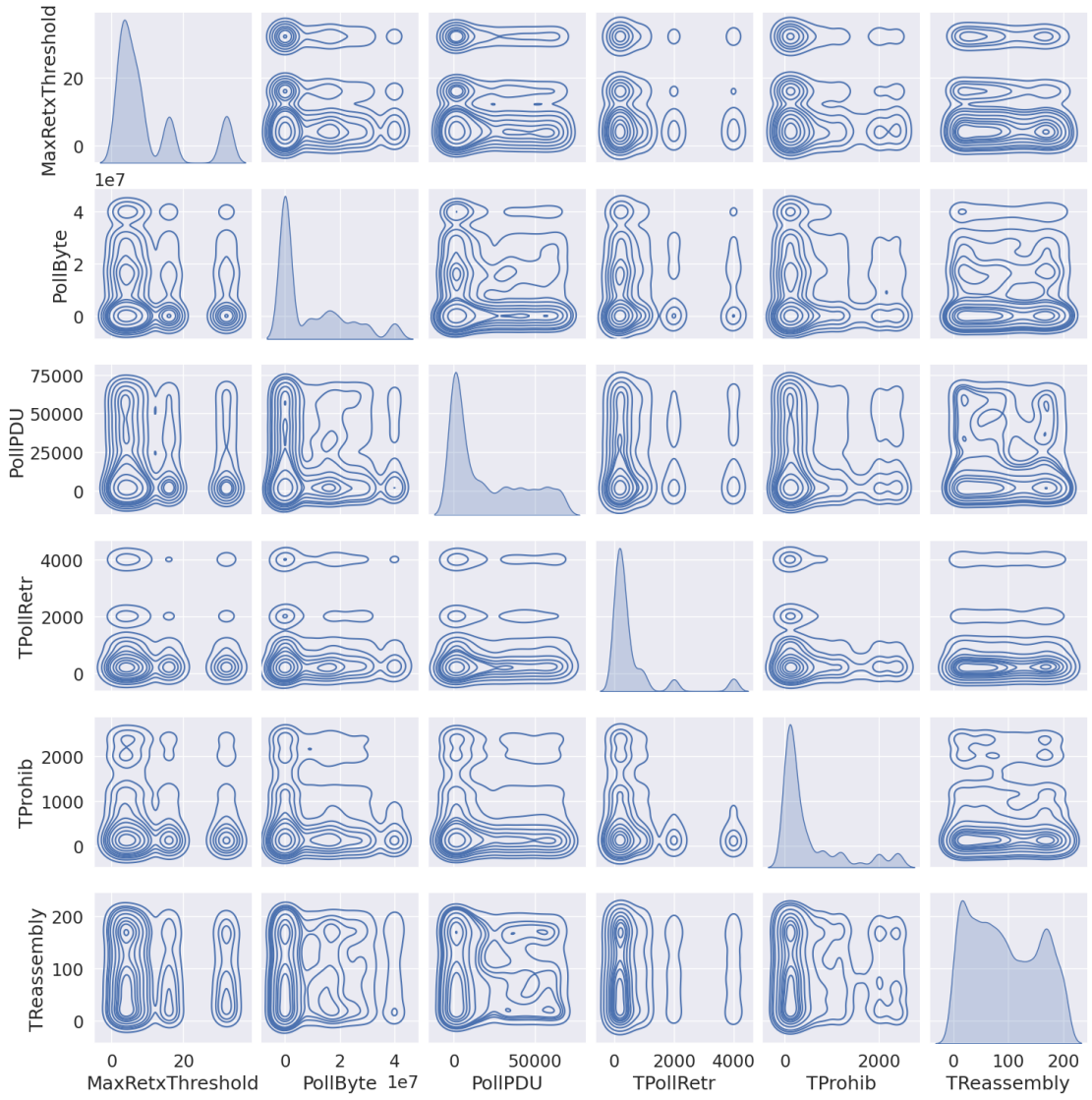


Figure 5.20: Relationship between RLC AM 5G RLC network parameters

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Chapter 6

Conclusions and Future Work

Contents

6.1 Conclusions 122
6.2 Future Work 124
6.3 Publications and Projects 125
6.3.1 Journals 125
6.3.2 Other Activities 125
6.3.3 Related Projects and Funding 126
6.3.4 Research Stay 126

Summary: This chapter offers a summary of the key findings of this thesis and explores possible directions for further investigation. It also lists the publications and projects associated with this thesis.

UNIVERSIDAD DE MÁLAGA

6.1 Conclusions

This thesis focuses on Zero-Touch reconfiguration of TSN over 5G network to meet the demanding requirements of Industry 4.0, namely for time-sensitive applications such as remote control and industrial automation. In particular, the work addresses the potential of the 5G network capabilities to support different TSN traffic profiles. In this context, the IEEE Time-Sensitive Networking (TSN) standards aim to provide deterministic communications over Ethernet networks. In addition, as Industry 4.0 use cases emerge with requirements such as mobility, wireless technologies are gaining importance, where 5G emerges as a prime candidate for integration with TSN. In this sense, 3GPP is driving standardization, defining the necessary components and procedures. However, interoperability requires solving certain challenges, such as the development of TSN translators at the network endpoints to enable interconnection between both domains, or accurate time synchronization through the system. Even more importantly, the TSN Application Function ensures the quality of the end-to-end traffic session, exchanging the necessary information to configure the network based on the requirements demanded by the traffic. Nonetheless, configuring the 5G network to maintain expected quality is not trivial considering the variability of Key Performance Indicators (KPIs) such as latency or jitter during a TSN traffic session. Therefore, methods and techniques are needed to ensure the correct operation of the system automatically and with minimum human intervention due to its complexity. For this purpose, a TSN over 5G testbed developed as part of this thesis is presented, including the prototyping of the fundamental entities mentioned above to enable the interoperability between both domains, which provides a more complete understanding of the feasibility of integrating these technologies in a real environment under various conditions. This answers Research Question 1 (RQ1) *"Does existing equipment support the integration of Time-Sensitive Networking into the 5G system?"*, asserting that such integration is possible, taking into account that it will be compliant with the 3GPP definition once the necessary entities are final versions rather than prototypes.

The use of model learning (or automata learning) is proposed to address the objective of configuring the 5G network to maintain the TSN traffic requirements. Specifically, it demonstrates the ability to automatically build 5G network behavior models from the results obtained in terms of the traffic KPIs being sent corresponding to different TSN traffic patterns, while using a variety of 5G network configurations. To this effect, the *Learn* algorithm is designed to be responsible for the construction of the automaton composed of abstract network states that represent the evolution of its behavior in relation to time, including traffic KPIs and the configuration of the 5G network, which is possible due to the definition of the type of automaton called Traffic Oriented input/output Automata (TOA) explained in detail in Chapter 4. Furthermore, the algorithm *Predict* is designed to exploit the previous learned automata to make predictions (e.g. possible deviation from KPIs) based on the states of the network. Therefore, the algorithm is able to suggest feasible 5G network configurations to be applied systematically and continue to maintain the expected quality. A tool that integrates the aforementioned

UNIVERSIDAD DE MÁLAGA

algorithms to automatically reconfigure the 5G network, called AutomAdapt [225] is introduced. It is divided into four main modules: monitoring, learning, prediction, and reconfiguration. This tool is evaluated on the TSN over 5G testbed at the University of Malaga, where a set of experiments have been carried out to analyze and evaluate the operational effectiveness of these modules working in a closed loop. Therefore, it is established that the use of model learning together with a Zero-Touch approach answers the Research Question 2 (RQ2): *"What automated methods can be used to configure the 5G network to meet the requirements of TSN traffic?"*. Moreover, it is possible to be unified in one tool.

The results obtained from the evaluation provide valuable information on the effectiveness and performance of AutomAdapt, demonstrating that the 5G network is capable of meeting the requirements of a substantial part of industrial traffic patterns defined by the IEEE standard and achieving improvements in terms of the time in which the network meets the traffic requirements of a TSN session in a real environment. In particular, when the automaton has a certain number of states (e.g. more than 500 states), improvement values of up to 45% are achieved. However, a greater number of states does not always translate into improved operational efficiency, as the additional aspects and parameters that can be configured (e.g., stability time or the ordering method of the configurations suggested by the algorithm) have a direct impact on the results. Therefore, the correct choice of the values of these parameters, together with the cycle time chosen to average the KPI results, is crucial to the final results. In addition, these experiments have resulted in a catalog that contains a large number of network behavior models, including feasible 5G network configurations for a set of TSN traffic profiles. Note that evaluation on a testbed with commercial equipment allows more representative and reliable data to be obtained in a real environment, reducing the possibility of errors associated with simulation models and increasing the usefulness of the results obtained. Furthermore, this experience not only improves understanding of the system, but also provides key findings for implementation and development. For example, understanding how the values of different 5G network parameters affect the KPIs associated with certain TSN traffic profiles. Moreover, it is important to highlight the importance of using techniques such as model learning, which allow interpreting the correspondence between the learned model and the observed behavior due to explainability. This enables the production of more explainable models, which is useful due to the nature of the data, for example, allowing to know how the network behaves with respect to time with certain buffers or timers when they change their value. Subsequently, Research Question 3 (RQ3): *"How can the explainability of the system be enhanced to ensure a clearer understanding of its functioning?"* is addressed.

All things considered, the work developed in this thesis exhibits a contribution to the feasibility toward Zero-Touch configuration to meet the TSN traffic requirements over 5G networks by providing a method that combines Zero-Touch and model learning approaches to achieve this objective. Furthermore, due to the nature of the algorithms designed and the approach considered in this thesis, the work could be extrapolated to

UNIVERSIDAD DE MÁLAGA

other types of applications and environments, even including other technologies, with the same objective of improving operating efficiency, while maintaining explainability as one of the main characteristics to justify and interpret decision making. Nevertheless, it should be noted that in complex networks such as TSN over 5G it is not yet possible to achieve fully autonomous reconfiguration without human intervention in all possible conditions, especially in those with more demanding traffic requirements. Accordingly, the exploitation of all available information on the network, as well as the use of complementary methods and techniques that can not only support the decisions taken, but also obtain the necessary information in situations where the automata do not have sufficient information, is essential to achieve this.

6.2 Future Work

With regard to future work, there exist several open research topics and challenges that require further investigation to be addressed, with the aim of enhancing the results obtained, extending the scope of application, and improving the methods and techniques employed.

Firstly, the work will be extended to incorporate additional layers of the 5G protocol stack, such as MAC and PDCP, that can help enhance the results obtained. Secondly, the algorithms designed in this thesis allow for the integration of other relevant parameters (e.g. TSN) into the automaton to provide more comprehensive insights. In addition, the use of multiple PDU sessions or QoS flows within a PDU session could contribute to improve performance, as it is possible to associate them with different DRBs (see Figure 3.10) and therefore configure the parameters of the different layers separately [171]. This can also facilitate comparative analysis to have a better understanding of the network behavior. Furthermore, the upgrade of the testbed developed at the University of Malaga as part of this thesis, to enable the procedures defined in Release 16 [28], including the support of the Ethernet PDU session for complete integration of TSN over 5G. For example, the development of advanced translation techniques including dynamic QoS mapping between both domains [98], [99] and native support for time synchronization over the 5G network [108].

On the other hand, Machine Learning techniques [199], [200] will be employed to assist model learning. In particular, in the case where the algorithm *Predict* reports a *Warning* message, indicating that has an undesirable quality level and it is not possible to suggest any reconfiguration of the network, these techniques can be used to train a model with the available information to generate new network configurations (which are not yet included in the automaton) capable of meeting traffic requirements. Additionally, when *nextState* is not recognized by the automaton and the *Unknown* message is reported, these methods can estimate the results of the observed KPIs in order to include this unknown state in the automaton. Consequently, this will ensure a more robust system capable of operating in a wider range of situations. However, the use of new configurations that have not been previously tested or the inclusion of unknown states in the automaton can lead

UNIVERSIDAD DE MÁLAGA

to instability or errors in the system. Moreover, most of the time it is not feasible to evaluate such situations in a real network. In order to address this challenge, a digital twin of the 5G network will be developed. This digital twin will support the evaluation of these situations when the real network is not available. It can also help make predictions, not only of possible deviations from the observed KPIs, but also to know the state of the network after a reconfiguration, which can help to decide which network configuration to apply when there are several suggested by the *Predict* algorithm. Finally, the inclusion of the digital twin in the closed loop to move toward an active learning approach [205] and XAI to capture the causes of potential violations [206] (e.g. observed KPIs or network configuration values) can improve the overall efficiency and responsiveness of network management procedures.

Eventually, AutomAdapt evaluation can be extended to include the impact of additional existing parameters such as configuration goal time, reconfiguration number, and stability time to optimize these values based on current network needs and traffic requirements. Also, since determining the optimal configuration from the list is beyond the scope of the algorithm, having a list of available options can be significant in the results. In addition, the use of tools to perform run-time verification [248] against certain defined properties can positively contribute to these results. Putting it all together can have a positive impact on the operational efficiency of AutomAdapt, translating into the possibility of improving current results or even adding new traffic patterns that are more demanding in terms of requirements.

6.3 Publications and Projects

6.3.1 Journals

- **J1**. F. Luque-Schempp, L. Panizo, M. -d. -M. Gallardo, P. Merino and J. Rivas, "Toward Zero Touch Configuration of 5G Non-Public Networks for Time Sensitive Networking" in IEEE Network, vol. 36, no. 2, pp. 50-56, March/April 2022, doi: 10.1109/MNET.006.2100442 [34]
- **J2**. F. Luque-Schempp, L. Panizo, M. -D. -M. Gallardo and P. Merino, "AutomAdapt: Zero Touch Configuration of 5G QoS Flows Extended for Time-Sensitive Networking" in IEEE Access, vol. 11, pp. 82960-82977, 2023, doi: 10.1109/ACCESS.2023.3302264 [35]

6.3.2 Other Activities

- **A1. Publication under review**. L. Panizo, M. -d. -M. Gallardo, F. Luque-Schempp and P. Merino, "Runtime monitoring of 5G network slicing using STAn" (submitted, under review).
- **A2. Oral presentation**. F. Luque-Schempp, L. Panizo, M. -d. -M. Gallardo, P. Merino and J. Rivas, "Toward Zero Touch Configuration of 5G Non-Public

UNIVERSIDAD DE MÁLAGA

Networks for Time Sensitive Networking" in Proceedings of the XXVI Jornadas de la Sociedad de Ingeniería de Software y Tecnologías de Desarrollo de Software (SISTEDES '22), Santiago de Compostela, Spain, Sept. 2022.

- **A3. Oral presentation**. F. Luque-Schempp, D. Rico, "Deterministics Communications over 5G - Validation with Victoria Network at UMA", in the Workshop IoT platforms for indoor air quality study, Universite Sorbonne Paris Nord (USPN), Villetaneuse, France, May 2023.
- **A4. Oral presentation**. F. Luque-Schempp, L. Panizo, M. -d. -M. Gallardo and P. Merino, "AutomAdapt: Zero Touch Configuration of 5G QoS Flows Extended for Time-Sensitive Networking," in Proceeding of the Jornadas de Concurrencia y Sistemas Distribuidos (JCSD 2024), Cádiz, Spain, Jun. 2024.

6.3.3 Related Projects and Funding

- **P1: FPI**, Ministry of Science, Innovation and Universities of Spain - grant agreement PRE2019-089212.
- **P2: RFOG**, Ministry of Science, Innovation and Universities of Spain - grant agreement RTI2018-099777-B-I00.
- **P3: 5GENESIS**, European Union's Horizon 2020 research and innovation programme - grant agreement 815178.
- **P4: Affordable5G**, European Union's Horizon 2020 research and innovation - grant agreement 957317.
- **P5: EVOLVED5G**, European Union's Horizon 2020 research and innovation - grant agreement 101016608.
- **P6: 5G+TACTILE-1**, Ministry of Economic Affairs and Digital Transformation of Spain, UNICO I+D - grant agreement TSI-063000-2021-11.
- **P7: 6G-SANDBOX**, European Union's Horizon Smart Networks - grant agreement 101096328.
- **P8: FIDAL**, European Union's Horizon Smart Networks - grant agreement 101096146.

6.3.4 Research Stay

During the thesis work, a three-month internship was held in 2022 at Institute Industrial IT - inIT at Technische Hochschule Ostwestfalen Lippe (Germany), which has been made possible thanks to the financial support of the Ministry of Science, Innovation and Universities of Spain via grant PRE2019-089212.

UNIVERSIDAD DE MÁLAGA

Bibliography

[1] M. Shaikh, P. Shah, and R. Sekhar, "Communication protocols in industry 4.0," Cited by: 0, 2023, pp. 709–714. DOI: 10.1109/ICSEIET58677.2023.10303397.

[2] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta, "Time-sensitive networking in IEEE 802.11be: On the way to low-latency wifi 7," *Sensors*, vol. 21, no. 15, 2021. DOI: 10.3390/s21154954.

[3] 5G-ACIA, *White Paper: Integration of 5G with Time-sensitive networking for industrial communications*, [Online] Available at: https://5g-acia.org/whitepapers/integration-of-5g-with-time-sensitive-networking-for-industrial-communications/, (accessed on 13.01.2024), 2021.

[4] N. Finn, P. Thubert, B. Varga, and J. Farkas, *Deterministic Networking Architecture*, RFC 8655, (accessed on 07.03.2024), Oct. 2019. DOI: 10.17487/RFC8655.

[5] E. Grossman, *Deterministic Networking Use Cases*, RFC 8578, (accessed on 07.03.2024), May 2019. DOI: 10.17487/RFC8578.

[6] International Organization for Standardization, *Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model*, [Online] Available at: https://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip, (accessed on 13.01.2024), 1994.

[7] N. Finn, *Introduction to time-sensitive networking*, eng, 2022.

[8] J. L. Messenger, *Time-sensitive networking: An introduction*, eng, 2018.

[9] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks," *IEEE Std 802.1Q-2022 (Revision of IEEE Std 802.1Q-2018)*, pp. 1–2163, 2022. DOI: 10.1109/IEEESTD.2022.10004498.

[10] "IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications," *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pp. 1–421, 2020. DOI: 10.1109/IEEESTD.2020.9121845.

[11] "IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams," *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)*, pp. C1–72, 2010. DOI: 10.1109/IEEESTD.2009.5375704.

[12] "ISO/IEC/IEEE International Standard – Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 1Q: Bridges and bridged networks

UNIVERSIDAD DE MÁLAGA

AMENDMENT 3: Enhancements for scheduled traffic," *ISO/IEC/IEEE 8802-1Q:2016/Amd.3:2017(E)*, pp. 1–62, 2018. DOI: 10.1109/IEEESTD.2018.8290654.

[13] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks - Amendment 34: Asynchronous Traffic Shaping," *IEEE Std 802.1Qcr-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, IEEE Std 802.1Qcy-2019, and IEEE Std 802.1Qcx-2020)*, pp. 1–151, 2020. DOI: 10.1109/IEEESTD.2020.9253013.

[14] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 29: Cyclic Queuing and Forwarding," *IEEE 802.1Qch-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd(TM)-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, IEEE Std 802.1Qbz-2016, and IEEE Std 802.1Qci-2017)*, pp. 1–30, 2017. DOI: 10.1109/IEEESTD.2017.7961303.

[15] "IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks – Amendment 26: Frame Preemption," *IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014)*, pp. 1–52, 2016. DOI: 10.1109/IEEESTD.2016.7553415.

[16] "IEEE Standard for Local and metropolitan area networks–Virtual Bridged Local Area Networks–Bridge Port Extension," *IEEE Std 802.1BR-2012*, pp. 1–135, 2012. DOI: 10.1109/IEEESTD.2012.6239543.

[17] "IEEE Standard for Local and metropolitan area networks–Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks–Amendment 17: Priority-based Flow Control," *IEEE Std 802.1Qbb-2011 (Amendment to IEEE Std 802.1Q-2011 as amended by IEEE Std 802.1Qbe-2011 and IEEE Std 802.1Qbc-2011)*, pp. 1–40, 2011. DOI: 10.1109/IEEESTD.2011.6032693.

[18] "IEEE Standard for Local and metropolitan area networks— Bridges and Bridged Networks - Amendment 24: Path Control and Reservation," *IEEE Std 802.1Qca-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qcd-2015 and IEEE Std 802.1Q-2014/Cor 1-2015)*, pp. 1–120, 2016. DOI: 10.1109/IEEESTD.2016.7434544.

[19] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing," *IEEE Std 802.1Qci-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, and IEEE Std 802.1Qbz-2016)*, pp. 1–65, 2017. DOI: 10.1109/IEEESTD.2017.8064221.

[20] "IEEE Standard for Local and Metropolitan Area Networks–Link-local Registration Protocol," *IEEE Std 802.1CS-2020*, pp. 1–151, 2021. DOI: 10.1109/IEEESTD.2021.9416320.

[21] "IEEE Standard for Local and metropolitan area networks–Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)," *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)*, pp. 1–119, 2010. DOI: 10.1109/IEEESTD.2010.5594972.

UNIVERSIDAD DE MÁLAGA

[22] 3rd Generation Partnership Project (3GPP); Technical Specification Group Services and System Aspects; Service requirements for cyber-physical control applications in vertical domains; Stage 1, Technical Specification (TS) 22.104, v19.1.0 Sept. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/22_series/22.104/22104-j10.zip, (accessed on 07.03.2024).

[23] IEC/IEEE 60802 TSN Profile for Industrial Automation, draft v2.1, Sept. 2023, [Online]. Available: https://www.ieee802.org/1/files/private/60802-drafts/d2/60802-d2-1.pdf, (accessed on 07.03.2024).

[24] Time Sensitive Networks for Flexible Manufacturing Testbed – Characterization and Mapping of Converged Traffic Types", v1.0, Mar. 2019, [Online]. Available: https://www.iiconsortium.org/pdf/IIC_TSN_Testbed_Char_Mapping_of_Converged_Traffic_Types_Whitepaper_20180328.pdf, (accessed on 07.03.2024).

[25] "IEEE Standard for Ethernet," *IEEE Std 802.3-2018 (Revision of IEEE Std 802.3-2015)*, pp. 1–5600, 2018. DOI: 10.1109/IEEESTD.2018.8457469.

[26] 3rd Generation Partnership Project (3GPP), 3GPP System architecture for the 5G system (5GS),Technical Specification (TS) 23.501, v18.3.0 Sept. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-i30.zip, (accessed on 13.02.2024).

[27] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5g network slicing using sdn and nfv: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167, 2020, Cited by: 448; All Open Access, Green Open Access, Hybrid Gold Open Access. DOI: 10.1016/j.comnet.2019.106984.

[28] 3GPP System architecture for the 5G system (5GS) 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 23.501, v16.18.0 Sept. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-gi0.zip, (accessed on 13.02.2024).

[29] 3rd Generation Partnership Project (3GPP), 3GPP System architecture for the 5G system (5GS),Technical Specification (TS) 23.501, v17.12.0 Mar. 2024, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-hc0.zip, (accessed on 01.05.2024).

[30] 3rd Generation Partnership Project (3GPP); Technical Specification Group Core Network and Terminals; 5G System (5GS); Device-Side Time Sensitive Networking (TSN) Translator (DS-TT) to Network-Side TSN Translator (NW-TT) protocol aspects; Stage 3, Technical Specification (TS) 24.535, v17.2.0 Jan. 2022, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/24_series/24.535/24535-h20.zip, (accessed on 02.02.2024).

[31] 3rd Generation Partnership Project (3GPP); Technical Specification Group Core Network and Terminals; 5G System (5GS); Time-Sensitive Networking (TSN) Application Function (AF) to Device-Side TSN Translator (DS-TT) and Network-Side TSN Translator (NW-TT) protocol aspects; Stage 3, Technical Specification (TS) 24.519, v16.9.0 Sept. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/24_series/24.519/24519-g90.zip, (accessed on 13.02.2024).

UNIVERSIDAD DE MÁLAGA

[32] M. Gutiérrez, A. Ademaj, W. Steiner, R. Dobrin, and S. Punnekkat, "Self-configuration of IEEE 802.1 TSN networks," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8. DOI: 10.1109/ETFA.2017.8247597.

[33] M. Pozza, P. K. Nicholson, D. F. Lugones, A. Rao, H. Flinck, and S. Tarkoma, "On reconfiguring 5g network slices," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1542–1554, 2020. DOI: 10.1109/JSAC.2020.2986898.

[34] F. Luque-Schempp, L. Panizo, M.-d.-M. Gallardo, P. Merino, and J. Rivas, "Toward zero touch configuration of 5g non-public networks for time sensitive networking," *IEEE Network*, vol. 36, no. 2, pp. 50–56, 2022. DOI: 10.1109/MNET.006.2100442.

[35] F. Luque-Schempp, L. Panizo, M.-D.-M. Gallardo, and P. Merino, "Automadapt: Zero touch configuration of 5g qos flows extended for time-sensitive networking," *IEEE Access*, vol. 11, pp. 82 960–82 977, 2023. DOI: 10.1109/ACCESS.2023.3302264.

[36] N. Velásquez Villagrán, E. Estevez, P. Pesado, and J. De Juanes Marquez, "Standardization: A key factor of industry 4.0," in *2019 Sixth International Conference on eDemocracy and eGovernment (ICEDEG)*, 2019, pp. 350–354. DOI: 10.1109/ICEDEG.2019.8734339.

[37] A. Rojko, "Industry 4.0 concept: Background and overview," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 11, no. 5, pp. 77–90, Jul. 2017. DOI: 10.3991/ijim.v11i5.7072.

[38] C. E. Pereira, C. Diedrich, and P. Neumann, "Communication protocols for automation," *Springer Handbooks*, vol. Part F674, pp. 535–560, 2023. DOI: 10.1007/978-3-030-96729-1_23.

[39] C. E. Pereira and P. Neumann, "Industrial communication protocols," in *Springer Handbook of Automation*, S. Y. Nof, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 981–999. DOI: 10.1007/978-3-540-78831-7_56.

[40] E. Tovar and F. Vasques, "Real-time fieldbus communications using profibus networks," *IEEE Transactions on Industrial Electronics*, vol. 46, no. 6, pp. 1241–1251, 1999. DOI: 10.1109/41.808018.

[41] A. S. Siddiqui, Y. Gui, J. Plusquellic, and F. Saqib, "Secure communication over canbus," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2017, pp. 1264–1267. DOI: 10.1109/MWSCAS.2017.8053160.

[42] G. Thomas, "Introduction to the modbus protocol," *The Extension*, vol. 9, no. 4, pp. 1–4, 2008.

[43] P. S. Marshall and J. S. Rinaldi, *Industrial Ethernet*. ISA, 2004.

[44] A. L. Dias, G. S. Sestito, A. C. Turcato, and D. Brandão, "Panorama, challenges and opportunities in profinet protocol research," in *2018 13th IEEE International Conference on Industry Applications (INDUSCON)*, 2018, pp. 186–193. DOI: 10.1109/INDUSCON.2018.8627173.

[45] D. Jansen and H. Buttner, "Real-time ethernet the ethercat solution," *Computing and Control Engineering Journal*, vol. 15, pp. 16–21, Mar. 2004. DOI: 10.1049/cce:20040104.

UNIVERSIDAD DE MÁLAGA

[46] B. I. A. GmbH, *Ethernet powerlink xml device description – implementation guideline*.

[47] Q.-F. He, Q.-J. Zeng, and X.-M. Tang, "Research and implement on industry control networks based on embedded sercos-iii protocol," in *2011 International Conference on Electronics, Communications and Control (ICECC)*, 2011, pp. 3868–3872. DOI: 10.1109/ICECC.2011.6067582.

[48] M. Bengtsson and R. Örn, *Fieldbus communication module: An adapter for the control and communication link protocol*, 2000.

[49] S. K. Sen, *Fieldbus and networking in process automation*. CRC Press, 2021.

[50] A. Swales *et al.*, "Open modbus/tcp specification," *Schneider Electric*, vol. 29, no. 3, p. 19, 1999.

[51] H. Herath, S. Ariyathunge, and H. Priyankara, "Development of a data acquisition and monitoring system based on modbus rtu communication protocol," *Development*, vol. 5, no. 6, pp. 433–440, 2020.

[52] M. H. Schwarz and J. Börcsök, "A survey on opc and opc-ua: About the standard, developments and investigations," in *2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT)*, 2013, pp. 1–6. DOI: 10.1109/ICAT.2013.6684065.

[53] J. Farkas, L. L. Bello, and C. Gunther, "Time-sensitive networking standards," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 20–21, 2018. DOI: 10.1109/MCOMSTD.2018.8412457.

[54] S. Schriegel and J. Jasperneite, "A migration strategy for profinet toward ethernet tsn-based field-level communication: An approach to accelerate the adoption of converged it/ot communication," *IEEE Industrial Electronics Magazine*, vol. 15, no. 4, pp. 43–53, 2021. DOI: 10.1109/MIE.2020.3048925.

[55] K. Weber, "Ethercat and tsn-best practices for industrial ethernet system architectures," *EtherCAT Technology Group*, 2018.

[56] L. Sciullo, S. Bhattacharjee, and M. Kovatsch, "Bringing deterministic industrial networking to the w3c web of things with tsn and opc ua," Cited by: 10, 2020. DOI: 10.1145/3410992.3410997.

[57] A. B. D. Kinabo, J. B. Mwangama, and A. A. Lysko, "An overview of time-sensitive communications for the factory floor," in *2021 IST-Africa Conference (IST-Africa)*, 2021, pp. 1–9.

[58] N. Finn and P. Thubert, *Deterministic Networking Problem Statement*, RFC 8557, (accessed on 07.03.2024), May 2019. DOI: 10.17487/RFC8557.

[59] X. L. Yang, D. Scholz, and M. Helm, "Deterministic networking (detnet) vs time sensitive networking (tsn)," 2019.

[60] Y. J. Lin and B. K. Zhao, "A survey on queue scheduling algorithms in a router," in *7th International Symposium on Advances in Electrical, Electronics, and Computer Engineering*, T. Zhang, Ed., International Society for Optics and Photonics, vol. 12294, SPIE, 2022, p. 122943X. DOI: 10.1117/12.2641384.

UNIVERSIDAD DE MÁLAGA

[61] G. Sharma, W. Tavernier, D. Colle, and M. Pickavet, "Routing and scheduling for 1+1 protected detnet flows," *Computer Networks*, vol. 211, p. 108 960, May 2022. DOI: 10.1016/j.comnet.2022.108960.

[62] J. Krolikowski, S. Martin, P. Medagliani, *et al.*, "Joint routing and scheduling for large-scale deterministic ip networks," *Computer Communications*, vol. 165, pp. 33–42, 2021. DOI: https://doi.org/10.1016/j.comcom.2020.10.016.

[63] B. Liu, S. Ren, C. Wang, *et al.*, "Towards large-scale deterministic ip networks," in *2021 IFIP Networking Conference (IFIP Networking)*, 2021, pp. 1–9. DOI: 10.23919/IFIPNetworking52078.2021.9472798.

[64] K. Alexandris, P. Pop, and T. Wang, "Configuration and Evaluation of Multi-CQF Shapers in IEEE 802.1 Time-Sensitive Networking (TSN)," *IEEE Access*, vol. 10, pp. 109 068–109 081, 2022. DOI: 10.1109/ACCESS.2022.3214007.

[65] X. Su, C. Tang, and C. Gong, "Efficient traffic scheduling using genetic algorithm in time-sensitive networking," in *2022 6th International Symposium on Computer Science and Intelligent Control (ISCSIC)*, 2022, pp. 232–238. DOI: 10.1109/ISCSIC57216.2022.00056.

[66] A. A. Syed, S. Ayaz, T. Leinmüller, and M. Chandra, "Dynamic scheduling and routing for tsn based in-vehicle networks," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2021, pp. 1–6. DOI: 10.1109/ICCWorkshops50388.2021.9473810.

[67] Y. Huang, S. Wang, T. Huang, and Y. Liu, "Cycle-based time-sensitive and deterministic networks: Architecture, challenges, and open issues," *IEEE Communications Magazine*, vol. 60, no. 6, pp. 81–87, 2022. DOI: 10.1109/MCOM.001.2100865.

[68] J. Prados-Garzon, T. Taleb, and M. Bagaa, "Learnet: Reinforcement learning based flow scheduling for asynchronous deterministic networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6. DOI: 10.1109/ICC40277.2020.9149092.

[69] N. Kerö, A. Puhm, T. Kernen, and A. Mroczkowski, "Performance and reliability aspects of clock synchronization techniques for industrial automation," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 1011–1026, 2019. DOI: 10.1109/JPROC.2019.2915972.

[70] J. Prados-Garzon, L. Chinchilla-Romero, P. Ameigeiras, P. Muñoz, and J. M. Lopez-Soler, "Asynchronous time-sensitive networking for industrial networks," in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2021, pp. 130–135. DOI: 10.1109/EuCNC/6GSummit51104.2021.9482597.

[71] M. Gutiérrez, W. Steiner, R. Dobrin, and S. Punnekkat, "Synchronization Quality of IEEE 802.1AS in Large-Scale Industrial Automation Networks," in *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2017, pp. 273–282. DOI: 10.1109/RTAS.2017.10.

[72] Network Time Protocol Version 4: Protocol and Algorithms Specification, Internet Engineering Task Force (IETF), June 2010, [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc5905.txt.pdf, (accessed on 13.01.2024).

UNIVERSIDAD DE MÁLAGA

[73] Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI, Internet Engineering Task Force (IETF), Jan. 2006, [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc4330.txt.pdf, (accessed on 13.01.2024).

[74] GPS Interface Specification IS-GPS-200, Revision N, RFC-00467, Aug. 2022, [Online]. Available: https://www.navcen.uscg.gov/sites/default/files/pdf/gps/IS-GPS-200N.pdf, (accessed on 13.01.2024).

[75] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," *IEEE Std 1588-2019 (Revision ofIEEE Std 1588-2008)*, pp. 1–499, 2020, (accessed on 13.01.2024). DOI: 10.1109/IEEESTD.2020.9120376.

[76] D. Chefrour, "Evolution of network time synchronization towards nanoseconds accuracy: A survey," *Computer Communications*, vol. 191, pp. 26–35, 2022. DOI: https://doi.org/10.1016/j.comcom.2022.04.023.

[77] I. Freire, I. Sousa, A. Klautau, I. Almeida, C. Lu, and M. Berg, "Analysis and evaluation of end-to-end ptp synchronization for ethernet-based fronthaul," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6. DOI: 10.1109/GLOCOM.2016.7841597.

[78] W. Chen, J. Sun, L. Zhang, X. Liu, and L. Hong, "An implementation of IEEE 1588 protocol for IEEE 802.11 WLAN," *Wireless Networks*, vol. 21, no. 6, pp. 2069–2085, Aug. 2015. DOI: 10.1007/s11276-015-0898-z.

[79] A. Mahmood, R. Exel, and T. Sauter, "Delay and jitter characterization for software-based clock synchronization over wlan using ptp," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1198–1206, 2014. DOI: 10.1109/TII.2014.2304413.

[80] M. Schüngel, S. Dietrich, D. Ginthör, S.-P. Chen, and M. Kuhn, "Single message distribution of timing information for time synchronization in converged wired and wireless networks," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 286–293. DOI: 10.1109/ETFA46521.2020.9211936.

[81] A. Willig, K. Matheus, and A. Wolisz, "Wireless technology in industrial networks," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1130–1151, 2005, Cited by: 577; All Open Access, Green Open Access. DOI: 10.1109/JPROC.2005.849717.

[82] T. Sauter, J. Jasperneite, and L. Lo Bello, "Towards new hybrid networks for industrial automation," in *2009 IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, 2009, pp. 1–8. DOI: 10.1109/ETFA.2009.5347127.

[83] G. Cena, A. Valenzano, and S. Vitturi, "Hybrid wired/wireless networks for real-time communications," *IEEE Industrial Electronics Magazine*, vol. 2, no. 1, pp. 8–20, 2008. DOI: 10.1109/MIE.2008.917155.

[84] D. M. Anand, J. R. Moyne, and D. M. Tilbury, "Performance evaluation of wireless networks for factory automation applications," in *2009 IEEE International Conference on Automation Science and Engineering*, 2009, pp. 340–346. DOI: 10.1109/COASE.2009.5234126.

UNIVERSIDAD DE MÁLAGA

[85] Ó. Seijo, X. Iturbe, and I. Val, "Tackling the challenges of the integration of wired and wireless tsn with a technology proof-of-concept," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7361–7372, 2022. DOI: 10.1109/TII.2021.3131865.

[86] A. Frotzscher, U. Wetzker, M. Bauer, *et al.*, "Requirements and current solutions of wireless communication in industrial automation," in *2014 IEEE International Conference on Communications Workshops (ICC)*, 2014, pp. 67–72. DOI: 10.1109/ICCW.2014.6881174.

[87] *Industrial network market shares 2023*, [Online]. Available: https://www.hms-networks.com/news-and-insights/news-from-hms/2023/05/05/industrial-network-market-shares-2023, (accessed on 02.02.2024).

[88] A. Larrañaga, M. C. Lucas-Estañ, I. Martinez, I. Val, and J. Gozalvez, "Analysis of 5g-tsn integration to support industry 4.0," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1111–1114. DOI: 10.1109/ETFA46521.2020.9212141.

[89] P. M. Rost and T. Kolding, "Performance of Integrated 3GPP 5G and IEEE TSN Networks," *IEEE Communications Standards Magazine*, vol. 6, no. 2, pp. 51–56, 2022. DOI: 10.1109/MCOMSTD.0001.2000013.

[90] D. Ginthör, J. von Hoyningen-Huene, R. Guillaume, and H. Schotten, "Analysis of multi-user scheduling in a tsn-enabled 5g system for industrial applications," in *2019 IEEE International Conference on Industrial Internet (ICII)*, 2019, pp. 190–199. DOI: 10.1109/ICII.2019.00044.

[91] J. Yang and G. Yu, "Traffic scheduling for 5g-tsn integrated systems," in *2022 International Symposium on Wireless Communication Systems (ISWCS)*, 2022, pp. 1–6. DOI: 10.1109/ISWCS56560.2022.9940254.

[92] A. Garbugli, L. Rosa, L. Foschini, A. Corradi, and P. Bellavista, "A framework for tsn-enabled virtual environments for ultra-low latency 5g scenarios," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 5023–5028. DOI: 10.1109/ICC45855.2022.9839193.

[93] J. Song, M. Kubomi, J. Zhao, and D. Takita, "Time synchronization performance analysis considering the frequency offset inside 5g-tsn network," in *2021 17th International Symposium on Wireless Communication Systems (ISWCS)*, 2021, pp. 1–6. DOI: 10.1109/ISWCS49558.2021.9562179.

[94] T. Striffler and H. D. Schotten, "The 5g transparent clock: Synchronization errors in integrated 5g-tsn industrial networks," in *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, 2021, pp. 1–6. DOI: 10.1109/INDIN45523.2021.9557468.

[95] M.-T. Thi, S. Guédon, S. B. H. Said, *et al.*, "IEEE 802.1 TSN Time Synchronization over Wi-Fi and 5G Mobile Networks," in *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*, 2022, pp. 1–7. DOI: 10.1109/VTC2022-Fall57202.2022.10012852.

[96] Y. Zhang, Q. Xu, M. Li, C. Chen, and X. Guan, "Qos-aware mapping and scheduling for virtual network functions in industrial 5g-tsn network," in *2021*

UNIVERSIDAD DE MÁLAGA

IEEE Global Communications Conference (GLOBECOM), 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685749.

[97] Z. Satka, M. Ashjaei, H. Fotouhi, M. Daneshtalab, M. Sjödin, and S. Mubeen, "Qos-man: A novel qos mapping algorithm for tsn-5g flows," in *2022 IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2022, pp. 220–227. DOI: 10.1109/RTCSA55878.2022.00030.

[98] Z. Satka, D. Pantzar, A. Magnusson, *et al.*, "Developing a translation technique for converged tsn-5g communication," in *2022 IEEE 18th International Conference on Factory Communication Systems (WFCS)*, 2022, pp. 1–8. DOI: 10.1109/WFCS53837.2022.9779191.

[99] Y. Cai, X. Zhang, S. Hu, and X. Wei, "Dynamic qos mapping and adaptive semi-persistent scheduling in 5g-tsn integrated networks," *China Communications*, vol. 20, no. 4, pp. 340–355, 2023. DOI: 10.23919/JCC.fa.2022-0548.202304.

[100] S. Senk, H. K. Nazari, H.-H. Liu, G. T. Nguyen, and F. H. P. Fitzek, "Open-source testbeds for integrating time-sensitive networking with 5g and beyond," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, 2023, pp. 1–7. DOI: 10.1109/CCNC51644.2023.10060159.

[101] N. Ambrosy, A. Abazari, and L. Underberg, *Simulations of the 5G-TSN Bridge Delay: Towards a Joint QoS Model*. Universitätsbibliothek Otto-von-Guericke-Universität, 2023.

[102] A. Magnusson and D. Pantzar, *Integrating 5g components into a tsn discrete event simulation framework*, 2021.

[103] S.-J. Wu, C. Chen, and J.-C. Chen, "Design and implementation of data plane supporting time-sensitive networking in 5g networks," in *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*, 2024, pp. 905–911. DOI: 10.1109/CCNC51664.2024.10454733.

[104] S. Bhattacharjee, K. Katsalis, O. Arouk, *et al.*, "Network slicing for tsn-based transport networks," *IEEE Access*, vol. 9, pp. 62 788–62 809, 2021. DOI: 10.1109/ACCESS.2021.3074802.

[105] M. Gundall, C. Huber, P. Rost, R. Halfmann, and H. D. Schotten, "Integration of 5G with TSN as Prerequisite for a Highly Flexible Future Industrial Automation: Time Synchronization based on IEEE 802.1AS," in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, 2020, pp. 3823–3830. DOI: 10.1109/IECON43393.2020.9254296.

[106] P. Kehl, J. Ansari, M. H. Jafari, *et al.*, "Prototype of 5g integrated with tsn for edge-controlled mobile robotics," *Electronics*, vol. 11, no. 11, 2022. DOI: 10.3390/electronics11111666.

[107] S. Senk, S. A. W. Itting, J. Gabriel, *et al.*, "5g nsa and sa campus network testbeds for evaluating industrial automation," in *European Wireless 2021; 26th European Wireless Conference*, 2021, pp. 1–8.

[108] K. Nikhileswar, K. Prabhu, D. Cavalcanti, and A. Regev, "Time-sensitive networking over 5g for industrial control systems," in *2022 IEEE 27th International*

UNIVERSIDAD DE MÁLAGA

Conference on Emerging Technologies and Factory Automation (ETFA), 2022, pp. 1–8. DOI: 10.1109/ETFA52439.2022.9921680.

[109] T. Muhammad and M. Munir, "Network automation," *European Journal of Technology*, vol. 7, no. 2, pp. 23–42, Aug. 2023. DOI: 10.47672/ejt.1547.

[110] 3rd Generation Partnership Project (3GPP); Technical Specification Group Services and System Aspects; Telecommunication Management; Self-Organizing Networks (SON); Concepts and requirements, Technical Specification (TS) 32.500, v17.0.0 Apr. 2022, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/32_series/32.500/32500-h00.zip, (accessed on 07.03.2024).

[111] Next Generation Mobile Networks Alliance; NGMN Use Cases related to Self Organising Network, Overall Description, v2.02 Dec. 2008, [Online]. Available: https://ngmn.org/wp-content/uploads/NGMN_Use_Cases_related_to_Self_Organising_Network_Overall_Description.pdf, (accessed on 07.03.2024).

[112] S. Hämäläinen, H. Sanneck, and C. Sartori, *LTE self-organising networks (SON): network management automation for operational efficiency*. John Wiley and Sons, 2012.

[113] A. A. Atayero, O. I. Adu, and A. A. Alatishe, "Self Organizing Networks for 3GPP LTE", booktitle="Computational Science and Its Applications – ICCSA 2014," B. Murgante, S. Misra, A. M. A. C. Rocha, *et al.*, Eds., Cham: Springer International Publishing, 2014, pp. 242–254.

[114] J. Zhang, C. Sun, Y. Yi, and H. Zhuang, "A hybrid framework for capacity and coverage optimization in self-organizing lte networks," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2013, pp. 2919–2923. DOI: 10.1109/PIMRC.2013.6666646.

[115] M. K. Hasan, A. F. Ismail, A. H. Abdalla, *et al.*, "Inter-cell interference coordination in lte-a hetnets: A survey on self organizing approaches," in *2013 INTERNATIONAL CONFERENCE ON COMPUTING, ELECTRICAL AND ELECTRONIC ENGINEERING (ICCEEE)*, 2013, pp. 196–201. DOI: 10.1109/ICCEEE.2013.6633932.

[116] I. N. Md Isa, M. D. Baba, R. Ab Rahman, and A. L. Yusof, "Self-organizing network based handover mechanism for lte networks," in *2015 International Conference on Computer, Communications, and Control Technology (I4CT)*, 2015, pp. 11–15. DOI: 10.1109/I4CT.2015.7219527.

[117] T. Bag, S. Garg, D. Preciado, Z. Shaik, J. Mueckenheim, and A. Mitschele-Thiel, "Self-organizing network functions for handover optimization in lte cellular networks," in *Mobile Communication - Technologies and Applications; 24. ITG-Symposium*, 2019, pp. 1–7.

[118] S. Chaudhuri, I. Baig, and D. Das, "Self organizing method for handover performance optimization in lte-advanced network," *Computer Communications*, vol. 110, pp. 151–163, 2017. DOI: https://doi.org/10.1016/j.comcom.2017.06.009.

[119] M. I. Tiwana, "Self organizing networks: A reinforcement learning approach for self-optimization of lte mobility parameters," *Automatika*, vol. 55, no. 4,

UNIVERSIDAD DE MÁLAGA

pp. 504–513, 2014. DOI: 10.7305/automatika.2014.12.502. eprint: https://doi.org/10.7305/automatika.2014.12.502.

[120] O. Østerbø and O. Grøndalen, "Benefits of self-organizing networks (son) for mobile operators," *Journal of Computer Networks and Communications*, vol. 2012, p. 862 527, Sep. 2012. DOI: 10.1155/2012/862527.

[121] Y. Ouyang, Z. Li, L. Su, W. Lu, and Z. Lin, "Application behaviors driven self-organizing network (son) for 4g lte networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 3–14, 2020. DOI: 10.1109/TNSE.2018.2877353.

[122] K. N. Premnath and S. Rajavelu, "Challenges in self organizing networks for wireless telecommunications," in *2011 International Conference on Recent Trends in Information Technology (ICRTIT)*, 2011, pp. 1331–1334. DOI: 10.1109/ICRTIT.2011.5972332.

[123] H. Fourati, R. Maaloul, L. Chaari, and M. Jmaiel, "Comprehensive survey on self-organizing cellular network approaches applied to 5g networks," *Computer Networks*, vol. 199, p. 108 435, 2021. DOI: https://doi.org/10.1016/j.comnet.2021.108435.

[124] R. Bo, Q. Xuesong, M. Kadoch, S. Sun, and W. Li, *5G Heterogeneous Networks, Self-organizing and Optimization*. Jan. 2016. DOI: 10.1007/978-3-319-39372-8.

[125] Y.-Z. Sun, S.-Y. Leng, Y.-C. Lai, C. Grebogi, and W. Lin, "Closed-loop control of complex networks: A trade-off between time and energy," *Phys. Rev. Lett.*, vol. 119, p. 198 301, 19 Nov. 2017. DOI: 10.1103/PhysRevLett.119.198301.

[126] H. CHAN and Ü. ÖZGÜNER, "Closed-loop control of systems over a communications network with queues," *International Journal of Control*, vol. 62, no. 3, pp. 493–510, 1995. DOI: 10.1080/00207179508921553. eprint: https://doi.org/10.1080/00207179508921553.

[127] N. Ploplys, P. Kawka, and A. Alleyne, "Closed-loop control over wireless networks," *IEEE Control Systems Magazine*, vol. 24, no. 3, pp. 58–71, 2004. DOI: 10.1109/MCS.2004.1299533.

[128] M. Pajic, S. Sundaram, J. Le Ny, G. J. Pappas, and R. Mangharam, "Closing the loop: A simple distributed method for control over wireless networks," ser. IPSN '12, Beijing, China: Association for Computing Machinery, 2012, pp. 25–36. DOI: 10.1145/2185677.2185681.

[129] A. D. Ambike, "Closed-loop real-time control on distributed networks," Master's thesis, Texas A and M University., 2004.

[130] N. F. S. d. Sousa and C. E. Rothenberg, "Clara: Closed loop-based zero-touch network management framework," in *2021 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2021, pp. 110–115. DOI: 10.1109/NFV-SDN53031.2021.9665048.

[131] B. K. Saha, D. Tandur, L. Haab, and L. Podleski, "Intent-based networks: An industrial perspective," ser. FICN '18, New Delhi, India: Association for Computing Machinery, 2018, pp. 35–40. DOI: 10.1145/3243318.3243324.

UNIVERSIDAD DE MÁLAGA

[132] E. Zeydan and Y. Turk, "Recent advances in intent-based networking: A survey," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–5. DOI: 10.1109/VTC2020-Spring48590.2020.9128422.

[133] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *IEEE Communications Surveys and Tutorials*, vol. 25, no. 1, pp. 625–655, 2023. DOI: 10.1109/COMST.2022.3215919.

[134] B. E. Ujcich, A. Bates, and W. H. Sanders, "Provenance for intent-based networking," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 195–199. DOI: 10.1109/NetSoft48620.2020.9165519.

[135] M. Beshley, A. Pryslupskyi, O. Panchenko, and H. Beshley, "Sdn/cloud solutions for intent-based networking," in *2019 3rd International Conference on Advanced Information and Communications Technologies (AICT)*, 2019, pp. 22–25. DOI: 10.1109/AIACT.2019.8847731.

[136] M. Riftadi and F. Kuipers, "P4i/o: Intent-based networking with p4," in *2019 IEEE Conference on Network Softwarization (NetSoft)*, 2019, pp. 438–443. DOI: 10.1109/NETSOFT.2019.8806662.

[137] T. A. Khan, A. Muhammad, K. Abbas, and W.-C. Song, "Intent-based networking platform: An automated approach for policy and configuration of next-generation networks," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, ser. SAC '21, Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 1921–1930. DOI: 10.1145/3412841.3442064.

[138] A. Campanella, "Intent based network operations," in *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, 2019, pp. 1–3.

[139] L. Velasco, M. Signorelli, O. G. De Dios, *et al.*, "End-to-end intent-based networking," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 106–112, 2021. DOI: 10.1109/MCOM.101.2100141.

[140] K. Abbas, T. A. Khan, M. Afaq, and W.-C. Song, "Network slice lifecycle management for 5g mobile networks: An intent-based networking approach," *IEEE Access*, vol. 9, pp. 80 128–80 146, 2021. DOI: 10.1109/ACCESS.2021.3084834.

[141] Y. Tsuzaki and Y. Okabe, "Reactive configuration updating for intent-based networking," in *2017 International Conference on Information Networking (ICOIN)*, 2017, pp. 97–102. DOI: 10.1109/ICOIN.2017.7899484.

[142] Y. Wei, M. Peng, and Y. Liu, "Intent-based networks for 6g: Insights and challenges," *Digital Communications and Networks*, vol. 6, no. 3, pp. 270–280, 2020. DOI: https://doi.org/10.1016/j.dcan.2020.07.001.

[143] A. Collet, A. Banchs, and M. Fiore, "Lossleap: Learning to predict for intent-based networking," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 2138–2147. DOI: 10.1109/INFOCOM48880.2022.9796918.

[144] P. Szilágyi, "I2bn: Intelligent intent based networks," *Journal of ICT Standardization*, vol. 9, no. 2, pp. 159–200, 2021. DOI: 10.13052/jicts2245-800X.926.

[145] S.-C. Lin, C.-H. Lin, and W.-C. Chen, *Zero-touch network on industrial iot: An end-to-end machine learning approach*, 2022. arXiv: 2204.12605 [cs.LG].

UNIVERSIDAD DE MÁLAGA

[146] I. Šimunić and I. Grgurević, "Automation of network device configuration using zero-touch provisioning - a case study," in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, D. Perakovic and L. Knapcikova, Eds., Cham: Springer International Publishing, 2021, pp. 105–119.

[147] I. Boškov, H. Yetgin, M. Vučnik, C. Fortuna, and M. Mohorčič, "Time-to-provision evaluation of iot devices using automated zero-touch provisioning," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–7. DOI: 10.1109/GLOBECOM42002.2020.9348119.

[148] I. Šimunić and I. Grgurević, "Automation of network device configuration using zero-touch provisioning - a case study," in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, D. Perakovic and L. Knapcikova, Eds., Cham: Springer International Publishing, 2021, pp. 105–119.

[149] C. de Alwis, Q.-V. Pham, and M. Liyanage, "Zero-touch network and service management," in *6G Frontiers: Towards Future Wireless Systems*. 2023, pp. 63–72. DOI: 10.1002/9781119862321.ch6.

[150] B. Koley, "The zero touch network," in *International Conference on Network and Service Management (CNSM)*, 2016.

[151] J. Ma, Y. Guo, C. Fang, and Q. Zhang, "Digital twin-based zero-touch management for iot," *Electronics*, vol. 11, no. 24, 2022. DOI: 10.3390/electronics11244104.

[152] M. Friesen, L. Wisniewski, and J. Jasperneite, "Machine learning for zero- touch management in heterogeneous industrial networks - a review," in *2022 IEEE 18th International Conference on Factory Communication Systems (WFCS)*, 2022, pp. 1–8. DOI: 10.1109/WFCS53837.2022.9779183.

[153] L. Bonati, S. D'Oro, L. Bertizzolo, *et al.*, "Cellos: Zero-touch softwarized open cellular networks," *Computer Networks*, vol. 180, p. 107 380, 2020. DOI: https://doi.org/10.1016/j.comnet.2020.107380.

[154] M. Liyanage, Q.-V. Pham, K. Dev, *et al.*, "A survey on zero touch network and service management (zsm) for 5g and beyond networks," *Journal of Network and Computer Applications*, vol. 203, p. 103 362, 2022. DOI: https://doi.org/10.1016/j.jnca.2022.103362.

[155] E. Coronado, R. Behravesh, T. Subramanya, *et al.*, "Zero touch management: A survey of network automation solutions for 5g and 6g networks," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 4, pp. 2535–2578, 2022. DOI: 10.1109/COMST.2022.3212586.

[156] R. Niboucha, S. B. Saad, A. Ksentini, and Y. Challal, "Zero-touch security management for mmtc network slices: Ddos attack detection and mitigation," *IEEE Internet of Things Journal*, vol. 10, no. 9, pp. 7800–7812, 2023. DOI: 10.1109/JIOT.2022.3230875.

[157] S. Ben Saad, B. Brik, and A. Ksentini, "Toward securing federated learning against poisoning attacks in zero touch b5g networks," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1612–1624, 2023. DOI: 10.1109/TNSM.2023.3278838.

UNIVERSIDAD DE MÁLAGA

[158] H. Chergui, L. Blanco, L. A. Garrido, *et al.*, "Zero-touch ai-driven distributed management for energy-efficient 6g massive network slicing," *IEEE Network*, vol. 35, no. 6, pp. 43–49, 2021. DOI: 10.1109/MNET.111.2100322.

[159] P. Yu, F. Zhou, X. Zhang, X. Qiu, M. Kadoch, and M. Cheriet, "Deep learning-based resource allocation for 5g broadband tv service," *IEEE Transactions on Broadcasting*, vol. 66, no. 4, pp. 800–813, 2020. DOI: 10.1109/TBC.2020.2968730.

[160] S. Yoon, J.-H. Cho, D. S. Kim, T. J. Moore, F. Free-Nelson, and H. Lim, "Desolater: Deep reinforcement learning-based resource allocation and moving target defense deployment framework," *IEEE Access*, vol. 9, pp. 70 700–70 714, 2021. DOI: 10.1109/ACCESS.2021.3076599.

[161] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia, "Neutran: An open ran neutral host architecture for zero-touch ran and spectrum sharing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 5786–5798, 2024. DOI: 10.1109/TMC.2023.3311728.

[162] Y. Zhang, L. Xiong, and J. Yu, "Deep learning based user association in heterogeneous wireless networks," *IEEE Access*, vol. 8, pp. 197 439–197 447, 2020. DOI: 10.1109/ACCESS.2020.3033133.

[163] H. Gacanin, E. Perenda, and R. Atawia, "Self-deployment of non-stationary wireless systems by knowledge management with artificial intelligence," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1004–1018, 2019. DOI: 10.1109/TCCN.2019.2932418.

[164] B. Angui, R. Corbel, V. Q. Rodriguez, and E. Stephan, "Towards 6g zero touch networks: The case of automated cloud-ran deployments," in *2022 IEEE 19th Annual Consumer Communications and Networking Conference (CCNC)*, 2022, pp. 1–6. DOI: 10.1109/CCNC49033.2022.9700507.

[165] M. Pahlevan, J. Schmeck, and R. Obermaisser, "Evaluation of tsn dynamic configuration model for safety-critical applications," in *2019 IEEE Intl Conf on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking (ISPA/BDCloud/SocialCom/SustainCom)*, 2019, pp. 566–571. DOI: 10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00086.

[166] S. Seitner, "Tsn scheduling with re-configuration for dynamic industrial networks," Ph.D. dissertation, Technische Universität Wien, 2023.

[167] I. Álvarez, A. Servera, J. Proenza, M. Ashjaei, and S. Mubeen, "Implementing a first cnc for scheduling and configuring tsn networks," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2022, pp. 1–4. DOI: 10.1109/ETFA52439.2022.9921518.

[168] T. Kobzan, I. Blöcher, M. Hendel, *et al.*, "Configuration solution for tsn-based industrial networks utilizing sdn and opc ua," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1629–1636. DOI: 10.1109/ETFA46521.2020.9211897.

[169] L. Dürkop, J. Imtiaz, H. Trsek, L. Wisniewski, and J. Jasperneite, "Using opc-ua for the autoconfiguration of real-time ethernet systems," in *2013 11th IEEE*

UNIVERSIDAD DE MÁLAGA

International Conference on Industrial Informatics (INDIN), 2013, pp. 248–253. DOI: 10.1109/INDIN.2013.6622890.

[170] M. L. Raagaard, P. Pop, M. Gutiérrez, and W. Steiner, "Runtime reconfiguration of time-sensitive networking (tsn) schedules for fog computing," in *2017 IEEE Fog World Congress (FWC)*, 2017, pp. 1–6. DOI: 10.1109/FWC.2017.8368523.

[171] A. El-Mougy, M. Ibnkahla, G. Hattab, and W. Ejaz, "Reconfigurable wireless networks," *Proceedings of the IEEE*, vol. 103, no. 7, pp. 1125–1158, 2015. DOI: 10.1109/JPROC.2014.2376812.

[172] J. S. Camargo, E. Coronado, W. Ramirez, *et al.*, "Dynamic slicing reconfiguration for virtualized 5g networks using ml forecasting of computing capacity," *Computer Networks*, vol. 236, p. 110 001, 2023. DOI: https://doi.org/10.1016/j.comnet.2023.110001.

[173] D. Sabella, P. Serrano, G. Stea, *et al.*, "Designing the 5g network infrastructure: A flexible and reconfigurable architecture based on context and content information," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, p. 199, Aug. 2018. DOI: 10.1186/s13638-018-1215-1.

[174] I. Leyva-Pupo and C. Cervelló-Pastor, "An intelligent scheduling for 5g user plane function placement and chaining reconfiguration," *Computer Networks*, vol. 237, p. 110 037, 2023. DOI: https://doi.org/10.1016/j.comnet.2023.110037.

[175] M. G. Pérez, A. H. Celdrán, F. Ippoliti, *et al.*, "Dynamic reconfiguration in 5g mobile networks to proactively detect and mitigate botnets," *IEEE Internet Computing*, vol. 21, no. 5, pp. 28–36, 2017. DOI: 10.1109/MIC.2017.3481345.

[176] C. Christophorou, A. Pitsillides, and I. Akyildiz, "Celec framework for reconfigurable small cells as part of 5g ultra-dense networks," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–7. DOI: 10.1109/ICC.2017.7996489.

[177] G. Miranda, E. Municio, J. Haxhibeqiri, J. Hoebeke, I. Moerman, and J. M. Marquez-Barja, "Enabling time-sensitive network management over multi-domain wired/wi-fi networks," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 2386–2399, 2023. DOI: 10.1109/TNSM.2023.3274590.

[178] K. A. Nsiah, K. Alkhouri, and A. Sikora, "Configuration of wireless tsn networks," in *2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*, 2020, pp. 1–5. DOI: 10.1109/IDAACS-SWS50031.2020.9297066.

[179] N. Sousa, M. T. Islam, R. Mustafa, D. Perez, C. Esteve Rothenberg, and P. Gomes, "Machine learning-assisted closed-control loops for beyond 5g multi-domain zero-touch networks," *Journal of Network and Systems Management*, vol. 30, Jul. 2022. DOI: 10.1007/s10922-022-09651-x.

[180] C. Benzaid and T. Taleb, "Ai-driven zero touch network and service management in 5g and beyond: Challenges and research directions," *IEEE Network*, vol. 34, no. 2, pp. 186–194, 2020. DOI: 10.1109/MNET.001.1900252.

[181] R. Sutton and A. Barto, "Reinforcement learning: An introduction," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 1054–1054, 1998. DOI: 10.1109/TNN.1998.712192.

UNIVERSIDAD DE MÁLAGA

[182] K. S. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*. USA: Prentice-Hall, Inc., 1989.

[183] D. Angluin, "Learning regular sets from queries and counterexamples," *Inf. Comput.*, vol. 75, no. 2, pp. 87–106, 1987. DOI: 10.1016/0890-5401(87)90052-6.

[184] B. Steffen, F. Howar, and M. Merten, "Introduction to active automata learning from a practical perspective," in *Formal Methods for Eternal Networked Software Systems - 11th Int. School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM 2011. Advanced Lectures*, M. Bernardo and V. Issarny, Eds., ser. LNCS, vol. 6659, Springer, 2011, pp. 256–296. DOI: 10.1007/978-3-642-21455-4_8.

[185] R. Yampolskiy, "Behavioral modeling: An overview," *American Journal of Applied Sciences*, vol. 5, May 2008. DOI: 10.3844/ajassp.2008.496.503.

[186] D. Lorenzoli, L. Mariani, and M. Pezzè, "Automatic generation of software behavioral models," in *Proceedings of the 30th International Conference on Software Engineering*, ser. ICSE '08, Leipzig, Germany: Association for Computing Machinery, 2008, pp. 501–510. DOI: 10.1145/1368088.1368157.

[187] D. FitzPatrick and I. Miller, *Analog behavioral modeling with the Verilog-A language*. Springer Science & Business Media, 1998.

[188] M. D. Cin, "Modeling fault-tolerant system behavior," in *Systems: Theory and Practice*, R. Albrecht, Ed. Vienna: Springer Vienna, 1998, pp. 213–234. DOI: 10.1007/978-3-7091-6451-8_10.

[189] M. Yaghoubi Suraki, "Fclcc: Fuzzy cross-layer congestion control in mobile ad hoc networks," vol. 18, Jan. 2018.

[190] J. Sun, W. Shi, Z. Yang, J. Yang, and G. Gui, "Behavioral modeling and linearization of wideband rf power amplifiers using bilstm networks for 5g wireless systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 10 348–10 356, 2019. DOI: 10.1109/TVT.2019.2925562.

[191] S. Das, A. dey, A. Pal, and N. Roy, "Applications of artificial intelligence in machine learning: Review and prospect," *International Journal of Computer Applications*, vol. 115, pp. 31–41, Apr. 2015. DOI: 10.5120/20182-2402.

[192] D. Rafique and L. Velasco, "Machine learning for network automation: Overview, architecture, and applications [invited tutorial]," *Journal of Optical Communications and Networking*, vol. 10, no. 10, pp. D126–D143, 2018. DOI: 10.1364/JOCN.10.00D126.

[193] A. Martínez Lopera, "Scheduling in tsn networks using machine learning," B.S. thesis, Universitat Politècnica de Catalunya, 2023.

[194] D. Bezerra, A. T. de Oliveira Filho, I. R. Rodrigues, *et al.*, "A machine learning-based optimization for end-to-end latency in tsn networks," *Computer Communications*, vol. 195, pp. 424–440, 2022. DOI: https://doi.org/10.1016/j.comcom.2022.09.011.

[195] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. Emad Ul Haq, "Machine learning techniques for 5g and beyond," *IEEE Access*, vol. 9, pp. 23 472–23 488, 2021. DOI: 10.1109/ACCESS.2021.3051557.

UNIVERSIDAD DE MÁLAGA

[196] J. E. Preciado-Velasco, J. D. Gonzalez-Franco, C. E. Anias-Calderon, J. I. Nieto-Hipolito, and R. Rivera-Rodriguez, "5g/b5g service classification using supervised learning," *Applied Sciences*, vol. 11, no. 11, 2021. DOI: 10.3390/app11114942.

[197] J. Lam and R. Abbas, *Machine learning based anomaly detection for 5g networks*, 2020. arXiv: 2003.03474 [cs.CR].

[198] F. Tang, Y. Zhou, and N. Kato, "Deep reinforcement learning for dynamic uplink/downlink resource allocation in high mobility 5g hetnet," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 12, pp. 2773–2782, 2020. DOI: 10.1109/JSAC.2020.3005495.

[199] J. Prados-Garzon and T. Taleb, "Asynchronous time-sensitive networking for 5g backhauling," *IEEE Network*, vol. 35, no. 2, pp. 144–151, 2021. DOI: 10.1109/MNET.011.2000402.

[200] X. Wang, H. Yao, T. Mai, S. Guo, and Y. Liu, "Reinforcement learning-based particle swarm optimization for end-to-end traffic scheduling in tsn-5g networks," *IEEE/ACM Transactions on Networking*, vol. 31, no. 6, pp. 3254–3268, 2023. DOI: 10.1109/TNET.2023.3276363.

[201] M. N. Qureshi, M. I. Tiwana, and M. Haddad, "Distributed self optimization techniques for heterogeneous network environments using active antenna tilt systems," *Telecommun. Syst.*, vol. 70, no. 3, pp. 379–389, Mar. 2019. DOI: 10.1007/s11235-018-0494-5.

[202] Y. Guo, S. Li, W. Jiang, B. Zhang, and Y. Ma, "Learning automata-based algorithms for solving the stochastic shortest path routing problems in 5g wireless communication," *Physical Commun.*, vol. 25, pp. 376–385, 2017. DOI: https://doi.org/10.1016/j.phycom.2017.06.010.

[203] P. P. Angelov, E. A. Soares, R. Jiang, N. I. Arnold, and P. M. Atkinson, "Explainable artificial intelligence: An analytical review," *WIREs Data Mining and Knowledge Discovery*, vol. 11, no. 5, e1424, 2021. DOI: https://doi.org/10.1002/widm.1424. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1424.

[204] I. Ahmed, G. Jeon, and F. Piccialli, "From artificial intelligence to explainable artificial intelligence in industry 4.0: A survey on what, how, and where," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5031–5042, 2022. DOI: 10.1109/TII.2022.3146552.

[205] W. Guo, "Explainable artificial intelligence for 6g: Improving trust between human and machine," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, 2020. DOI: 10.1109/MCOM.001.2000050.

[206] A. Terra, R. Inam, S. Baskaran, P. Batista, I. Burdick, and E. Fersman, "Explainability methods for identifying root-cause of sla violation prediction in 5g network," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–7. DOI: 10.1109/GLOBECOM42002.2020.9322496.

[207] W. Yang, Y. Wei, H. Wei, *et al.*, "Survey on explainable ai: From approaches, limitations and applications aspects," *Human-Centric Intelligent Systems*, vol. 3, Aug. 2023. DOI: 10.1007/s44230-023-00038-y.

UNIVERSIDAD DE MÁLAGA

[208] J. E. Hopcroft, R. Motwani, and J. D. Ullman, "Introduction to automata theory, languages, and computation, 2nd edition," *SIGACT News*, vol. 32, no. 1, pp. 60–65, Mar. 2001. DOI: 10.1145/568438.568455.

[209] B. K. Aichernig, E. Muškardin, and A. Pferscher, "Active vs. passive: A comparison of automata learning paradigms for network protocols," *Electronic Proceedings in Theoretical Computer Science*, vol. 371, pp. 1–19, Sep. 2022. DOI: 10.4204/eptcs.371.1.

[210] C. McMahon Stone, T. Chothia, and J. de Ruiter, "Extending automated protocol state learning for the 802.11 4-way handshake," in *Computer Security*, J. Lopez, J. Zhou, and M. Soriano, Eds., Cham: Springer International Publishing, 2018, pp. 325–345.

[211] M. Chlosta, D. Rupprecht, and T. Holz, "On the challenges of automata reconstruction in LTE networks," in *Proc. 14th ACM Conf. on Security and Privacy in Wireless and Mobile Netw. (WiSec '21)*, Abu Dhabi, United Arab Emirates: Association for Computing Machinery, 2021, pp. 164–174. DOI: 10.1145/3448300.3469133.

[212] J. Antunes, N. Neves, and P. Verissimo, "Reverse engineering of protocols from network traces," in *Proc. 18th Working Conf. on Reverse Eng.*, 2011, pp. 169–178. DOI: 10.1109/WCRE.2011.28.

[213] C. Lee, J. Bae, and H. Lee, "Prett: Protocol reverse engineering using binary tokens and network traces," in *ICT Systems Security and Privacy Protection*, L. J. Janczewski and M. Kutyłowski, Eds., Cham: Springer International Publishing, 2018, pp. 141–155.

[214] *Victoria network experimentation platform*, [Online]. Available: https://www.victoria-network.eu/, (accessed on 01.05.2024).

[215] *Tsn bridge pcie brochure*, [Online]. Available: https://www.relyum.com/web/wp-content/uploads/Brochure-RELY_TSN_PCIe.pdf, (accessed on 10.01.2024).

[216] *Time-sensitive networking switch 1g brochure*, [Online]. Available: https://www.relyum.com/web/wp-content/uploads/RELY-TSN-BRIDGE.pdf, (accessed on 10.01.2024).

[217] *Adva fsp 150-ge100pro series datasheet*, [Online]. Available: https://www.adva.com/-/media/adva-main-site/resources/data-sheets/pdfs/fsp-150-ge-100-pro-series.pdf?rev=-1&hash=766DDC8E3D4D886E43DC6AF49DF4CE7A, (accessed on 10.01.2024).

[218] *Intel ethernet controller i210 datasheet*, [Online]. Available: https://cdrdv2-public.intel.com/333016/333016%20-%20I210_Datasheet_v_3_7.pdf, (accessed on 08.01.2024).

[219] *Telit fn980-fn980m product brief*, [Online]. Available: https://www.telit.com/wp-content/uploads/2023/10/TC_FN980-FN980m_Product_Brief.pdf, (accessed on 12.01.2024).

[220] *Nokia airscale radio access*, [Online]. Available: https://www.nokia.com/networks/mobile-networks/airscale-radio-access/, (accessed on 24.01.2024).

[221] *Open5gs open source implementation*, [Online]. Available: https://github.com/open5gs/open5gs, (accessed on 24.01.2024).

UNIVERSIDAD DE MÁLAGA

[222] *E7515b uxm 5g wireless test platform*, [Online]. Available: https://www.keysight.com/es/en/assets/7018-06559/data-sheets/5992-3811.pdf, (accessed on 24.01.2024).
[223] *P416 language specification*, [Online]. Available: https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html, (accessed on 27.01.2024).
[224] *Behavioral model (bmv2) repository*, [Online]. Available: https://github.com/p4lang/behavioral-model/tree/main, (accessed on 08.02.2024).
[225] *Automadapt repository*, [Online]. Available: https://github.com/FLSchempp/AutomAdapt, (accessed on 08.02.2024).
[226] *Ptp4l - ptp boundary/ordinary clock*, [Online]. Available: https://linux.die.net/man/8/ptp4l, (accessed on 10.01.2024).
[227] *Phc2sys - synchronize two clocks*, [Online]. Available: https://linux.die.net/man/8/phc2sys, (accessed on 10.01.2024).
[228] *P4runtime specification*, [Online]. Available: https://p4.org/p4-spec/p4runtime/main/P4Runtime-Spec.html, (accessed on 08.02.2024).
[229] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NG-RAN; Architecture description, Technical Specification (TS) 38.401, v18.0.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.401/38401-i00.zip, (accessed on 02.02.2024).
[230] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; NR and NG-RAN Overall Description; Stage 2, Technical Specification (TS) 38.300, v18.0.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.300/38300-i00.zip, (accessed on 02.02.2024).
[231] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Radio Resource Control (RRC) protocol specification, Technical Specification (TS) 38.331, v18.0.0 Jan. 2024, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.331/38331-i00.zip, (accessed on 08.03.2024).
[232] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Physical channels and modulation, Technical Specification (TS) 33.211, v18.1.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.211/38211-i10.zip, (accessed on 08.03.2024).
[233] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Multiplexing and channel coding, Technical Specification (TS) 33.212, v18.1.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.212/38212-i10.zip, (accessed on 08.03.2024).
[234] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Physical layer procedures for control, Technical Specification (TS) 33.213, v18.1.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.213/38213-i10.zip, (accessed on 08.03.2024).
[235] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Physical layer procedures for data, Technical Specification

UNIVERSIDAD DE MÁLAGA

(TS) 33.214, v18.1.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.214/38214-i10.zip, (accessed on 08.03.2024).

[236] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Medium Access Control (MAC) protocol specification, Technical Specification (TS) 38.321, v18.0.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.321/38321-i00.zip, (accessed on 08.03.2024).

[237] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification, Technical Specification (TS) 36.321, v18.0.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/36_series/36.321/36321-i00.zip, (accessed on 08.03.2024).

[238] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; NR and NG-RAN Overall Description; Stage 2, Technical Specification (TS) 36.133, v18.4.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/36_series/36.133/36133-i40.zip, (accessed on 08.03.2024).

[239] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Radio Link Control (RLC) protocol specification, Technical Specification (TS) 38.322, v18.0.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.322/38322-i00.zip, (accessed on 08.03.2024).

[240] 3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network; NR; Packet Data Convergence Protocol (PDCP) specification, Technical Specification (TS) 38.323, v18.0.0 Jan. 2024, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.323/38323-i00.zip, (accessed on 08.03.2024).

[241] 3rd Generation Partnership Project (3GPP); Technical Specification Group Services and System Aspects; Security architecture and procedures for 5G system, Technical Specification (TS) 33.501, v18.4.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/33_series/33.501/33501-i40.zip, (accessed on 08.03.2024).

[242] 3rd Generation Partnership Project (3GPP), Technical Specification Group Radio Access Network; E-UTRA and NR; Service Data Adaptation Protocol (SDAP) specification , Technical Specification (TS) 37.324, v17.0.0 Mar. 2022, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/37_series/37.324/37324-h00.zip, (accessed on 08.03.2024).

[243] 3rd Generation Partnership Project (3GPP); Technical Specification Group Core Network and Terminals; Non-Access-Stratum (NAS) protocol for 5G System (5GS); Stage 3, Technical Specification (TS) 24.501, v18.5.0 Dec. 2023, [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/24_series/24.501/24501-i50.zip, (accessed on 02.02.2024).

UNIVERSIDAD DE MÁLAGA

[244] *Apache kafka: An open-source distributed event streaming platform*, [Online]. Available: https://kafka.apache.org/, (accessed on 24.01.2024).

[245] *Influxdb: Real-time analytics platform*, [Online]. Available: https://www.influxdata.com, (accessed on 24.01.2024).

[246] *Pcap4j java library repository*, [Online]. Available: https://github.com/kaitoy/pcap4j, (accessed on 24.01.2024).

[247] *Pandas dataframe documentation*, [Online]. Available: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html, (accessed on 01.05.2024).

[248] L. Panizo and M.-d.-M. Gallardo, "STAn: analysis of data traces using an event-driven interval temporal logic," *Automated Software Engineering*, vol. 30, Nov. 2022. DOI: 10.1007/s10515-022-00367-5.

UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD
DE MÁLAGA

Appendix A

Resumen en Español

A.1 Introducción

A.1.1 Motivación

La competencia en la industria está creciendo en los últimos años debido a la demanda de productos que tengan un cierto nivel de calidad y fiabilidad sin dejar de ser rentables. La automatización industrial se está convirtiendo en un pilar fundamental en el contexto de la Industria 4.0, y muchas aplicaciones presentan nuevos retos que deben ser abordados. Los protocolos y estándares industriales [1], como PROFINET y OPC, se utilizan comúnmente para mejorar las redes Ethernet, garantizando una comunicación fiable entre los dispositivos. El grupo de trabajo 802.1 del IEEE ha desarrollado una serie de estándares conocidas como Time-Sensitive Networking (TSN) para garantizar una conectividad determinista (con latencia acotada, bajo jitter y una mínima pérdida de paquetes) en las redes Ethernet tradicionales. Características como la sincronización en tiempo y el tráfico programado son dos de los principales componentes que garantizan la calidad de servicio de estas comunicaciones tan exigentes. Por lo tanto, estos mecanismos son necesarios para cumplir los requisitos de comunicación en entornos cableados, como puede ser una fábrica.

La integración de los estándares comentados anteriormente con las tecnologías de comunicación inalámbrica permite a las redes industriales aprovechar las ventajas de estas tecnologías. Las tecnologías inalámbricas pueden complementar las redes industriales, permitiendo ventajas como la accesibilidad, la flexibilidad y la movilidad. Sin embargo, debido a los estrictos requisitos de las aplicaciones, no es factible sustituir por completo las soluciones cableadas en entornos industriales en la actualidad. Esto ha dado lugar a la aparición de redes híbridas, que son una combinación de tecnologías cableadas e inalámbricas. Principalmente, existen dos tecnologías inalámbricas que están impulsando el desarrollo de redes TSN híbridas: IEEE 802.11 y 5G. La primera se ha propuesto como tecnología potencial para TSN inalámbricas por su sincronización de reloj precisa y su

UNIVERSIDAD DE MÁLAGA

programación parcial del tráfico [2]. Sin embargo, no puede garantizar una comunicación determinista, ya que no protege contra las interferencias de otros usuarios o servicios radio que operan en la misma banda de frecuencias. Por otro lado, la tecnología 5G es la principal candidata para implantar redes TSN inalámbricas, ya que ofrece mayor calidad al usar espectros reservados. Las principales técnicas y procedimientos relacionados con la integración de TSN sobre 5G se describen en [3].

La combinación de las tecnologías mencionadas conlleva una serie de retos que deben tenerse en cuenta (por ejemplo, el desarrollo de nuevas entidades con capacidad de traducción entre ambos dominios). Además, hay que tener en cuenta que en una red pueden coexistir un número considerable de patrones de tráfico diferentes, cada uno con sus propios requisitos (por ejemplo, tolerancia máxima al retardo o a las pérdidas de paquetes). Por lo tanto, la tarea de configurar manualmente cada sesión de tráfico para que cumpla dichos requisitos resulta imposible. Existen técnicas como Closed Control Loop (CCL) y Zero-Touch Network and Service Management (ZSM) que pretenden controlar y configurar la red con una intervención humana mínima o, idealmente, sin intervención humana.

A.1.2 Preguntas de Investigación y Objetivos

+ **Pregunta de Investigación 1 (PI1)**: ¿Son compatibles los equipos existentes en la actualidad con la integración de redes TSN en el sistema 5G?
+ **Pregunta de Investigación 2 (PI2)**: ¿Qué métodos de automatización pueden utilizarse para configurar la red 5G con el fin de cumplir los requisitos del tráfico TSN?
+ **Pregunta de Investigación 3 (PI3)**: ¿Cómo puede mejorarse la explicabilidad del sistema para garantizar una comprensión más clara de su funcionamiento?

Objetivos principales

- **Objetivo 1 (O1)**: Demostrar la viabilidad de TSN sobre redes 5G mediante el desarrollo de un prototipo de las entidades y procedimientos necesarios para la traducción de tráfico entre ambos dominios.
- **Objetivo 2 (O2)**: Demostrar la capacidad de construir automáticamente modelos basados en autómatas de aprendizaje para la auto-configuración eficiente de una red industrial 5G.
- **Objetivo 3 (O3)**: Diseñar e implementar algoritmos automatizados de reconfiguración de redes 5G industriales a partir de los modelos construidos para garantizar los requisitos impuestos por el tráfico determinista en tiempo real (TSN) para diferentes perfiles de tráfico, usuarios y condiciones extremo a extremo.

UNIVERSIDAD DE MÁLAGA

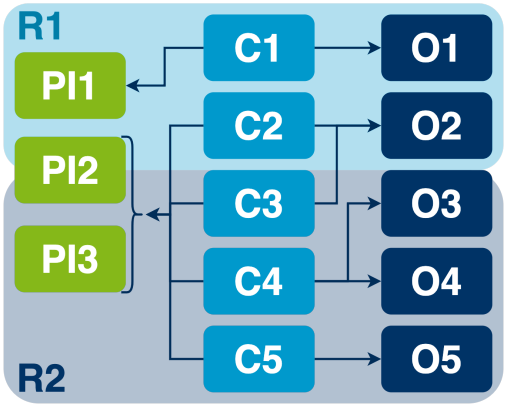


Figure A.1: Relación entre las preguntas de la investigación, objetivos y contribuciones

A.1.3 Objetivos secundarios

- Objetivo 4 (O4): Creación de un catálogo de modelos de red 5G que incluya configuraciones de red 5G adecuadas para un conjunto de perfiles de tráfico TSN sobre 5G.
- Objetivo 5 (O5): Diseño de una herramienta que integre tanto los algoritmos como los módulos necesarios para reconfigurar la red 5G sin intervención humana.

A.1.4 Contribuciones

Esta tesis doctoral se centra en los objetivos definidos anteriormente para abordar las preguntas de investigación. Durante el desarrollo del trabajo de tesis se han obtenido una serie de contribuciones que se explican a continuación. La Figura A.1 presenta la relación entre las preguntas de la investigación, objetivos y contribuciones de esta tesis.

- **Contribución 1 (C1): Construcción de un banco de pruebas (testbed) para la validación de TSN sobre 5G como prueba de concepto.**

  PI1 explora el estado actual de la integración de ambas tecnologías, TSN y 5G. Para ello, se ha realizado un análisis de trabajos relacionados y lo descrito en los estándares de 3GPP [28], incluyendo las entidades y procedimiento asociados. Por lo tanto, la construcción del testbed que permita la validación de la red 5G usando diferentes perfiles de tráfico TSN permite abordar **PI1** y **O1**.

+ **Contribución 2 (C2): Diseño de algoritmos para construir automáticamente un modelo de red 5G a partir de trazas de red (autómatas de aprendizaje).** El objetivo de **PI2** es estudiar y determinar el enfoque adecuado para lograr la reconfiguración automática de la red 5G en función de los requisitos del tráfico TSN. Además, en la **PI3** se persigue la investigación de técnicas y enfoques de aprendizaje para lograr la reconfiguración de la red.

UNIVERSIDAD DE MÁLAGA

Se desarrolla un algoritmo para construir un modelo que representa el comportamiento de la red 5G para hacer frente a **PI2** y **O2**. El modelo está representado por un autómata de aprendizaje que consta de estados de red para diferentes perfiles de tráfico TSN con requisitos específicos, que se centra en **PI3**. Cada estado está compuesto por la configuración de la red y los KPI del tráfico. Además, el autómata se crea automáticamente a partir de trazas de red siguiendo el concepto de aprendizaje pasivo.

+ **Contribución 3 (C3): Extensión del algoritmo de construcción automática de modelos de red 5G (C2) a partir de tráfico en tiempo real (extensión de autómatas de aprendizaje).** Siguiendo el trabajo de **C2**, se define un nuevo tipo de autómata llamado Traffic Oriented input/output Automata (TOA), que contribuye directamente a **O2** y permite representar la evolución del comportamiento de la red en relación con el tiempo. Además, el algoritmo se ha ampliado para generar TOA basado en el tráfico TSN a través de 5G, ya sea a partir de trazas de red capturadas previamente o monitorización en tiempo real, donde el autómata puede interactuar parcialmente con el sistema. En consecuencia, este desarrollo responde a **PI2** y **PI3**.

+ **Contribución 4 (C4): Diseño de algoritmos para la predicción y reconfiguración de redes 5G.** Para finalizar la respuesta de **PI2** y abordar **PI3**, es imprescindible el desarrollo de un algoritmo de predicción y reconfiguración automática que pueda aprovechar la información aprendida sobre el comportamiento de la red 5G. Por lo tanto, este algoritmo tiene como objetivo explotar TOA para predecir posibles desviaciones de los KPI observados hacia valores de calidad no deseados. El objetivo es proponer de antemano una configuración de red alternativa basada en un TOA aprendido para evitar estas desviaciones, que es el objetivo principal de **O3**. El uso de diferentes TOAs aprendidos conduce a la producción de un catálogo que incluye los modelos y configuraciones de red 5G, que son factibles para diversos perfiles de tráfico TSN sobre 5G (**O4**).

+ **Contribución 5 (C5): Implementación de una herramienta que permita reconfigurar automáticamente la red 5G para satisfacer las necesidades de tráfico, incluyendo algoritmos diseñados (C2, C3 y C4) y capaces de ejecutarse en el banco de pruebas (C1).** La implementación de una herramienta capaz de integrar los algoritmos desarrollados, así como los módulos necesarios (**O5**) es vital para alcanzar el objetivo principal de esta tesis. El estudio de las relaciones discretas, eventos temporales o procesos estocásticos que pueden ocurrir en la red es esencial para dar respuesta a **PI2** y **PI3**. Primero, la implementación de la herramienta permite capturar esta información a partir de trazas de red previamente capturadas (**C2**) para construir el modelo de red 5G. Segundo, la herramienta se amplió para capturar directamente el tráfico que está siendo enviado, permitiendo que el modelo de red 5G pueda construirse en tiempo real (**C3**). En ambos casos, los modelos de red 5G pueden utilizarse para la reconfiguración de la red (**C4**). Además, la herramienta puede ejecutarse en el

UNIVERSIDAD DE MÁLAGA

banco de pruebas (**C1**), permitiendo una completa integración en el sistema.

Estas contribuciones han dado lugar a la publicación de dos revistas [34], [35] que contienen los aspectos clave del trabajo realizado durante la tesis. La primera revista (**R1**) contiene los resultados del trabajo realizado durante la primera iteración de la metodología de esta tesis, abordando las **PI1** y **PI2**, además de las contribuciones y objetivos **C1**, **C2**, **O1** y **O2**, respectivamente. La segunda revista (**R2**) resulta del trabajo realizado en la segunda iteración de la metodología, siendo una continuación de la primera, centrándose principalmente en responder a **PI2** y **PI3**, mientras que se abordan las contribuciones **C3**, **C4**, **C5**y los objetivos **O3**, **O4** y **O5**.

A.2 Antecedentes

A.2.1 Tecnologías Avanzadas de Redes

Las *Redes Deterministas* son un intento del grupo de trabajo Internet Engineering Task Force (IETF) Deterministic Networking (DetNet) para implementar comunicaciones deterministas para aplicaciones de tiempo real como la automatización industrial [4], [5]. Esto implica características como latencia acotada, tasas de pérdida de datos y jitter (fluctuación de latencia) muy bajas. Para abarcar las características que demandaba el mercado nació IEEE 802.1 **Time-Sensitive Networking** (TSN), habilitando la transmisión de datos sensibles al tiempo sobre las redes Ethernet tradicionales compartidas por aplicaciones con requisitos de calidad de servicio diferentes, donde se diferencian el tráfico TSN (crítico) y no TSN (no crítico). TSN garantiza las características perseguidas por las redes deterministas como la latencia acotada, jitter bajo y una mínima pérdida de paquetes, así como diferentes técnicas que permiten alojar recursos para priorizar y garantizar el tráfico crítico. Además, dispone de una sincronización de tiempo precisa como parte fundamental para garantizar estas características. Por lo tanto, TSN se puede considerar un conjunto de estándares o herramientas que permiten satisfacer los requisitos de tráfico sensible al tiempo [8]. La Figura A.2 representa la arquitectura centralizada TSN, donde los dispositivos finales TSN envían y reciben tráfico (conocidos como talker y listener, respectivamente) a través de los bridges TSN, que son switches particulares que implementan los diferentes estándares de TSN para garantizar la transmisión de los flujos de tráfico. Además las entidades Centralized Network Configuration (CNC) y Centralized User Configuration (CUC) se encargan de la comunicación con los dispositivos finales con el fin de aplicar una configuración que cumpla con los requisitos del tráfico, destacando el periodo típico (tiempo entre paquetes), tolerancia al jitter y a las pérdidas y tamaño de paquete.

La quinta generación de **redes móviles 5G** se presenta como una tecnología habilitadora para nuevos sectores como la automoción, automatización industrial o incluso para aplicaciones críticas en el tiempo. Las redes 5G ofrece altas velocidades de datos y baja latencia entre otras características, permitiendo la mejora considerable de la calidad de servicio, y consecuentemente la calidad de experiencia que recibe el usuario final.

UNIVERSIDAD DE MÁLAGA

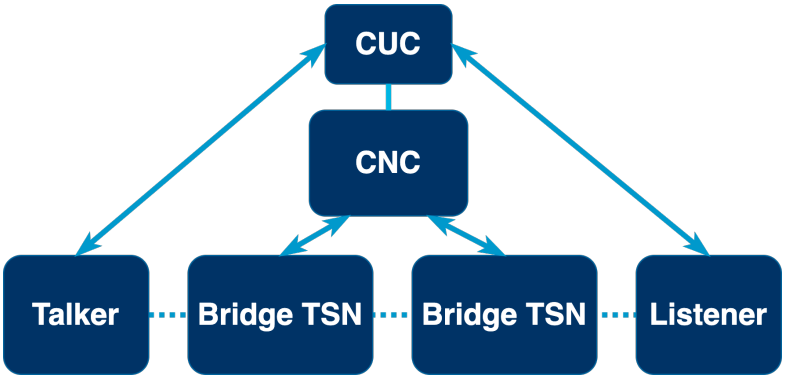


Figure A.2: Arquitectura de red TSN totalmente centralizada - IEEE 802.1Qcc [9]

La arquitectura de red 5G es similar a la de tecnologías anteriores (4G), la cual está principalmente compuesta por el dispositivo móvil (UE), la red de acceso radio (RAN) y el núcleo de red (5G Core o 5GC), tal y como se muestra en la Figura A.3. Esta arquitectura se conoce como 5G standalone (5G SA), sin embargo, existe una arquitectura adicional construida sobre la infraestructura 4G conocida como 5G non-standalone (5G NSA). Concretamente se usa el núcleo de red 4G (conocido como EPC) y la red de acceso radio de ambas tecnologías, 4G (conocido como eNodeB) como anclaje y 5G (conocido como gNodeB) para la transmisión de datos. Para el desarrollo de esta tesis se ha usado la tecnología 5G SA, que dispone de nuevas características y capacidades. En particular, en el núcleo de red 5G destacan las siguiente funciones de red (FR): AF (Application Function), que provee un papel esencial en la gestión de la calidad de servicio; AMF (Access and Mobility Management Function) para el control del usuario y la mobilidad; SMF (Session Management Function) para la gestión de la sesión de tráfico; UPF (User plane Function) para el manejo del plano de datos del usuario.

El 3GPP ha avanzado en lo que respecta a la combinación de estas dos tecnologías, dando lugar a **TSN sobre 5G**, aprovechando los beneficios de 5G (redes inalámbricas) en la Industria 4.0 (por ejemplo, para la automatización). Concretamente, desde Release 16 [28] en adelante, 3GPP está trabajando para definir una arquitectura, donde la red 5G (llamada 5GS) actúe como un bridge TSN de forma transparente a los dispositivos finales (DF) TSN, que en este caso se conoce como bridge 5GS, como se representa en la Figura A.4. En esta arquitectura se incluyen nuevas entidades especializadas en dar soporte a las comunicaciones sensibles al tiempo sobre la red 5G. Por un lado, los *traductores TSN*, Device-Side Time-Sensitive Networking Translator (DS-TT) junto al dispositivo móvil (UE) y Network-Side Time-Sensitive Networking Translator (NW-TT) junto al UPF se encargan de la interoperabilidad de ambos sistemas, TSN y 5GS, incluyendo los planos de control y usuario. Por otro lado, el *TSN AF* es la función de red específica del núcleo de red 5G responsable de la comunicación entre el 5GS y la red TSN (concretamente con la entidad CNC), intercambiando información con el fin de configurar de forma apropiada la red para la correcta integración.

UNIVERSIDAD DE MÁLAGA

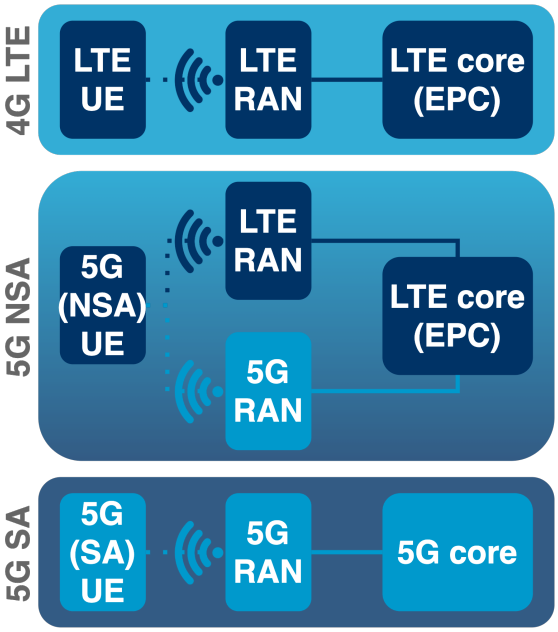


Figure A.3: Evolución de la arquitectura: 4G, 5G NSA y 5G SA

A.2.2 Técnicas y Métodos para Reconfiguración

La integración de ambas tecnologías trae consigo una serie de retos que deben ser considerados. Por una lado, el desarrollo de los traductores TSN que se ubican en los extremos de la red 5G y la sincronización de tiempo a lo largo de todo el sistema. Por otro lado, el desarrollo de un TSN AF que sea capaz de configurar la red. Además, esta configuración debe cumplir en todo momento con los requisitos del tráfico que esté siendo enviado entre los dispositivos finales TSN. Por lo tanto, el TSN AF será el encargado de configurar de forma dinámica la red 5G con este objetivo.

Actualmente existen varias técnicas y métodos con un enfoque similar. Primero, las **redes auto-organizadas** (SON) es un concepto que engloba varias funciones principales: auto-configuración, auto-curación, y auto-optimización. La idea es que la red sea capaz de actuar de forma autónoma ante determinados problemas. Por ejemplo, gestión de la capacidad de cobertura entre celdas vecinas dependiendo del número de usuarios conectados. Segundo, conceptos como **bucles de control en lazo cerrado** (CCL) para ajustar de forma autónoma un sistema con el fin de mantener un estado específico sin necesidad de intervención humana en base a la información recibida del sistema; o **redes basadas en intenciones** (IBN) con el objetivo de capturar la intención del usuario y transformarla en políticas que pueden automatizarse y aplicarse de forma continua en la red. Por último, uno de los conceptos fundamentales es **Zero-Touch Network** (ZSM), que pretende ir un paso más allá y conseguir una red completamente autónoma, no solo en problemas concretos, si no en todos los aspectos sin ningún tipo de intervención humana. Este enfoque es justo el que se persigue en el caso de la configuración de la red

UNIVERSIDAD DE MÁLAGA

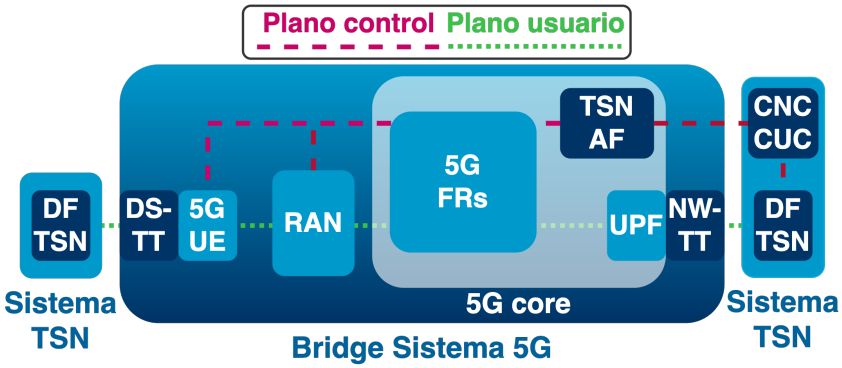


Figure A.4: Arquitectura Time-Sensitive Networking sobre 5G

5G para cumplir con los requisitos del tráfico TSN. Debido a la complejidad del sistema y de la red 5G, esta configuración implica la monitorización de una gran cantidad de información para aplicar los valores necesarios de determinados parámetros. Por otro lado, teniendo en cuenta la inmensa cantidad de parámetros que pueden ser configurados en una red 5G, la combinación de diferentes técnicas es esencial para aplicar este tipo de reconfiguraciones de forma automática. Además, es imprescindible el uso de técnicas de aprendizaje para afinar estas configuraciones en base a la información monitorizada y de los parámetros que pueden ser ajustados para conseguir el cumplimiento de los requisitos.

A.2.3 Métodos de aprendizaje

Con el objetivo que se persigue, se presentan técnicas de aprendizaje con diferentes enfoques para crear representaciones o modelos que incluyen observaciones como acciones, eventos, e interacciones que describen el comportamiento del sistema. Estas observaciones se pueden usar para aprender y tomar decisiones relacionadas con la gestión del sistema. Por ejemplo, un modelo de red 5G puede utilizarse para supervisar la calidad de las sesiones de tráfico de extremo a extremo y aplicar una reconfiguración de la red para cumplir los requisitos a lo largo del tiempo de las distintas sesiones de tráfico.

Para ello, los modelos de comportamiento permiten representar de forma abstracta el comportamiento y los aspectos dinámicos de un sistema. Esto resulta en un diagrama formado por un conjunto de estados y transiciones que puede ser usado en diferentes ámbitos. Por un lado, existen técnicas de aprendizaje como Inteligencia Artificial (AI) y Machine Learning (ML) que pueden ser utilizadas para analizar datos de la red y realizar predicciones de posibles problemas, incluso permitiendo optimizar diferentes parámetros a la hora de construir un modelo. Sin embargo, los algoritmos usados por los métodos de AI/ML mantienen la conciencia del estado durante su ciclo de operación. Por lo tanto, los estados intermedios no se representan de forma explícita en el modelo resultante, haciendo difícil explicar las decisiones llevadas a cabo (por ejemplo, una reconfiguración de red). Aunque el concepto de IA explicable (XAI) nace para intentar poner solución a este problema, intentando justificar las decisiones y acciones tomadas, aún existen limitaciones

UNIVERSIDAD DE MÁLAGA

y retos que necesitan ser solventados, especialmente para comprender el comportamiendo de sistemas complejos como el caso de una red TSN sobre 5G. Por otro lado, los autómatas de aprendizaje (o aprendizaje con modelos) permiten construir un diagrama de estados a partir de un sistema no conocido (caja negra), mediante dos enfoques principales: el enfoque activo, construyendo el modelo a través de consultas directas al sistema; y el enfoque pasivo, mediante observaciones del sistemas, sin una interacción directa. Además, en contraste a los métodos de AI, los autómatas de aprendizaje permiten la construcción de un modelo formal que puede ser analizado formalmente en relación al original, y sus decisiones se pueden justificar matemáticamente. Por lo tanto, esta tesis adopta el enfoque de autómatas de aprendizaje para representar la evolución del sistema TSN sobre 5G, permitiendo la explicación lógica y la comprensión de cada decisión tomada.

A.3 Implementación del Testbed TSN sobre 5G

En esta sección se describe el testbed desarrollado en la Universidad de Málaga como parte de esta tesis, donde la tecnología TSN se ha integrado con la red 5G. Se detallan los diferentes elementos del testbed, destacando los que están destinados a la integración de ambas tecnologías, es decir, los traductores TSN (DS-TT y NW-TT) y el TSN AF. Concretamente, el TSN AF se especifica como *AutomAdapt*, y se compone de una serie de herramientas implementadas durante el desarrollo de esta tesis, explicadas en detalle en la Sección A.5. Además, se describen las soluciones propuestas a los retos de integración previamente mencionados.

A.3.1 Componentes del Testbed

El testbed TSN sobre 5G se presenta en la Figura A.5. Primero, incluye los componentes de una red TSN clásica: dispositivos finales TSN *Relyum* [215], bridge TSN *Relyum* [216], las entidades software CNC/CUC, y el equipamiento de sincronización de tiempo (conocidos como relojes grand master) *ADVA* [217]. Segundo, los componentes de la red 5G: dispositivo móvil (UE) Telit fn980m [219], la red de acceso radio (RAN) y el núcleo de red 5G (5GC). Actualmente existen dos opciones para la red 5G: la primera se compone de una RAN de *Nokia* [220] y un núcleo de red *Open5GS* [221]; la segunda se trata de la plataforma de pruebas inalámbricas *Keysight* UXM 5G [222]. Por último, los elementos fundamentales para la integración de TSN y 5G, los traductores TSN y el TSN AF para que la red 5G pueda actuar como bridge TSN desde la perspectiva de los dispositivos finales TSN.

Una versión inicial de los **traductores TSN** (DS-TT y NW-TT) se ha implementado usando el lenguaje de programación P4 [223]. Realizan algunas de los procedimientos necesarios para interconectar los dominios TSN y 5G, como gestionar las cabeceras 802.1Q, eliminándolas de los paquetes antes de ser enviados por la red 5G y añadiéndolas una vez el paquete ha llegado al otro extremo. Además, el **TSN AF** como software encargado del control y configuración de la red 5G. Intercambia información relevante con la red TSN y guarda información relacionada con las conexiones extremo a extremo

UNIVERSIDAD DE MÁLAGA

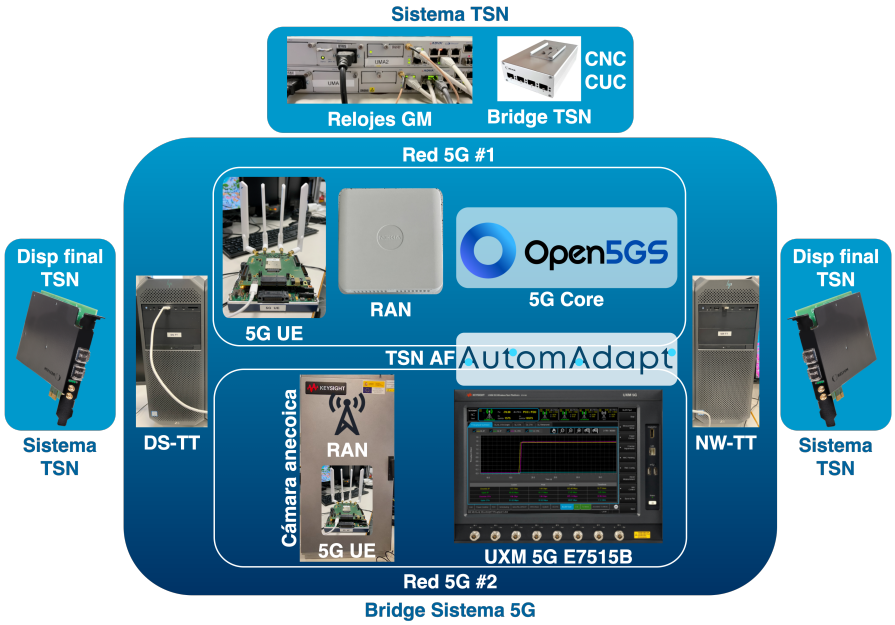


Figure A.5: Testbed TSN sobre 5G en la Universidad de Málaga

establecidas (por ejemplo, el mapeo de calidad de servicio definido entre ambas redes). Debido a su complejidad, su desarrollo ha sido una de las contribuciones principales de esta tesis y todos los detalles de su enfoque, e implementación junto con su evaluación se introducen en las Secciones A.4 y A.5, respectivamente.

A.3.2 Retos y Parámetros 5G Relevantes

Como se introdujo previamente, existen ciertos **retos** que trae consigo la integración de las tecnologías TSN y 5G. En la Figura A.6 se presentan las soluciones principales de éstos, que son la sincronización de tiempo y la traducción del tráfico entre dominios, los cuales deben ser implementados en los traductores TSN.

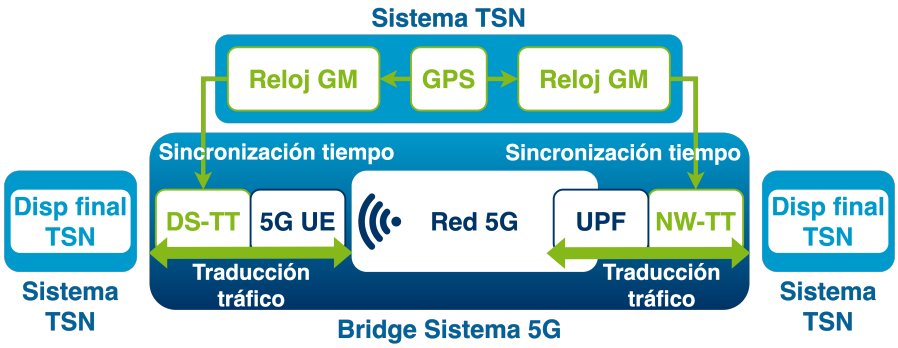


Figure A.6: Solución propuesta para sincronización de tiempo y traducción del tráfico

UNIVERSIDAD DE MÁLAGA

Por un lado, la **sincronización de tiempo** precisa es vital para el correcto funcionamiento del sistema. Los mecanismos y procedimientos para implementarla se introducen en Release 16 [28], donde se contempla que la red 5G se comporte como un sistema temporizado (IEEE 802.1AS). Actualmente el testbed es compatible con Release 15+ y no soporta la sincronización nativa descrita en Release 16. Para ello, se propone una solución alternativa, donde los traductores TSN reciben los paquetes de sincronización desde dos relojes grand master, ambos sincronizados con una única señal GPS. La diferencia de sincronización que introducen ambos relojes se considera insignificante tras haber sido medida con un osciloscopio. Por lo tanto, se asume que ambos traductores TSN tienen la misma señal de sincronización y consecuentemente están sincronizados. Por otro lado, la **traducción del tráfico** es otra parte fundamental para el intercambio de mensajes entre dominios. Con este fin, el lenguaje P4 ofrece la programabilidad para la gestión de los paquetes necesarios. En el ejemplo mencionado anteriormente, donde se añadían o eliminaba la cabecera 802.1Q es importante destacar que los detalles específicos de cada sesión de tráfico pueden variar entre sesiones diferentes. Por lo tanto, las acciones a aplicar deben ser diferentes y adaptarse de forma dinámica. Ya que P4 es un lenguaje que funciona a través de reglas establecidas en unas tablas coincidencia-acción, estas reglas deben ser adaptadas dinámicamente. Para este propósito, P4Runtime [228] permite su gestión (añadir, eliminar y modificar) en tiempo real, habilitando la flexibilidad necesaria para este caso donde las reglas pueden ser gestionadas de forma automatizada.

Teniendo en cuenta que el objetivo de la tesis es configurar dinámicamente la red 5G para cumplir los requisitos del tráfico TSN, es fundamental conocer que tipo de **parámetros de la red 5G** pueden ser modificados. Específicamente, aquellos que sean más relevantes para este tipo de tráfico y puedan modificarse en tiempo real, es decir, mientras la sesión de tráfico está activa y no conlleve una reinicio total o parcial del sistema. Existen dos grupos principales de parámetros que se pueden modificar en una red 5G, los pertenecientes al núcleo de red (5GC) y los de la red de acceso radio (RAN).

Los parámetros del 5GC definidos por los estándares del 3GPP [26] a destacar se pueden categorizar en varios grupos, a continuación se presentan algunos ejemplos destacados:

- **Parámetros QoS**: Identificador de calidad de servicio 5G (5QI) usado para aplicar reglas de calidades de servicio individuales dentro de una sesión de tráfico (conocida como sesión PDU). También existe el Identificador de flujo QoS (QFI) para identificar diferentes flujos dentro de la misma sesión y la Asignación y Prioridad de retención (ARP) para determinar el nivel de prioridad a cada flujo.
- **Parámetros de la portadora**. Una portadora es un canal de comunicación que establece conectividad extremo a extremo entre el UE y el 5GC. Principalmente existen dos tipos, la portadora por defecto para la comunicación inicial y la portadora dedicada (establecida sobre la de por defecto) para tráfico específico como aplicaciones de comunicación en tiempo real. Existen otros parámetros como el bit rate máximo (MBR) o el bit rate garantizado (GBR).
- **Parámetros de direccionamiento del tráfico**. Dentro de las sesiones PDU

UNIVERSIDAD DE MÁLAGA

existen varios tipos: IPv4, Ethernet o sin estructurar. El UPF se encarga de gestionar el plano de usuario e interconectar el 5GC con la red de datos (DN) exterior. Por lo tanto, su ubicación puede influenciar en la calidad de servicio ofrecida.

- **Parámetros de movilidad**. Existen parámetros críticos en relación a la movilidad dentro de la red para permitir procesos de handover (cambiar la conectividad entre estaciones base) de forma eficiente y sin cortes.
- **Parámetros de temporizadores**. Existen algunos temporizadores que son vitales para procedimientos como el inicio o actualización del estado de la conexión de un UE.

De forma similar, 3GPP [229] define parámetros dentro de la red de acceso radio 5G, los cuales se dividen principalmente en capas de la pila de protocolos 5G [230]. Algunos de los más relevantes que pueden ser modificados son:

- Capa física: Primera capa de la pila de protocolos que se encarga de la transmisión y recepción de los datos sobre el interfaz aire. Algunos parámetros son: el esquema de modulación y codificación (MCS) que se aplica, o el número máximo de transmisiones HARQ (peticiones de retransmisión en el caso de paquete dañado o perdido).
- Capa MAC. Responsable de gestionar recursos radio compartidos y la priorización del tráfico. Los parámetros principales son: retardo máximo de paquetes (delayTarget), sólo portadoras GBR, o prioridad dentro del canal lógico (schedulPrio) que se configura en base al valor de 5QI.
- Capa RLC. Tiene tres modos de funcionamiento: con confirmación (AM), sin confirmación (UM) y transparente (TM). Dependiendo del modo realiza funciones como detección de errores y duplicados, corrección de errores o segmentación y reensamblado. Debido a la relevancia de estos parámetros, se han utilizado como base para la mayor parte del trabajo de esta tesis, especialmente los asociados al modo AM: número máximo de retransmisiones (ARQ) permitidas (maxRetxThreshold), número de bytes entre solicitudes de datos no confirmados (pollByte), tiempo entre solicitudes de datos no confirmados (pollPDU) o temporizador para reesamblaje (tReassembly).
- Capa PDCP. Gestión de la compresión y descompresión de las cabeceras, cifrado y descifrado. También se encarga de la transmisión de datos del plano de usuario y control. Algunos parámetros relevantes son: número máximo de PDUs que pueden ser retransmitidas (pdcpMaxNumRetxPdu), habilitar el descarte basado en búfer (pdcpBufDiscardEnabled) o la entrega fuera de orden (outOfOrderDelivery).
- Capa SDAP. Gestión de la asignación de la calidad de un flujo de datos específico dentro de una sesión PDU a un DRB correspondiente (ya establecido con la calidad adecuada). Existe una entidad SDAP por cada sesión PDU, asegurando el cumplimiento de los requisitos para los diferentes tipos de tráfico.

UNIVERSIDAD DE MÁLAGA

- Capa RRC. Controla los recursos radio (establecimiento, liberación, configuración y mantenimiento) y funciones de movilidad. Algunos de sus parámetros son: tiempo que el UE espera antes de volver a enviar un mensaje RRC para establecer la conexión (timerRejectWait) o tiempo antes de abortar algún procedimiento ya iniciado (timerRRCGuard).
- Capa NAS. Establecimiento, mantenimiento y liberación de sesiones entre el UE y el 5GC. También se encarga de procedimientos de autenticación, seguridad y registro.

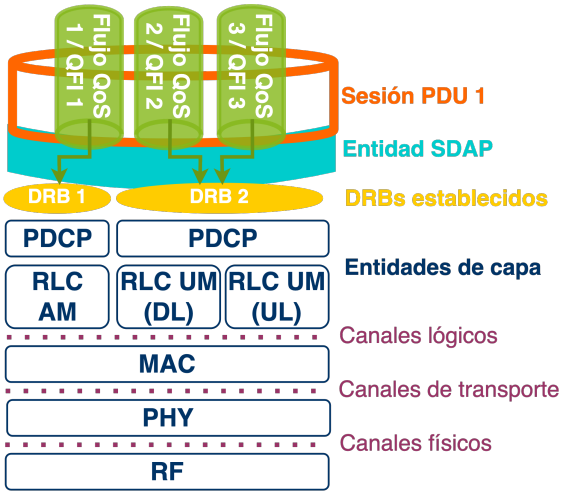


Figure A.7: Asignación de flujos QoS dentro de una sesión PDU en DRBs

A.3.3 Hallazgos clave

El testbed presentado anteriormente dispone de dos opciones de red 5G, que han sido usadas durante del desarrollo de la tesis. Durante la primera iteración de la metodología, el UXM 5G de Keysight (opción 2) fue usado para una operación manual de grano más fino en un entorno controlado gracias al uso de la cámara anecoica, permitiendo observar de forma más directa el efecto de los parámetros de la red 5G sobre los KPIs observados. Estas pruebas se centraron principalmente en la capa física, que puede llegar a tener un gran impacto en los resultados. Sin embargo, la modificación de estos parámetros resulta complicada en un entorno realista, ya que o bien no pueden ser modificados o se requiere un reinicio del sistema. Teniendo esto en cuenta, la segunda iteración de la metodología se centró en el uso de la RAN de Nokia junto con el núcleo de red Open5GS (opción 1). Tras realizar un estudio más profundo de los parámetros a modificar, se decidió centrarse en un conjunto reducido, ya que las posibles combinaciones dan lugar a un número inmanejable de configuraciones distintas. En particular, debido a que el objetivo de esta tesis es configurar la red 5G de forma automática para cumplir con los requisitos mientras que el tráfico TSN está siendo enviado, es fundamental que estos parámetros puedan ser modificados sin afectar al sistema, es decir, de forma dinámica.

UNIVERSIDAD DE MÁLAGA

Por un lado, las capas física y MAC se centran en la gestión de física y el acceso al entorno. Aunque existen algunos parámetros que pueden modificarse y que pueden influir en los resultados, ya existen algoritmos para modificar valores de la capa física de forma automática (por ejemplo, el índice de MCS), o están asociados a un tipo concreto de portadora en la capa MAC (no GBR o GBR). Por lo tanto, no se consideran estos parámetros con suficiente potencial para el contexto de esta tesis. Por otro lado, las capas RLC y PDCP se encuentran en un nivel más alto en la pila de protocolos, permitiendo un control adaptativo de la transmisión de datos. La capa RLC se encarga de funciones esenciales para proporcionar una transmisión de datos eficaz y fiable (por ejemplo, detección de errores y duplicados, segmentación y reensamblado), mientras que la capa PDCP tiene un rol importante en la seguridad de los datos (cifrado). Consecuentemente, las características de los parámetros de la capa RLC la hacen más adecuadas para alcanzar el objetivo de esta tesis. Específicamente, se ha elegido los parámetros de la capa RLC en el modo de funcionamiento AM, que permite incrementar la fiabilidad sin afectar a la latencia o jitter excesivamente.

A.4 Construcción Automática de Modelos con Autómatas de Aprendizaje

Esta sección presenta el enfoque seguido para perseguir el objetivo de aplicar reconfiguraciones automáticas en la red 5G en base a los requisitos del tráfico TSN que está siendo enviado. Para ello, se construye un modelo de la red 5G usando autómatas de aprendizaje, donde se aprende el comportamiento de la red, es decir, los indicadores de rendimiento o KPIs) mientras que diferentes perfiles de tráfico son usados con una amplia variedad de configuraciones 5G. Este enfoque se representa en la Figura A.8, donde los cuatro módulos de monitorización, aprendizaje, predicción y reconfiguración trabajan en un bucle cerrado. Centrándose en el aprendizaje, el trabajo se ha realizado en dos fases principales. La primera, siguiendo un enfoque de aprendizaje de autómata pasivo, en el que el autómata se induce a partir de un conjunto de datos dado, es decir, trazas de tráfico de red capturadas previamente. En la segunda fase, se expande este método, habilitando la capacidad de construir dicho autómata directamente desde la red 5G mientras se envía el tráfico, permitiendo representar la evolución del comportamiento de la red en base al tiempo, enriqueciendo la información aprendida, junto con la configuración de red usada en cada caso. En este caso, el autómata puede interactuar con la red de forma parcial.

A continuación, se describen de forma breve los cuatro módulos.

- **Monitorización**. Monitorización de la red 5G, incluyendo tráfico, KPIs extremo a extremo (por ejemplo, latencia y jitter), configuración de la red 5G y eventos como nuevas conexiones de tráfico junto con sus requisitos.
- **Aprendizaje**. Implementación del algoritmo de construcción del autómata de aprendizaje que representa el comportamiento de la red 5G frente a diferentes combinaciones de perfiles de tráfico y configuraciones de red, a partir de trazas de

UNIVERSIDAD DE MÁLAGA

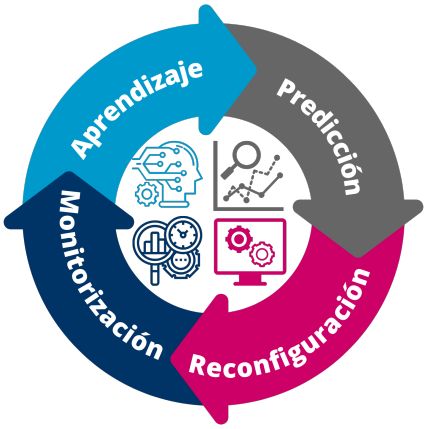


Figure A.8: Enfoque global

la red previamente capturadas o directamente desde la red en tiempo real.

- **Predicción**. Predicción de las posibles desviaciones de los KPI con respecto a los requisitos de tráfico utilizando el autómata de aprendizaje construido.
- **Reconfiguración**. Responsable de aplicar una reconfiguración de red cuando es necesario. Por ejemplo, cuando el módulo de predicción detecta una desviación de los KPIs, una nueva configuración para continuar cumpliendo con los requisitos se aplica.

A.4.1 Fase 1. Autómata de Aprendizaje Pasivo

Esta primera fase está centrada en la monitorización y captura de trazas de la red de forma previa, que serán aprendidas para crear el autómata de aprendizaje pasivo. Como es lógico, los módulos de predicción y reconfiguración no son relevantes en esta fase, al no poder actuar sobre el sistema en tiempo real.

Para **obtener trazas de red** usando el testbed presentado en la sección anterior, se genera tráfico TSN con unos determinados requisitos, mientras que la configuración 5G se modifica de forma pseudo-aleatoria durante un periodo de tiempo. Además, se identifican los eventos en relación al inicio y fin de la sesión de tráfico ($join(intd)?$ y $leave?$, respectivamente), y la modificación de la configuración de red 5G ($config(conf)?$). También se incluyen los resultados medidos del tráfico que se envía, es decir, los KPIs observados ($obsd$).

La **construcción del autómata de aprendizaje pasivo** se realiza mediante el algoritmo Learn, a partir de una colección finita de trazas $\mathcal{T}$, compuestas por una serie finita de estados observados s_i durante un periodo de tiempo en el que se envía tráfico y se cam-

UNIVERSIDAD DE MÁLAGA

bia la configuración de la red 5G. Los estados observados son tuplas $\langle ts, conf, intd, obsd \rangle$ con las marcas de tiempo, configuración de la red 5G y los KPIs (previstos y observados). Asimismo, el algoritmo aprende transiciones que consisten en los estados origen y destino junto con el evento que la desencadenó. El símbolo rd denota una desviación relevante en los valores observados con respecto a los previstos, donde la función $f_{\mathcal{A}}$ abstrae estos valores para indicar si están alineados, por encima o por debajo (usando 0, 1 o -1, respectivamente).

En resumen, el algoritmo Learn recibe una colección de trazas como entrada para construir el autómata capaz de reconocer los comportamientos representados en éstas a través de la función *Compose*, que se encarga de incorporar los comportamientos de cada traza, es decir, añadir los estados o transiciones necesarias. Existen varias funciones auxiliares que asisten a esta función. Debido a que el algoritmo se actualiza continuamente con nuevas trazas para capturar nuevos comportamientos, no es sencillo asegurar la convergencia. El tamaño del autómata depende del número de configuraciones aplicables y de los intervalos que definen los valores objetivo de las métricas, que al ser abstraídos no afectan a la terminación del algoritmo. Asumiendo que existe un número limitado de configuraciones [26], el autómata generado por el algoritmo Learn es finito.

A.4.2 Fase 2. Autómata de Entrada/Salida Orientado a Tráfico

Esta fase es la continuación de la anterior, en la cual se ha definido un nuevo tipo de autómata, llamado Traffic Oriented input/output Automata (TOA), que representa la evolución de una sesión de tráfico TSN con respecto al tiempo. También se mejora del algoritmo de aprendizaje definido previamente y se diseña un algoritmo para predicción y reconfiguración. En este caso, los cuatro módulos de la Figura A.8 se usan en bucle cerrado para habilitar la implementación de los ajustes necesarios en la red de forma autónoma.

En este contexto se pueden **obtener trazas de la red** directamente, es decir, mientras que se está enviando tráfico y se modifica la configuración de la red 5G, sin necesidad de guardarlas previamente, permitiendo obtener *estados concretos* que incluyen la configuración de la red 5G y los KPIs en tiempo real, lo cuales están alineados con las trazas usadas como entrada para que el módulo de aprendizaje sea capaz de generar el autómata (TOA) que represente el comportamiento de la red de una forma abstracta, lo cual es fundamental para crear TOA manejable y finito que pueda ser analizado durante la fase de predicción. La función qty realiza esta función de abstracción, condensando conjuntos de valores (rangos) previamente asignados para todos los KPIs que se monitorizan en valores individuales. La Tabla A.1 muestra un ejemplo de definición de valores de la función qty, donde los valores positivos muestran aquellos rangos considerados como válidos y los valores negativos las situaciones donde al menos un rango de los KPIs observados no es válido. Es importante destacar que la función qty está caracterizada para cada patrón de tráfico usado en la red.

De forma similar, la **construcción del autómata de aprendizaje** se produce mediante

UNIVERSIDAD DE MÁLAGA

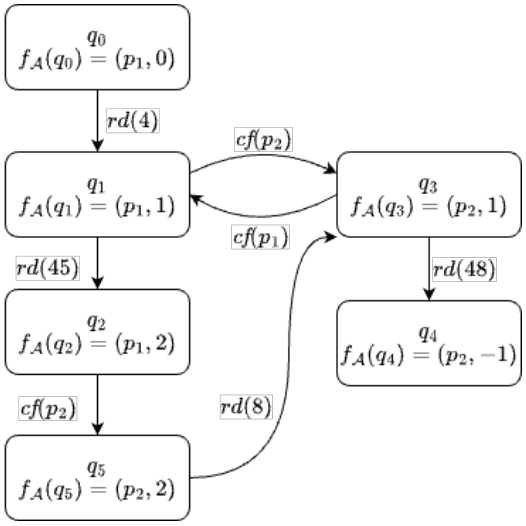


Figure A.9: Ejemplo de TOA generado

el algoritmo *Learn* usando un conjunto limitado de trazas de red $\mathcal{T}$, compuestas por una secuencia finita de estados de red observados generados por cambios de configuración de red 5G o cuando se alcanza un umbral de tiempo predefinido durante las sesiones de tráfico. Igualmente, el algoritmo añade nuevos estados y transiciones cuando son necesarias. Un estado observado de la red se describe como la tupla con formato $\langle ts, conf, \bar{o} \rangle$, con la marca de tiempo, la configuración de la red 5G y los KPIs observados. En este caso, se guardan las transiciones con los estados origen y destino, junto con la etiqueta de acción con los símbolos **cf**(*conf*) y **rd**(n) para representar un cambio de configuración o un cambio de calidad en los KPIs observados (indicando las n unidades de tiempo transcurrido). En este caso, los valores usados para la abstracción son los definidos por la función *qty*. Se muestra un ejemplo de TOA generado en la Figura A.9.

En conclusión, el algoritmo *Learn* usa la colección de trazas de entrada para construir el autómata que representa el comportamiento de la red con respecto al tiempo empleando

qty	*latencia* (ms)	*jitter* (ms)
-18	[15-20]	[15-20]
-8	[15-20]	[0-5]
-1	[0-5]	[15-20]
1	[0-5]	[0-1]
2	[0-5]	[1-5]
12	[5-15]	[0-1]
13	[5-15]	[1-15]

Table A.1: Ejemplo de valores de la función *qty* para un patrón de tráfico determinado

UNIVERSIDAD DE MÁLAGA

la función *Compose* para extender progresivamente TOA integrando los estados y transiciones de cada traza de red. Del mismo modo, varias funciones auxiliares se utilizan para esta tarea. La convergencia del algoritmo se basa en el número de estados creados, por lo tanto, al igual que en la fase anterior, asumiendo que el número de configuraciones y los valores abstractos de los KPIs son limitados, se obtiene un autómata finito. Cabe destacar que se produce un autómata que define el comportamiento de la red durante una sesión de tráfico para un patrón de tráfico particular. Dado que los potenciales perfiles de tráfico están acotados [23] y éste no cambia durante una sesión, nuestro enfoque consiste en crear una serie de autómatas únicos (para cada patrón de tráfico diferente), que pueden ser usados durante las etapas de predicción y reconfiguración.

Por otro lado, esta fase incluye el algoritmo *Predict*, diseñado para **predecir las desviaciones** de los KPIs observados con respecto a los esperados y **sugerir reconfiguraciones** de red basadas en un TOA aprendido. Para ello, se analizan las transiciones de tipo **rd**(n) del autómata y se reporta la necesidad aplicar una reconfiguración de la red (si es posible) antes de la degradación que ocurre a las n unidades de tiempo aprendida previamente. Este algoritmo reporta diferentes respuestas comprobando el siguiente estado: cumple la calidad esperada (*Continue*); la red necesita una reconfiguración, sugiriendo una lista de posibles configuraciones para seguir cumpliendo los requisitos del tráfico ($Reconf(List(conf))$); la red necesita una reconfiguración, pero no se puede sugerir ninguna configuración alternativa (*Warning*); el estado no es reconocido por el autómata (*Unknown*), lo que lleva a la terminación del algoritmo. Este algoritmo también incluye funciones auxiliares para su operación.

A.5 Implementación y Evaluación de la Propuesta

Este capítulo presenta los detalles de la implementación de la propuesta. Concretamente, se detallan los aspectos clave de la herramienta AutomAdapt (código disponible en [225]), que incluye los módulos de monitorización, aprendizaje, predicción y reconfiguración. Además, se realiza una evaluación de esta herramienta sobre el testbed descrito anteriormente.

A.5.1 Implementación

Monitorización

El módulo de monitorización se encarga de recopilar datos de distintos puntos de la red, incluyendo la información necesaria para la operación del resto de módulos. Aunque esta información se centra en los dispositivos finales TSN con el fin de obtener KPIs extremo a extremo.

Principalmente, está compuesto por Apache Kafka [244] para recoger los datos del tráfico y de los diferentes eventos. Dos programas escritos en Java (Talker y Listener) se encargan de capturar los paquetes usando la librería Pcap4J [246] en estos dispositivos, extraer la información relevante (por ejemplo, timestamp y tamaño del paquete) e insertarla de

UNIVERSIDAD DE MÁLAGA

forma organizada en topics de Kafka. Otro programa en Java (KPI generator) se encarga de procesar esta información para generar los KPIs (throughput, latencia, jitter y pérdida de paquetes) mediante los cálculos necesarios. Finalmente, estos KPIs se exportan a una base de datos de Influx [245], donde se pueden representar de forma visual mediante diferentes tipos de gráficas. Además, la configuración de la red 5G, como los parámetros de la radio, también se guardan en la base de datos.

Aprendizaje

El módulo de aprendizaje es responsable de crear autómatas, donde cada autómata presenta el comportamiento de la red con un patrón de tráfico específico durante una sesión TSN. El algoritmo *Learn* define como generar estos autómatas a partir de trazas de red. Este módulo se centra en la construcción de autómatas tal como se definen en la segunda fase del trabajo (definidos como TOA), ya que corresponden a una evolución de la primera, permitiendo representar la evolución de la red con respecto al tiempo y poder ser usados para realizar predicciones junto a reconfiguraciones de la red.

En la operación de este modulo se distinguen dos pasos. Un primer paso de aprendizaje inicial, que servirá para construir el autómata con suficiente información para realizar predicciones y reconfiguraciones. Un segundo paso donde todos los módulos están involucrados para cumplir el objetivo propuesto en la tesis. En la **fase inicial de aprendizaje**, se construye TOA que representa el comportamiento abstracto de la red, mientras se usan diferentes perfiles de tráfico y configuraciones de red diferentes que cambian aleatoriamente, recolectando toda esta información a partir de trazas de red (proporcionadas por el módulo de monitorización), que será usada por los demás módulos posteriormente.

Los autómatas pueden ser generados con diferentes niveles de detalle, es decir, con información de grano más fino o grueso. Esto es posible usando el parámetro *tiempo de ciclo*, que define cada cuanto se obtiene la información relacionada con los KPIs observados. Por lo tanto, definir un ciclo más corto (1s) permite obtener un mayor nivel de detalle del comportamiento de la red que cuando se define un ciclo más largo (10s), ya que algunas desviaciones de los KPIs pueden enmascararse al realizar la media de los datos. Consecuentemente, el número de estados del autómata depende del nivel de detalle elegido. En la Tabla A.2 se presentan varios ejemplos de autómatas aprendidos con niveles de detalle diferentes, es decir, usando diferentes valores del tiempo de ciclo. Estos autómatas corresponden a 24 horas de datos donde se producen 90 cambios de configuración, empezando desde el mismo instante de tiempo. Como es de esperar, el número de estados decrece de forma progresiva al usar un tiempo de ciclo mayor. Además, si el autómata se construye a partir de trazas previamente capturadas, el tiempo de creación crece considerablemente a medida que el tiempo de ciclo es menor. Debe tenerse en cuenta que la función *qty* también influirá directamente en el nivel de detalle del autómata, ya que se pueden definir un número diferente de rangos para abstraer los KPIs observados. En este caso se han usado los valores definidos en la Tabla A.1, definiendo un valor adicional (qty = -30) para considerar cualquier rango de valores no especificados.

UNIVERSIDAD DE MÁLAGA

Autómata ID	*Tiempo construcción del autómata*	*Tiempo de ciclo*	*Número de estados*	*Número de configuraciones*	*Tiempo de creación*
1_1	24 h	1 s	342	90	9212 s
1_5	24 h	5 s	259	90	1866 s
1_{10}	24 h	10 s	243	90	934 s
1_{30}	24 h	30 s	205	90	316 s
1_{60}	24 h	60 s	187	90	146 s
1_{120}	24 h	120 s	170	90	57 s

Table A.2: Ejemplo de autómatas aprendidos con diferentes valores del tiempo de ciclo.

Una vez que TOA de la fase inicial esté listo, se activa el **aprendizaje en línea**, es decir, se activan los módulos de predicción y reconfiguración, mientras que el algoritmo *Learn* continua expandiendo (por ejemplo, añadiendo nuevos estados o transiciones) de forma progresiva el TOA aprendido.

Predicción y Reconfiguración

Por un lado, el módulo de predicción se encarga y detectar de antemano posibles desviaciones de los KPIs observados con respecto a los requisitos exigidos por el tráfico. En el caso de que se detecte una desviación, el módulo intenta obtener una lista de configuraciones la red 5G adecuadas para seguir cumpliendo dichos requisitos. La elección de la configuración más adecuada está fuera del ámbito del algoritmo *Predict*. Existen diferentes métodos para elegirla, por ejemplo, usando la configuración que más tiempo ha cumplido con los requisitos previamente o que corresponde a mejores valores de ciertos KPIs. Existen parámetros adicionales que pueden ser definidos para cambiar el comportamiento de estos módulos. Por ejemplo, el tiempo mínimo que una configuración ha cumplido los requisitos para considerarla como válida; el número de veces que una configuración puede ser aplicada para reconfigurar la red durante la sesión de tráfico; el intervalo de tiempo que se considera necesario para que la red se estabilice después de una reconfiguración, donde no se considera volver a reconfigurar la red durante ese periodo.

Por otro lado, el módulo de reconfiguración se encarga de aplicar la configuración seleccionada, realizando los cambios oportunos en la red 5G de forma automática, es decir, aplicando las modificaciones en los parámetros que han sido elegidos. Este módulo es capaz de reconfigurar la dos opciones de redes 5G disponibles en el testbed, la RAN de Nokia junto a Open5GS y la plataforma de pruebas UXM 5G de Keysight.

A.5.2 Evaluación

La evaluación de AutomAdapt para conocer su rendimiento se presenta a continuación. Para ello, una serie de experimentos se han llevado a cabo sobre el testbed desarrollado en la Universidad de Málaga durante el desarrollo de esta tesis. Estos experimentos están

UNIVERSIDAD DE MÁLAGA

definidos por: el escenario de evaluación, principalmente compuesto por el hardware y el software del testbed, con la posibilidad de elegir entre ambas opciones de la red 5G; el patrón de tráfico enviado entre los dispositivos finales TSN; y la configuración de red 5G compuesta por cientos de parámetros, dando lugar a una cantidad inmanejable de posibles combinaciones. Por lo tanto, para la evaluación se han elegidos ciertos valores relevantes para el tráfico TSN sobre redes 5G, concretamente de la capa RLC funcionando en modo AM (ver Sección A.3.3 para más detalles). Dichos parámetros se modificarán en el escenario compuesto por la RAN de Nokia y Open5GS, con el fin de mostrar un entorno realista de una red comercial, donde se usarán diferentes patrones de tráfico mientras la configuración de la red 5G puede ser modificada. Este escenario puede ser replicado usando los componentes de la Figura A.5. Además, el código de los módulos y las instrucciones para reproducir la evaluación están disponibles en el repositorio de AutomAdapt [225].

Antes de comenzar con los experimentos, es necesario definir la función *qty* para indicar las correspondencias entre rangos de los KPIs observados (latencia y jitter) con valores individuales específicos. Además, también se establece la función *desirable* para fijar el objetivo de los valores de KPIs observados durante la sesión TSN. Como se mencionó anteriormente, es necesaria una fase inicial de aprendizaje para construir el autómata que se usará posteriormente para predecir y reconfigurar en la siguiente fase.

Para la **evaluación de la fase inicial de aprendizaje**, AutomAdapt monitoriza la red para construir autómata. Concretamente, se obtienen los KPIs observados al enviar el patrón de tráfico elegido, que en este caso corresponde al envío de un paquete con 1000 bytes de payload cada 10ms. Al mismo tiempo, los valores de la configuración de red 5G seleccionados, correspondientes a la capa RLC en modo AM ($MaxRetx_{Threshold}$, $Poll_{Byte}$, $Poll_{PDU}$, $t_{PollRetr}$, $t_{StatusProhibit}$, $t_{Reassembly}$) se modifican de forma aleatoria cada 10-15 minutos. Esto resulta en la creación de un autómata (TOA) por el algoritmo Learn que dependerá del número de configuraciones y la definición de la función *qty*. En la Tabla A.3 se muestran diferentes autómatas aprendidos durante diferentes intervalos. Se observa que mientras mayor es el tiempo de construcción del autómata, el número de estados crece, debido a las desviaciones aprendidas y un mayor número de configuraciones aplicadas. Esta información es la que se usará para predicción y reconfiguración, respectivamente.

Autómata ID	*Tiempo construcción del autómata*	*Número de estados*	*Número de configuraciones*
1	24 h	234	90
2	48 h	473	181
3	72 h	724	273
4	96 h	986	407
5	120 h	1222	497

Table A.3: Autómatas aprendidos durante la fase inicial de aprendizaje

UNIVERSIDAD DE MÁLAGA

Una vez generados los autómatas que sirven como base para predecir y reconfigurar, se procede a **evaluar el ciclo de operación completo de AutomAdapt**, en el cuál el autómata podrá seguir expandiéndose usando la función *Compose* del algoritmo *Learn*, y el algoritmo *Predict* podrá predecir posibles desviaciones de los KPIs observados aprendidas previamente, para así sugerir una configuración adecuada para seguir cumpliendo con los requisitos del tráfico. En este caso, se establece un objetivo (función *desirable*) de mantener la latencia en un rango de $[0, 7]$ms y el jitter dentro de $[0, 1]$ms, que corresponde a $qty = 1$. Para comprobar el correcto funcionamiento de AutomAdapt, es decir, si es capaz de configurar la red 5G para evitar posibles desviaciones y mantener la red en el estado deseado durante más tiempo, son necesarios algunos resultados de referencia. Para ello, los KPIs observados se observan durante al menos 12 horas usando una configuración fija de red 5G. Además, para obtener resultados sólidos se han probado cuatro configuraciones distintas, que se presentan en la Tabla A.4. Además, junto a cada configuración se muestra el tiempo en el estado deseado ($qty = 1$), es decir, los KPIs observados están dentro del rango.

Configuración de referencia	*ID 1*	*ID 2*	*ID 3*	*ID 4*
$MaxRetx_{Threshold}$	t6	t3	t32	t32
$Poll_{Byte}$	kB8	kB6500	kB8	mB40
$Poll_{PDU}$	p24576	p20480	p1024	p40960
$t_{PollRetr}$	ms500	ms140	ms70	ms40
$t_{StatusProhibit}$	ms800	ms90	ms100	ms40
$t_{Reassembly}$	ms170	ms45	ms10	ms180
Tiempo en el estado deseado (%)	41,99	40,81	45,02	47,22

Table A.4: Configuraciones de red 5G (RLC AM) utilizadas para obtener resultados de referencia

En este momento, AutomAdapt se ejecuta repetitivamente usando los autómatas mostrados en la Tabla A.3 para obtener los resultados de los KPIs observados mientras que se realizar predicciones y reconfiguraciones con el fin de poder realizar la comparación. La Figura A.10 muestra un análisis comparativo. Para ello, se muestra el porcentaje de mejora en términos del tiempo transcurrido en el estado deseado de la red, usando los diferentes autómatas y comparando los resultados con cada configuración de referencia. En general, los datos muestran una mejora gradual (0-45% aproximadamente) en la mayoría de los casos cuando se usa un autómata con mayor número de estados (se dispone de más información aprendida en la fase inicial). Sin embargo, se comprueba que para el caso del Autómata ID 5, que corresponde con el autómata con mayor número de estados, los resultados son peores que con Autómata ID 4. Por lo tanto, tener un mayor número de estados no siempre lleva a un mejor rendimiento, ya que influyen otros factores como los parámetros adicionales mencionados anteriormente o la forma de ordenar las configuraciones sugeridas por el algoritmo. En conclusión, es importante

UNIVERSIDAD DE MÁLAGA

elegir el autómata preciso para cada situación particular.

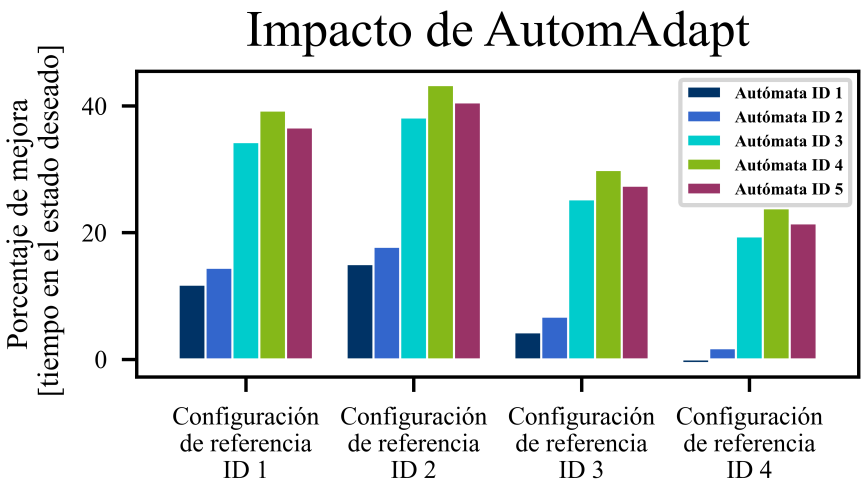


Figure A.10: Impacto de AutomAdapt: Tiempo transcurrido en el estado de red deseado

Por otro lado, debido a que AutomAdapt continua expandiendo el autómata que usa como base, la Tabla A.5 presenta los detalles del autómata resultante tras la evaluación del ciclo completo de operación. Se comprueba que se añaden un número similar de nuevos estados al autómata, mientras que el número de reconfiguraciones está alineado con los resultados de rendimiento, es decir, decrecen hasta el Autómata ID 4 y vuelve a crecer para el Autómata ID 5. En definitiva, el Autómata ID 4 es el que mejor rendimiento ofrece en este caso particular para el patrón de tráfico elegido, obteniendo un mayor tiempo transcurrido en el estado deseado y con un menor número de reconfiguraciones necesarias.

Autómata ID'	*Número de estados*	*Número de nuevos estados*	*Número de reconfiguraciones*
1	252	18	28
2	494	21	15
3	747	23	17
4	1004	18	11
5	1243	21	15

Table A.5: Autómatas resultantes durante la fase de predicción y reconfiguración

Por último, se muestra una **traza de ejemplo** extraída del proceso de evaluación, por lo tanto, usando $qty = 1$ y el mismo patrón de tráfico. Respecto a la configuración de red, la Tabla A.6 muestra los valores concretos de los parámetros (RLC AM) usados.

Primero, la Figura A.11 muestra la evolución de los KPIs observados durante la fase

UNIVERSIDAD DE MÁLAGA

Configuración	*ID 1*	*ID 2*
$MaxRetx_{Threshold}$	t3	t3
$Poll_{Byte}$	kB75	kB10
$Poll_{PDU}$	p8	p8192
$t_{PollRetr}$	ms50	ms220
$t_{StatusProhibit}$	ms140	ms10
$t_{Reassembly}$	ms5	ms120

Table A.6: Ejemplos de configuración de red 5G (RLC AM)

de aprendizaje inicial (latencia y jitter en los ejes y). En este caso, se aprende que la configuración ID 1 cumple los requisitos durante aproximadamente 230 segundos, y posteriormente se producen algunas desviaciones. Esta información será usada por AutomAdapt para realizar futuras predicciones y sugerir la reconfiguración de la red. Posteriormente, cuando AutomAdapt detecte que la red se encuentra en el mismo estado (usando la misma configuración de red y patrón de tráfico), se realizará dicha predicción y la consecuente sugerencia de reconfiguración (configuración ID 2 en este caso). Por lo tanto, antes de que se produzcan estas desviaciones, se aplicará de forma automática la configuración ID 2, para continuar cumpliendo con los requisitos del tráfico, como se muestra en la Figura A.12.

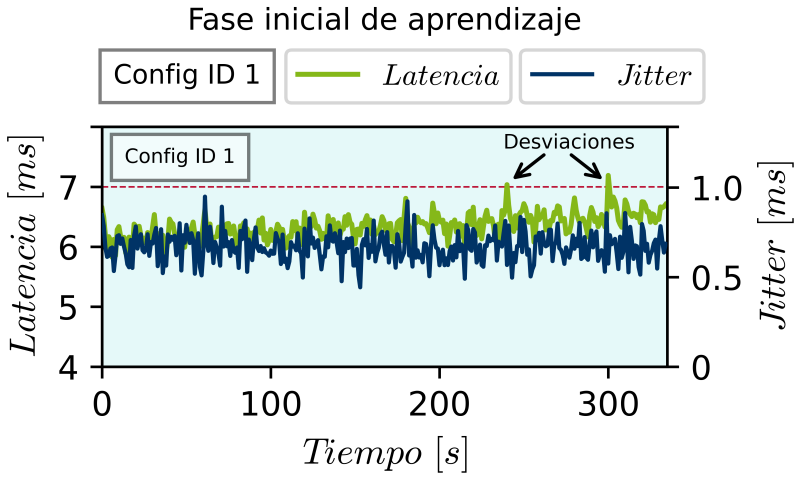


Figure A.11: Ejemplo de traza de red: KPI observados durante la fase inicial de aprendizaje

El conjunto completo de datos de AutomAdapt, que contiene los autómatas aprendidos durante la fase inicial de aprendizaje, las configuraciones de red aplicadas y las reconfiguraciones realizadas durante el ciclo de funcionamiento, está disponible en [225], incluyendo todos los detalles y archivos del proceso de evaluación. En total, el conjunto de datos consta de un total de **12071 estados de red aprendidos** distintos y **7211**

UNIVERSIDAD DE MÁLAGA

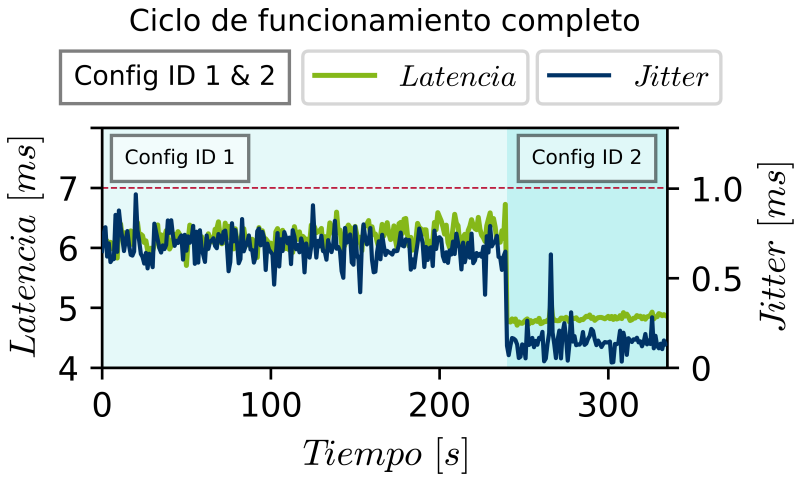


Figure A.12: Ejemplo de traza de red: KPI observados durante el ciclo completo de funcionamiento

configuraciones de red diferentes, lo que corresponde a **2310 horas de pruebas** realizadas. En este caso se ha usado un tiempo de ciclo de 10 segundos, por lo tanto, el número de estados puede variar en base al ciclo de tiempo seleccionado (más detalles en Tabla A.2), mientras que el número de configuraciones y horas de pruebas se mantienen intactas. Las Tablas A.7 and A.8 proporcionan información adicional. Primero, el número de transiciones de cada tipo (*config* = cambio de configuración; *qty* = cambio de valor de *qty*; *config+qty* = combinación de ambas; error = transiciones no capturadas). Segundo, el recuento del número de estados asociados a los valores definidos para la función *qty* durante la evaluación.

Tipo de transición	*Recuento*
config + qty	7284
qty	4675
config	105
error	7

Table A.7: Recuento del tipo de transiciones de los estados aprendidos del conjunto de datos

qty	*Recuento*
-30	100
-18	1
-8	447
-1	524
1	5964
2	2672
12	659
13	1704

Table A.8: Número de estados aprendidos para la definición de la función *qty* para la evaluación

UNIVERSIDAD DE MÁLAGA

A.5.3 Análisis de los Resultados

Para terminar, se interpretan los resultados obtenidos con el fin de extraer algunas conclusiones.

- El tiempo en el estado deseado crece de forma considerable a partir de un cierto número de estados aprendidos por el autómata. Esto se demuestra por los Autómatas ID 1 y 2 (menos de 500 estados), donde el porcentaje de mejora obtenido es más bajo (o incluso negativo). Por el contrario, los Autómatas ID 3, 4 y 5 obtienen mejoras significantes con respecto a los resultados de referencia de alrededor de 35-45% (ver Figura A.10).
- Debido a que AutomAdapt sigue aprendiendo durante su ciclo completo de operación, nuevos estados y transiciones (que previamente no hayan sido observadas) pueden ser añadidas. En base a la Tabla A.5 un número de estados nuevos son añadidos, posiblemente debido al tiempo que la red tarda en estabilizarse tras un cambio de configuración. Esto puede ajustarse mediante el parámetro adicional que define el tiempo de estabilización de la red, durante el cual no se aplica ninguna reconfiguración. Por lo tanto, configurar de forma precisa este parámetro es esencial para obtener un número menor de reconfiguraciones e incrementar el tiempo en el estado deseado.
- En general, es lógico asumir que mientras mayor número de estados hayan sido aprendido por AutomAdapt, mejor será su eficiencia de operación. Sin embargo, este no es siempre el caso (por ejemplo, Autómata ID 5 durante la evaluación). Además de que una mayor cantidad de información tiene que ser procesada, entran en juego otros factores (por ejemplo, como ordenar las configuraciones sugeridas por el algoritmo tras una predicción) que pueden influenciar en los resultados obtenidos de forma signiticativa.
- De forma similar, la información aprendida por AutomAdapt puede variar dependiendo del tiempo de ciclo elegido. Un tiempo de ciclo más alto lleva a un promedio de los resultados durante más tiempo, resultando en posibles pérdidas de información (por ejemplo, algunas desviaciones puede camuflarse). En el caso de que se esté aprendiendo a partir de trazas capturadas previamente, el tiempo de creación del autómata puede variar considerablemente. Además, el valor de la función *qty* también influye en estos resultados, ya que mientras mayor número de rangos se definan, posiblemente mayor número de estados serán aprendidos. Por lo tanto, es esencial ajustar estos parámetros en base a cada patrón de tráfico particular.
- Es importante obtener unos valores de referencia sólidos para poder comparar el rendimiento obtenido por AutomAdapt, ya que los resultados obtenidos con respecto a las diferentes configuraciones de referencia pueden variar alrededor de un 15-20%.
- Es necesario (al menos) alrededor de 7 días completos para evaluar de forma correcta

UNIVERSIDAD DE MÁLAGA

el rendimiento de AutomAdapt con un perfil de tráfico concreto. Esto se debe a que unos 4-5 días deben usarse para la fase de aprendizaje inicial con el fin de obtener un autómata con suficiente información para predecir y reconfigurar. Al menos dos días son necesarios para evaluar el ciclo completo de AutomAdapt usando los diferentes autómatas aprendidos y usando diferentes valores para los parámetros adicionales (tiempo de ciclo, tiempo de estabilización, etc.), con el objetivo de obtener la mejor combinación.

A.6 Conclusiones y Trabajo Futuro

Esta tesis se centra en la reconfiguración Zero-Touch de la red TSN sobre 5G para cumplir con los requisitos de tráfico de aplicaciones sensibles al tiempo. La integración de TSN y 5G, impulsada por la estandarización del 3GPP, se enfrenta a desafíos como la necesidad de traductores TSN para la interoperabilidad y la sincronización precisa del tiempo. El TSN Application Function es fundamental para gestionar la calidad de las sesiones de tráfico mediante la configuración de la red basada en los requisitos específicos del tráfico. Sin embargo, mantener esta calidad en las redes 5G es complejo debido a la variabilidad de los indicadores de rendimiento como la latencia y el jitter. Para ello, se ha desarrollado un testbed TSN sobre 5G como parte de esta tesis, en el cual a través del uso de autómatas de aprendizaje se construyen modelos de comportamiento de la red 5G de forma automática (algoritmo *Learn*), incluyendo los resultados de los indicadores de calidad y la configuración de la red 5G. Estos modelos son usados para predecir posibles desviaciones en los indicadores de rendimiento y aplicar automáticamente una reconfiguración de la red 5G (algoritmo *Predict*). Además, se introduce la herramienta AutomAdapt [225], la cual incluye los algoritmos mencionados anteriormente y se encarga de realizar de forma automática la reconfiguración de la red 5G para cumplir los requisitos del tráfico. Los resultados obtenidos de la evaluación de esta herramienta sobre el testbed demuestran la viabilidad de la red 5G para soportar una parte sustancial de los diferentes patrones de tráfico industrial, incluso llegando a mejoras en el tiempo que se cumplen los requisitos del tráfico de hasta el 45%. Esta evaluación permite obtener datos representativos y fiables en un entorno real, mejorando la compresión del sistema, que junto a los autómatas de aprendizaje habilitan la explicabilidad del sistema, permitiendo comprender mejor las decisiones tomadas. Sin embargo, alcanzar una reconfiguración completamente autónoma en este tipo de redes complejas con son TSN sobre 5G en cualquier circunstancia no es posible, especialmente en aquellas situaciones con unos requisitos de tráfico más exigentes. Consecuentemente, es fundamental el uso de toda la información disponible en la red, así como métodos y técnicas complementarias que ayuden en las situaciones más críticas, por ejemplo, cuando el autómata no dispone de suficiente información.

Para trabajo futuro, existen varios temas de investigación abiertos y retos que requieren atención. Primero, la extensión del trabajo realizado mediante la incorporación de capas adicionales de la pila de protocolos 5G (por ejemplo, MAC y PDCP), o parámetros

UNIVERSIDAD DE MÁLAGA

relevantes de la propia red TSN, pueden ayudar a mejorar los resultados obtenidos. Segundo, la mejora del testbed desarrollado durante esta tesis para incorporar nuevas funcionalidades basadas en los procedimientos definidos en Release 16 [28]. Por ejemplo, desarrollando técnicas avanzadas de traducción del tráfico [98], [99] o el soporte nativo de sincronización temporal sobre la red 5G [108]. Por otro lado, la incorporación de técnicas de ML [199], [200] para asistir en las situaciones en las que los autómatas de aprendizaje no son capaces de proporcionar una solución, apoyado por XAI para capturar posibles causas de error [206]. Por ejemplo, cuando no es posible sugerir una configuración adecuada tras una predicción o no se reconoce el estado leído en el autómata (mensajes *Warning* y *Unknown*, respectivamente). Además, la inclusión de un gemelo digital de la red que se ejecute de forma paralela [205] puede ayudar a solucionar este tipo de situaciones, por ejemplo, ayudando en la realización de predicciones de los KPIs observados o a decidir que configuración de red es más adecuada para continuar cumpliendo con los requisitos. Finalmente, la evaluación de AutomAdapt puede extenderse incluyendo el impacto de un conjuntos de valores dado para los parámetros adicionales (por ejemplo, tiempo de estabilización), o como afecta la elección de la configuración de la lista de las sugeridas por el algoritmo. También, la inclusión de herramientas de verificación de ciertas propiedades en tiempo de ejecución puede contribuir positivamente en los resultados, permitiendo incluso añadir nuevos patrones de tráfico más exigentes en términos de requisitos.

UNIVERSIDAD DE MÁLAGA