Toward optimized 802.15.4 industrial wireless networks : Harnessing machine learning and digital twins

Mehdi Kherbache

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Toward Optimized 802.15.4 Industrial Wireless Networks : Harnessing Machine Learning and Digital Twins

THÈSE

présentée et soutenue publiquement le 06/12/2023

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention Automatique, Traitement du signal et des images, Génie informatique)

par

Mehdi Kherbache

Composition du jury

Président :	William Derigent	Professeur, Université de Lorraine, CRAN
Rapporteurs :	Fabrice Theoleyre	Directeur de Recherche - CNRS, Université de Strasbourg, Icube
	Olivier Cardin	Maitre de conférences - HDR, Université de Nantes, LS2N
Examinateurs :	Ah-Lian Kor	Professeur, Leeds Beckett University
Invité :	Clément Poulain	Responsable de la Transformation Industrielle, SNCF
Encadrants :	Moufida Maimour	Maitre de conférences - HDR, Université de Lorraine, CRAN
	Eric Rondeau	Professeur, Université de Lorraine – Directeur de thèse, CRAN

CRAN — UMR 7039

Acknowledgments

First and foremost, I extend my heartfelt gratitude to my supervisors, Prof. Eric Rondeau and Dr. Moufida Maimour. Your guidance has been the cornerstone of my PhD journey. Your unwavering support, innovative ideas, and invaluable advice were instrumental in bringing this endeavor to fruition. I consider myself fortunate to have had the opportunity to work under your supervision and cherish the experience immensely.

I also wish to express my sincere appreciation to the jury members. Your dedication and time in scrutinizing my thesis, coupled with your insightful feedback and discussions, not only enhanced the quality of my work but also opened new avenues for future exploration.

A special acknowledgment goes to my friends, who have made this journey an unforgettable experience. To Alaa (Sat), thank you for the countless memorable moments, les pauses, and for being a beacon of positivity, even during the most challenging times. My gratitude extends to my close friends – Islam, Nazim, Anis, Adaoui (BIG5), Sami, and everyone at NDL and Placieux. Your camaraderie has been a source of great joy and comfort. A special mention for my girlfriend, whose unwavering support and presence have been a constant source of strength throughout this journey.

Lastly, I owe an immense debt of gratitude to my parents. To my Father, may you rest in peace, your belief in me and encouragement to pursue my dreams have been my guiding light. To my dear mother, your wisdom has a way of simplifying complexities, and I am forever grateful for your prayers that have eased my path during trying times. While words can never fully express my gratitude, I dedicate this thesis to you. I also extend my thanks to my brothers, Hamoud, Zaka, and Chafik, for their enduring support.

I must also apologize if I have inadvertently omitted to mention anyone who has been a part of this journey. Your contributions and support have not gone unnoticed, and I am thankful for your involvement.

This achievement is not just a reflection of my efforts, but a testament to the collective support and encouragement of each one of you. Thank you.

To my dear father resting in heaven. To my mother may Allah grant her many more years of life.

Abstract

The Industrial Internet of Things (IIoT) presents a complex landscape with numerous constraints, particularly due to their use to control critical applications in Industry 4.0. The requirements in such a context in terms of energy efficiency and quality of service (delay, reliability, determinism and robustness) are strict and of paramount importance. Consequently, there is a pressing need for sophisticated management mechanisms throughout their entire lifecycle to meet these needs. This thesis explores two technological fronts to address this challenge : Reinforcement Learning-based Time Slotted Channel Hopping (TSCH) scheduling and Network Digital Twin (NDT).

TSCH scheduling in IIoT, is identified as a crucial area to optimize the performance of these networks. Several works proposed Reinforcement Learning-based scheduling techniques for TDMA (Time Division Multiple Access) MAC protocols, and particularly for TSCH. However, using this approach in a constrained network like the IIoT carries the risk of elevated energy consumption. This is due to the continuous learning process and coordination among the nodes necessary to manage the non-stationarity issue in the network, which is viewed as a Multi-Agent System. This thesis introduces a novel Reinforcement Learning-based distributed scheduling algorithm, QL-TSCH-plus. This algorithm has been designed to be adaptive and efficient, with reduced energy consumption and delay targets inherent to IIoT environments.

Parallel to the development of TSCH scheduling, this thesis adopts the concept of NDT as a viable solution for effective IIoT management. Digital twins have been increasingly used to optimize the performances of industrial systems. Capitalizing on this technology, a holistic NDT architecture for the IIoT is proposed, where the network is integrated with other industrial components. The architecture leverages Software Defined Networking to enable closed-loop network management across the entire network life-cycle (from design to service). This architecture enables quick validation of networking solutions in an industrial environment because of the continuous link between the NDT and the physical IIoT network.

Moreover, we propose to model the IIoT in the NDT using Petri-nets, enabling data-driven Petri-nets. These serve as coarse-grained formal models enabling fast simulation time for what-if scenarios execution, and real-time fault detection that is crucial in mission-critical industrial applications.

Keywords: TSCH, Industry 4.0, Scheduling, IEEE 802.15.4, Machine Learning, Digital Twin

Résumé

L'Internet Industriel des Objets (IIoT) offre un paysage complexe avec de nombreuses contraintes, notamment en raison de son utilisation pour piloter des applications critiques dans l'Industrie 4.0. Les exigences dans un tel contexte, en termes d'efficacité énergétique et de qualité de service (délai, fiabilité, déterminisme et robustesse), sont strictes et d'une importance capitale. Il en découle un besoin impératif de mécanismes de gestion sophistiqués tout au long de leur cycle de vie pour répondre à ces exigences. Cette thèse explore deux axes technologiques pour relever ce défi : l'ordonnancement basé sur l'Apprentissage par Renforcement pour le protocole TSCH (Time Slotted Channel Hopping) et le Jumeau Numérique du Réseau (JNR).

La planification TSCH dans l'IIoT est identifiée comme un domaine essentiel pour optimiser la performance de ces réseaux. Plusieurs travaux ont proposé des techniques de planification basées sur l'Apprentissage par Renforcement pour les protocoles MAC TDMA (Time Division Multiple Access) , et plus particulièrement pour TSCH. Toutefois, l'utilisation de cette approche dans un réseau contraint comme l'IIoT présente le risque d'une consommation énergétique accrue. Cela est dû au processus d'apprentissage continu et à la coordination entre les nœuds nécessaire pour gérer le problème de non-stationnarité du réseau, considéré comme un Système Multi-Agents. Cette thèse présente un nouvel algorithme de planification distribuée basé sur l'Apprentissage par Renforcement, nommé QL-TSCH-plus. Cet algorithme est conçu pour être adaptatif et efficace, avec des objectifs de réduction de la consommation d'énergie et des délais propres aux environnements IIoT.

En parallèle du développement de l'ordonnancement pour TSCH, cette thèse adopte le concept de JNR comme solution viable pour une gestion efficace de l'IIoT. Les jumeaux numériques sont de plus en plus utilisés pour optimiser les performances des systèmes industriels. En capitalisant sur cette technologie, une architecture JNR holistique pour l'IIoT est proposée, où le réseau est intégré avec d'autres composants industriels. L'architecture exploite les réseaux définies par logiciel (SDN) pour permettre une gestion en boucle fermée du réseau tout au long de son cycle de vie (de la conception au service). Cette structure facilite la validation rapide des solutions de mise en réseau dans un environnement industriel grâce au lien continu entre le JNR et le réseau IIoT physique.

De plus, nous proposons de modéliser l'IIoT dans le JNR en utilisant des réseaux de Petri, permettant des réseaux de Petri basés sur les données. Ces modèles servent de modèles formels à gros grains, permettant une simulation rapide pour l'exécution de scénarios hypothétiques et une détection des défauts en temps réel, essentielle dans les applications industrielles critiques.

Mots-clés: TSCH, Industrie 4.0, Ordonnancement, IEEE 802.15.4, Apprentissage Automatique, Jumeau Numérique

List of Abbreviations

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks
6P 6top Protocol

AI Artificial Intelligence
ALICE Autonomous Link-based Cell Scheduling
AMQP Advanced Message Queuing Protocol
AMUS Adaptive MUlti-hop Scheduling
ANM Application-driven Network Manager
API Application Programming Interface
AP Action Peeking
APT Action Peeking Table

CAGR Compound Annual Growth Rate
CCA Clear Channel Assessment
CDT-M CDT-Manager
CDT Composed Digital Twin
CoAP Constrained Application Protocol
CPS Cyber Physical System
CSMA-CA Carrier Sense Multiple Access-Collision Avoidance

DCN Data-Center Network
DeTAS Decentralized Traffic Aware Scheduling
DiSCA Distributed Scheduling for Convergecast in multichannel WSNs Algorithm
DM Deadline Monotonic
DNN Deep Neural Network
DODAG Destination-Oriented Directed Acyclic Graph
DSI Data Send Interval
DTLS Datagram Transport Layer Security
DT Digital Twins
DTEO Digital Twin Entity Orchestrator

EB Enhanced Beacon

EDF Earliest Deadline First
EPD Earliest Proportional Deadline
ETX Expected Transmission Count

FI Freshness Indicator
FPS Flexible Production System

GNN Graph Neural Network

HTTP HyperText Transfer Protocol

IDT Intelligent Digital Twin
IIoT Industrial Internet of Things
IETF Internet Engineering Task Force
IoT Internet of Things
ISM Industrial, Scientific and Medical
ITU International Telecommunication Union

LAN Local Area Networks
LLC Logical Link Control
LLF Least Laxity First
LLN Low-power and Lossy Network
LPWA Low-Power Wide Area
LR-WPAN Low Rate Wireless Personal Area Network

MAC Medium Access Control
MEC Mobile Edge Computing
MODESA Multichannel Optimized DElay time Slot Assignment
MQTT Message Queuing Telemetry Transport
MSA Minimal Scheduling Algorithm
MSF 6TiSCH Minimal Scheduling Function
MTU Maximum Transmission Unit

NCS Networked Control Systems
NDT Network Digital Twins
NFV Network Function Virtualization

OF Objective Function
OTF On-The-Fly
OST On-Demand TSCH Scheduling with Traffic Awareness algorithm
O-QPSK Offset Quadrature Phase Shift Keying

PCE Path Computation Element

PD Proportional Deadline
PDR Packet Delivery Ratio
PPO Proximal Policy Optimization

QoE Quality of Experience
QoS Quality of Service

RL Reinforcement Learning
RPL Routing Protocol for Low-Power and Lossy Networks

S-CA SDN Control Agent
S-Ctrl SDN Controller
SDN-WISE Software Defined Networking solution for WIreless SEnsor Networks
SDT-M SDT-Manager
SDT Simple Digital Twin
SDVN Software-Defined Vehicular Network
SF Scheduling Function
SoC System on Chip

TASA Traffic Aware Scheduling Algorithm
TCP Transmission Control Protocol
TDMA Time Division Multiple Access
TPN Timed Petri-Nets
TSCH Time Slotted Channel Hopping

UDP User Datagram Protocol
UDGM Unit Disk Graph Medium

WSN Wireless Sensor Networks
WBAN Wireless Body Area Network

XML Extensible Markup Language

Résumé de la thèse

Tout au long de l'histoire, les révolutions industrielles ont marqué des changements significatifs dans notre manière de produire et de consommer. La première révolution industrielle, à la fin du 18ème siècle, a introduit la production mécanisée grâce à la puissance de l'eau et de la vapeur. La seconde, au début du 20ème siècle, a apporté la production de masse avec l'aide de l'électricité, conduisant aux chaînes de montage et à la croissance des usines. La troisième, à la fin du 20ème siècle, a inauguré l'ère de l'automatisation, avec les ordinateurs et l'électronique jouant un rôle essentiel dans les processus de production.

Aujourd'hui, alors que nous naviguons dans le 21ème siècle, nous assistons à la quatrième révolution industrielle, connue sous le nom d'Industrie 4.0. Cette dernière phase se caractérise par la fusion des mondes physique et numérique, avec des dispositifs et des systèmes interconnectés alimentés par des données et l'intelligence artificielle, redéfinissant les industries et les économies à l'échelle mondiale. Au cœur de cette transformation se trouvent les Réseaux de Capteurs Sans Fil (RCSF), qui ont évolué depuis leurs premières applications, telles que le suivi des sous-marins pendant la Guerre Froide, pour sous-tendre les villes intelligentes émergentes d'aujourd'hui. Ces avancées dans les RCSF ont ouvert la voie à l'Internet des Objets (IoT), qui est maintenant profondément intégré dans des industries telles que la fabrication et la logistique. Ce nouvel Internet Industriel des Objets (IIoT) a beaucoup de promesses, mais avec un grand potentiel viennent de grands défis. L'un des plus grands défis est de s'assurer que ces dispositifs connectés puissent communiquer rapidement, de manière fiable et efficace.

L'IIoT nécessite un réseau fixe et bien structuré pour des liaisons machine-machine à enjeux élevés qui répondent à des exigences strictes en termes de ponctualité et de fiabilité. Dans des applications telles que l'automatisation des processus, l'IIoT nécessite des retards limités au niveau de la milliseconde et une fiabilité de transmission de 99,99% (Åkerberg, Gidlund et Björkman, 2011). Cela représente un défi pour l'IIoT puisque le médium sans fil est peu fiable, exposé aux interférences dans la bande ISM qu'il partage avec d'autres technologies sans fil (par exemple, le Wi-Fi), et la qualité des liaisons radio est sujette à dégradation au fil du temps. L'amendement IEEE 802.15.4e (std., 2012) a proposé différents modes MAC pour répondre aux exigences des applications industrielles, parmi lesquels le TSCH (Time Slotted Channel Hopping). Les principaux domaines d'application du TSCH comprennent l'automatisation industrielle et le contrôle des processus, en particulier les applications de production d'énergie, les systèmes de gestion de l'eau, les usines de production alimentaire, les sites de production pharmaceutique et les usines de traitement du pétrole et du gaz. Ce protocole représente un grand potentiel pour répondre aux exigences industrielles en matière de ponctualité, de fiabilité et d'efficacité énergétique.

Le protocole TSCH assure le créneau temporel en établissant un cadre de créneaux périodique, composé d'un certain nombre de créneaux horaires, auquel les nœuds adhèrent tout au long du fonctionnement du réseau. De plus, il permet le saut de canal, où les nœuds oscillent/sautent parmi divers canaux de fréquence en suivant une séquence prédéterminée lors de la transmission

des données. Cependant, la norme ne définit pas comment les nœuds sont censés communiquer, c'est-à-dire sur quel créneau horaire et quel canal un nœud est censé transmettre/recevoir des données. Cet aspect est appelé l'ordonnancement TSCH, il a suscité un intérêt considérable parmi les chercheurs, et de nombreuses stratégies d'ordonnancement ont été proposées, soit dans un mode centralisé, soit décentralisé (Urke, Kure et Ovsthus, 2022). Une tendance plus récente dans l'ordonnancement TSCH est l'adoption de l'Apprentissage par Renforcement (RL) (Sutton et Barto, 2018). Ceci permet d'avoir des ordonnanceur adaptatif grâce à l'interaction continue avec l'environnement sans fil (Savaglio et al., 2019).

Cependant, l'application du RL pour l'ordonnancement TSCH apporte plusieurs défis, particulièrement liés à la nature sans fil du réseau et les capacités de calcul/stockage limités des capteurs à énergie limitée. Nous introduisons QL-TSCH-plus (Kherbache, Maimour et Eric Rondeau, 2023), un nouvel algorithme d'ordonnancement distribué basé sur le Q-learning pour les réseaux TSCH. Il s'agit d'une amélioration d'un ordonnanceur autonome existant basé sur le Q-learning, appelé QL-TSCH (H. Park, Haeyong Kim, S.-T. Kim et al., 2020). QL-TSCH-plus améliore le mécanisme de "peeking" d'action utilisé dans QL-TSCH pour aborder le problème de non-stationnarité dans le réseau à un schéma distribué. Plus spécifiquement, au lieu d'écouter en continu la communication des nœuds voisins, QL-TSCH-plus permet aux nœuds de diffuser les créneaux de transmission appris pour mettre à jour les tables de "Action Peeking". De plus, les créneaux de transmission diffusés sont utilisés pour allouer efficacement les créneaux de réception. Ainsi, QL-TSCH-plus réduit la consommation d'énergie jusqu'à 47% par rapport à QL-TSCH tout en maintenant la fiabilité et la ponctualité, le rendant approprié pour l'IIoT. Ces performances sont obtenues via une évaluation de notre algorithme dans un environnement de trafic hautement hétérogène avec des algorithmes d'ordonnancement autonomes et distribués bien connues dans la littérature. Cependant, l'algorithme est seulement meilleur dans des scénarios industriels avec des besoins strictes de ponctualité. Dans d'autres scénarios comme les applications multimédia de surveillance, l'algorithme autonome d'ordonnancement Orchestra (Duquennoy et al., 2015) est mieux adapté d'après nos résultats. Ceci montre le besoin d'aller plus loin et d'explorer des méthodes RL plus poussés, comme celles basées sur les réseaux de neurones, ou l'Apprentissage Profond par Renforcement (Deep RL). Toutefois, les architectures de réseau sans fil actuelles ne permettent pas l'adoption de telles méthodes. L'émergence des Jumeaux Numériques de Réseau (Network Digital Twins, NDT) apporte de nouvelles opportunités et avantages.

Les NDT sont des candidats prometteurs pour l'adoption des algorithmes d'ordonnancement TSCH basé sur le Deep RL car ils peuvent permettre une réplique en temps réel du réseau et activer des services intelligents de gestion de réseau, y compris l'allocation de ressources. En effet, un NDT peut fournir un environnement sûr, flexible et rentable pour développer, former et valider des solutions d'ordonnancement TSCH basées sur l'Apprentissage Profond par Renforcement. Il peut relever de nombreux défis associés à cette approche en offrant une plateforme virtuelle qui reflète étroitement le réseau réel. Cet environnement virtuel facilite le prototypage rapide, la génération abondante de données pour la formation, et la capacité à tester divers scénarios de réseau. De plus, il peut permettre une approche équilibrée entre l'apprentissage en ligne et hors ligne, assurant une performance robuste du modèle. Cependant, le concept de NDT en est encore à ses débuts, en particulier pour l'IIoT où de nombreux défis restent inexplorés dans la littérature.

Nous proposons une architecture de jumeau numérique holistique pour l'IIoT, où le composant réseau est considéré par l'adoption du concept de NDT (Network Digital Twin). L'objectif est de permettre une validation rapide des solutions de mise en réseau dans un environnement industriel, puisque le NDT est continuellement lié au monde physique, depuis les premières étapes

de conception jusqu'à la phase de production. De plus, le NDT permet une évolution continue du réseau industriel en exploitant les données collectées, ainsi que les algorithmes d'IA, pour améliorer la performance du réseau, prévenir les défaillances et augmenter la durée de vie utile restante du réseau. Le NDT proposé peut être inclus dans n'importe quelle architecture basée sur les Jumeaux Numériques où un réseau de communication est requis dans le monde physique. Nous validons l'architecture holistique proposée à travers son application aux premières étapes de conception d'un projet industriel au sein de notre université avec des exigences en temps réel. Précisément, nous utilisons le concept de NDT pour évaluer différents scénarios afin de choisir le mécanisme de communication le plus adapté répondant aux exigences de temps réel de l'application FPS. Nos résultats révèlent que le suréchantillonnage n'est plus nécessaire lors de l'utilisation d'un protocole MAC de type TDMA tel que TSCH avec TASA (Traffic Aware Scheduling Algorithm) (Palattella, Accettura et al., 2012) comme ordonnanceur. Une architecture détaillée du NDT est également fournie pour décrire les différents composants internes du NDT et leurs interactions. De plus, elle est conçue pour permettre divers services intelligents, y compris l'allocation de ressources. En particulier, cela permet d'intégrer des algorithmes d'ordonnancement TSCH basés sur l'Apprentissage Profond par Renforcement dans l'IIoT. Dans les architectures NDT proposées, un simulateur est utilisé pour permettre l'évaluation de la performance en mode 'et si' dans le but d'améliorer l'opération du réseau. Cependant, les simulateurs sont conçus avec délicatesse et étroitement couplés, ce qui conduit à un temps d'exécution élevé et une complexité computationnelle importante.

Ceci nous a conduit à proposer de modéliser le réseau IIoT dans le NDT avec les Réseaux de Pétri (RdP) pour plusieurs raisons : i) pour adresser le problème du coût d'exécution élevé des simulateurs, ii) pour apporter l'explicabilité dans le modèle contrairement aux travaux qui se basent sur les Réseaux de Neurones Graphiques (Graph Neural Networks, GNN) (Almasan et al., 2022), ces derniers ne sont pas explicables et donc pas encore acceptés dans les systèmes industriels critiques, iii) pour apporter l'aspect formel au modèle et donc la possibilité de faire de la vérification formelle, iv) pour fournir une modélisation sensible au temps avec les réseaux de pétri temporels. Nous introduisons une architecture de Jumeau Numérique axée sur les données pour l'IIoT, qui intègre les Réseaux de Petri. Cette architecture proposée est évaluée à travers une étude de cas d'un réseau à petite échelle modélisé en utilisant un modèle de RdP temporel (Timed Petri Net) producteur-consommateur à grains grossiers et léger, impliquant à la fois un banc d'essai simulé et réel pour l'IIoT. Les résultats démontrent que le modèle est capable de prédire avec précision la fiabilité du réseau IIoT via le PDR (Packet Delivery Ratio), en faisant un outil précieux pour exécuter des scénarios hypothétiques et maintenir une haute fiabilité du réseau. Plus explicitement, le NDT peut être configuré avec les exigences de l'application en termes de PDR, c'est-à-dire, quel est le seuil minimal pour qualifier un PDR de « bon » ou de « mauvais » pour l'application. En prédisant le PDR en fonction des paramètres opérationnels du réseau, le NDT peut envoyer des commandes pour réduire le DSI afin d'améliorer la fiabilité lorsque le PDR prédit est en dessous du seuil minimal.

De plus, le modèle sert de module de détection de fautes. En considérant le modèle de RdP comme le modèle de base représentant la réalité et en comparant continuellement le PDR du modèle de RdP avec le PDR du réseau réel. Lorsqu'une déviation excédant un certain seuil à définir selon l'application ait lieu, le NDT alerte d'une faute. Ceci permet une prise d'action rapide pour régler la faute potentielle. L'évaluation de l'observabilité du système met en évidence le rôle crucial des mises à jour fiables du jumeau dans l'influence sur la performance du NDT, comme démontré par la précision de la prédiction du PDR dans notre modèle. En assurant une haute fiabilité dans les mises à jour du jumeau, l'observabilité du système physique à l'aide du NDT est améliorée, car un plus grand volume de données de télémétrie est collecté. Cette

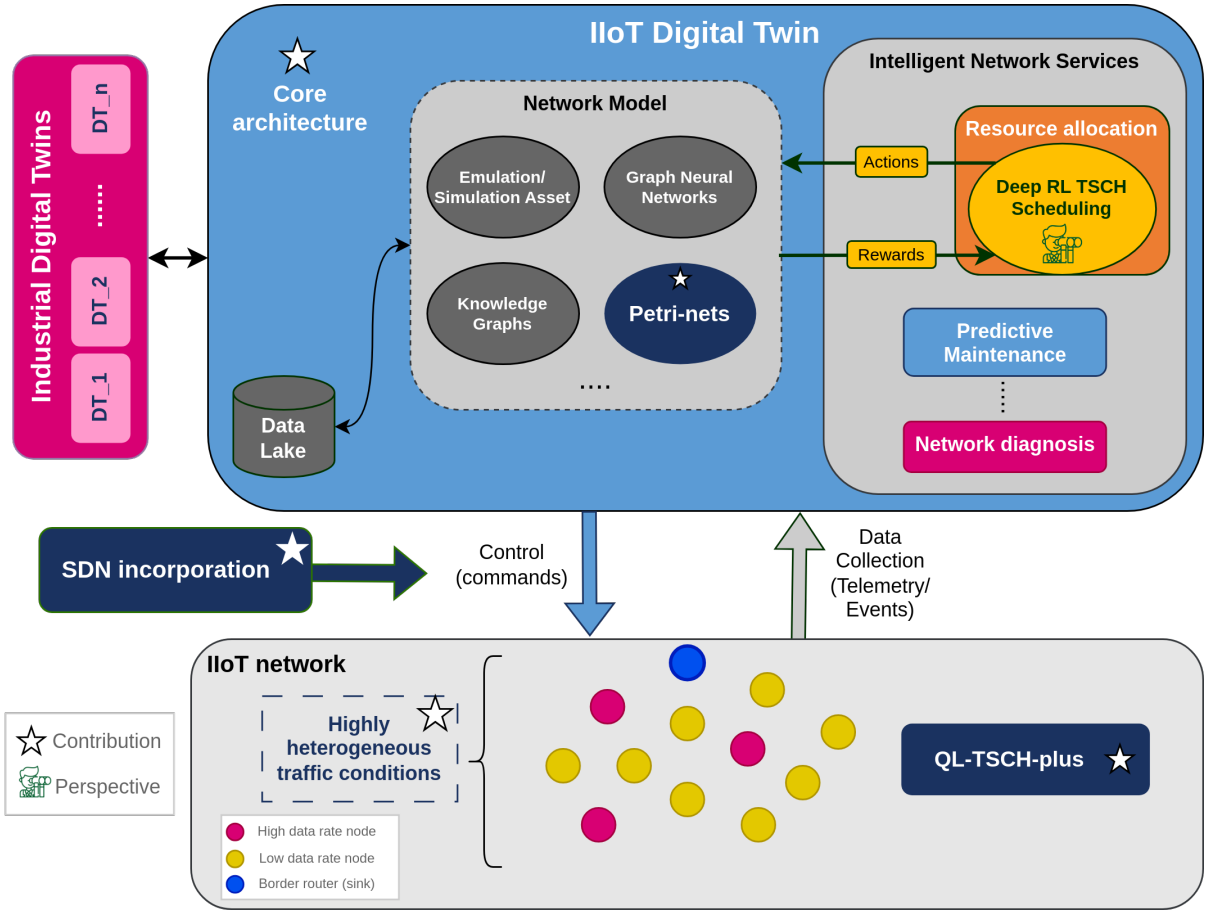


FIGURE 1 – Résumé des contributions

observabilité accrue permet une prédiction plus précise pour notre modèle.

Globalement, les contributions de ma thèse peuvent être résumées dans la Figure **??**. Le chapitre 3 contribue principalement au réseau physique IIoT en proposant QL-TSCH-plus et en évaluant les ordonnanceurs TSCH décentralisés sous des conditions de trafic hétérogènes. Le chapitre 4 propose une architecture de jumeau numérique holistique pour l'IIoT ainsi que l'architecture détaillé du NDT en exploitant le paradigme SDN. Le chapitre 5 intègre des réseaux de Petri pour modéliser l'IIoT au sein du NDT. L'implémentation de l'ordonnancement par apprentissage profond renforcé dans le NDT fait partie de nos perspectives que nous discutons dans la section suivante.

Les contributions de cette thèse peuvent être étendues dans plusieurs directions, nous en présentons certaines ci-après.

Expériences sur un banc d'essai réel Cette thèse évalue des algorithmes d'ordonnancement TSCH décentralisés sous conditions de trafic hautement hétérogènes dans un environnement de simulation. Les futures expériences prévues incluent des tests dans sur la plateforme FIT IoT-LAB pour observer les conditions réelles de réseaux sans fil et dans des environnements industriels, confrontés à des défis tels que des surfaces métalliques réfléchissantes, de la poussière et des températures élevées. L'objectif est d'évaluer la performance, en particulier de l'algorithme QL-TSCH-plus, dans ces conditions industrielles réelles et de confirmer les conclusions tirées de

la simulation.

Ordonnancement TSCH avec Deep RL en utilisant le NDT Comme je l'ai déjà mentionné dans les motivations de ma thèse dans la Section 1.2, le Deep RL peut apporter plusieurs avantages à l'ordonnancement TSCH. Principalement, cela améliorerait l'évolutivité et la réactivité des ordonnanceurs centralisés en fournissant un ordonnanceur globalement adaptatif. Dans cette thèse, nous avons préparé le terrain pour intégrer de telles techniques dans une vraie pile IIoT en proposant le NDT pour l'IIoT. Les prochaines étapes consistent à implémenter effectivement un ordonnanceur TSCH basé sur le Deep RL dans le NDT qui apprend continuellement l'ordonnancement optimal basé sur l'état actuel du réseau. Les actions de l'ordonnanceur seraient évaluées dans le NDT via l'interaction avec le modèle de réseau (Simulateur, GNN, réseau de Petri) et reçoit des récompenses de celui-ci pour apprendre la meilleure politique d'ordonnancement. Ainsi, l'ordonnanceur améliorerait progressivement sa capacité de généralisation à un large éventail de scénarios en exploitant non seulement les données historiques du réseau collectées par le NDT mais aussi l'état actuel du réseau. Néanmoins, le déploiement du modèle de l'ordonnancement Deep RL dans le NDT comporte le risque de temps d'arrêt du réseau en cas de défaillance du NDT. Ainsi, les ordonnanceurs décentralisés peuvent servir d'algorithmes de « secours » qui prennent en charge la responsabilité de l'ordonnancement en cas de défaillance du NDT.

Collaboration du NDT avec les DT Industriels Dans les environnements industriels modernes, divers Jumeaux Numériques (DTs) coexistent, chacun ayant un rôle spécifique, comme la gestion d'équipements de fabrication ou de chaînes d'approvisionnement. Ces DTs dépendent d'une communication réseau fiable pour leurs mises à jour et décisions en temps réel. Le Jumeau Numérique du Réseau (NDT) joue un rôle clé en agissant comme un miroir virtuel de l'infrastructure réseau industrielle. En collaborant continuellement avec les DTs industriels, le NDT peut comprendre et répondre à leurs besoins spécifiques en matière de réseau, comme la bande passante et les exigences en temps réel. Ainsi, le NDT aide à simuler et optimiser les politiques de réseau pour satisfaire les besoins de chaque DT industriel. Cette collaboration vise à adapter proactivement le NDT aux évolutions des exigences des DTs, assurant un environnement réseau résilient et optimisé. À l'avenir, une collaboration étroite entre les différents DTs, y compris le NDT, est envisagée pour mieux orchestrer l'ensemble du système industriel.

Scalabilité des réseaux de Petri. Bien que les réseaux de Petri offrent une rigueur formelle et une explicabilité, assurer leur scalabilité pour des scénarios de réseaux à grande échelle dans l'environnement NDT est un sujet qui reste à être pleinement abordé. Une direction prometteuse réside dans l'adoption de types de réseaux de Petri plus avancés, notamment les réseaux de Petri Hiérarchiques et les réseaux de Petri Colorés. Les réseaux de Petri Hiérarchiques offrent la capacité de modéliser des systèmes à différents niveaux de détail, introduisant une modularité particulièrement bénéfique pour les vastes réseaux IIoT interconnectés. D'autre part, les réseaux de Petri Colorés, avec leur capacité à représenter plusieurs processus similaires avec différents paramètres en utilisant un seul modèle, ajoutent de la compacité et de la clarté au processus de modélisation. En intégrant ces types avancés de réseaux de Petri, nous anticipons la simplification de la représentation des réseaux IIoT complexes. Cette progression renforce non seulement la capacité de généralisation du modèle mais l'équipe également pour s'adapter de manière transparente à une myriade de scénarios IIoT à grande échelle. Cependant, un défi subséquent émerge lors de la mise en miroir de réseaux IIoT à grande échelle en utilisant le NDT, qui est de concevoir

efficacement les données devant être transférées au NDT nécessaires pour peupler ces types de réseaux de Petri avec les paramètres opérationnels de l'IIoT.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Throughout history, industrial revolutions have marked significant shifts in the way we produce and consume. The first industrial revolution in the late 18th century, introduced mechanized production through water and steam power. The second, in the early 20th century, brought mass production with the help of electricity, leading to assembly lines and the growth of factories. The third, in the late 20th century, ushered in the era of automation, with computers and electronics playing a pivotal role in production processes.

Now, as we navigate the 21st century, we are witnessing the fourth industrial revolution, known as Industry 4.0. This latest phase is characterized by the fusion of the physical and digital worlds, with interconnected devices and systems powered by data and artificial intelligence, reshaping industries and economies globally. Central to this transformation are Wireless Sensor Networks (WSN), which have evolved from their early applications, such as submarine tracking during the Cold War, to underpin the emerging smart cities of today. These advancements in WSNs have paved the way for the Internet of Things (IoT), which is now deeply embedded in industries like manufacturing and logistics. This new Industrial Internet of Things (IIoT) has a lot of promise, but with great potential comes great challenges. One of the biggest challenges is making sure these connected devices can communicate quickly, reliably, and effectively.

In this introductory chapter, we start by describing the thesis context and problematic in Section 1.1. We will first delve into the world of WSNs. These systems comprising small devices that wirelessly gather and share data. We will explore their challenges, such as limited power and computational constraints. Transitioning to the broader IoT landscape, we will discuss its rapid growth and diverse applications, from healthcare to industry. Then, we will focus on the IIoT, a specialized segment of the IoT tailored for industrial processes, highlighting the unique challenges it presents in managing intricate industrial systems with stringent real-time and reliability demands. Further, we detail the research motivations of this thesis in Section 1.2 before describing our scientific contributions in Section 1.4. Finally, the thesis structure is described in Section 1.5.

1.1 Research Context and Problem Statement

1.1.1 Wireless Sensor Networks

WSNs are distributed systems that consist of small, typically battery-powered nodes capable of collecting, processing, and communicating data from their environment via a wireless communication medium (Kandris et al., 2020). These nodes (also called motes) are generally equipped with a processor, a storage unit, a transceiver module, one or multiple sensors, an actuator, an

analog-to-digital converter (ADC) and a battery as a power source. They are deployed in wide areas to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, motion, among others.

The first application of WSNs was during the Cold War where the United States detected Soviet submarines by deploying the Sound Surveillance System (SOSUS) which used acoustic sensors (Jindal, 2018). The technological advances have broadened the range of WSNs applications to cover various sectors, including health, environment, urban planning and industrial automation.

Unlike traditional networks, WSNs are resource constrained due to a limited power capacity (battery-powered), short communication range, low bandwidth and limited processing and storage capacities. Consequently, a WSN is considered a Low-power and Lossy Network (LLN). These constraints necessitate that sensor nodes transmit packets over multiple hops to reach a Base Station (or sink node) which is responsible for relaying the collected information to the end user (data processing system, human users, other networks, etc). This scheme is called multi-point to point or *convergecast* traffic pattern and it is typical in many WSN applications. In this case, source nodes send data to the sink either periodically, upon detecting an event (event-driven) (Bouabdallah, Rivero-Angeles et Sericola, 2009) or upon receiving a query from the sink node (query-driven) (Sahar et al., 2021).

However, this does not exclude the existence of other traffic patterns in WSNs : *a) point to multi-point (P2MP)* where a node sends data to all the other nodes in the network (or in its radio coverage area), typically called broadcast (invitation to join the network, software updates, commands, etc), *b) point-to-point (P2P)* where a node sends data directly to another node, often employed for exchanging control or signaling packets for routing or synchronization purposes.

The evolution of WSNs and their subsequent integration with the internet served as the foundation for the emergence of the IoT (Mainetti, Patrono et Vilei, 2011). While WSNs are primarily designed to gather environmental data and are often deployed in remote or difficult-to-access regions, the IoT presents a wider landscape. This encompasses a network of physical objects, ranging from everyday appliances to wearable devices, all equipped with the capacity to connect to the internet. These devices, or 'smart' objects, are designed to gather and share data about their surroundings, thereby, offering the potential to enhance efficiency, automate tasks, and yield insights into user behaviors. The shift from localized WSNs to the vast, interconnected expanse of the IoT thus, represents a significant step into the digital transformation of our physical world by offering an interface between the physical and digital worlds.

1.1.2 Internet of Things

The International Telecommunication Union (ITU) in its Y.2060 recommendation (ITU, 2012) defines the IoT as "*a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving inter-operable information and communication technologies*". As per the projections from Fortune Business Insights, the worldwide IoT market is anticipated to escalate to a value of 3352.97 billion USD by the year 2030 (Insights, 2023). This growth trajectory, charted from the period of 2023 to 2030, suggests a Compound Annual Growth Rate (CAGR) of 26.1%.

In fact, the IoT novelty does not come from the functionalities present in its devices, called smart objects, but in its very large scale deployment where every physical object will be connected to the internet paving the way for the *smart planet* vision (Kopetz et Steiner, 2022). The IoT is applied in a variety of domains including :

— **Health-care** : to detect and prevent chronic medical conditions (Fafoutis et al., 2016),

for remote mobile medical monitoring (Ding, Gang et J. Hong, 2015), for cancer care services (Onasanya et Elshakankiri, 2021).

— **Environment** : to control the environmental factors such as temperature, humidity, CO_2 and NH_3 (Hua Li et al., 2015), for precision agriculture (J. Ye et al., 2013), concerning climate change, an ecological monitoring system is used to observe and understand its impact on wild vegetation communities (N.-S. Kim, Lee et Ryu, 2015).
— **Smart city** : for digital forensics (Zia, P. Liu et W. Han, 2017), for vehicular monitoring (Lingling et al., 2011), for Quality of Service (QoS) assessment of mobile crowsensing services (Distefano, Longo et Scarpa, 2015).
— **Industrial applications** : for real-time tracking and monitoring of intelligent workshop products (Chen, 2020), for predictive maintenance (Rosati et al., 2023), in supply chain management (Tan et Sidhu, 2022).

Wireless communication standards such as IEEE 802.15.1 (Bluetooth Basic Rate/Enhanced Data Rate) (IEEE, 2002) and IEEE 802.11 (Wi-Fi) (IEEE, 2016) are not suitable for WSNs due to the additional constraints brought by such networks. These constraints include lower power consumption, larger transmission ranges, scalability to support large-scale deployments, smaller data rates, among others. This has led to the proposition of new standards and protocols to meet the requirements and challenges of WSNs. The IEEE 802.15.4 standard in its 2011 version (IEEE802.15WG, 2011) defined PHY (Physical) and MAC (Medium Access Control) layer specifications for Low Rate Wireless Personal Area Network (LR-WPAN). The standard is characterized by a maximum data rate of 250 *kbps*, a Maximum Transmission Unit (MTU) size of 127 *bytes*. The PHY layer of the standard exploits the ISM (Industrial, Scientific and Medical) radio band (2.4 *GHz*) which includes 16 channels numbered from 11 to 25, where each channel has a bandwidth of 2 *MHz* and center frequencies are separated by 5 *MHz*. The standard caters to applications with low power and data rate requirements, finding wide usage in areas such as home automation, industrial automation, smart cities, and Wireless Body Area Networks (WBANs). Notably, established standards such as ZigBee (Safaric et Malaric, 2006), WirelessHART (Song et al., 2008), and ISA100.11a (ISA, 2011) integrate IEEE 802.15.4 at the PHY layer, modifying the upper layers as needed. The main features offered by this standard is the duty cycling in the MAC layer where a node sleeps by turning off its transceiver to conserve energy periodically. However, several studies pointed out multiple limitations of the standard, mainly :

— a high level of collisions due to the randomness of the binary exponent selection (Khanafer, Guennoun et Mouftah, 2014) ;
— the non-flexibility of the backoff algorithm for large-scale networks making the delay non-deterministic and unbounded (Koubaa, Alves et Tovar, 2006) ;
— the default values of the MAC parameters result in lower throughput and high power consumption (Pollin et al., 2008 ; Fourty, van den Bossche et Val, 2012) ;
— poor packet delivery ratio when power management is enabled (Anastasi, Conti et Di Francesco, 2011).

This makes it unsuitable for mission-critical application scenarios (industrial, healthcare, etc) that requires low latency, high reliability/throughput and low power consumption. To overcome the above-mentioned limitations, the IEEE 802.15.4e-2012 amendment (std., 2012) was proposed as an extension to the 802.15.4 standard by introducing several MAC behavior modes to meet the emerging requirements of industrial applications. These MAC modes have been inspired from existing standards targeting industrial applications such as WirelessHART (Song et al., 2008) and ISA 1001.11.a (ISA, 2011). The main features inspired from these standards include slotted access, shared and dedicated slots, multi-channel communication and channel hopping (Domenico De Guglielmo, Giuseppe Anastasi et Alessio Seghetti, 2014).

1.1.3 Industrial Internet of Things

Industry 4.0, often referred to as the fourth industrial revolution, encapsulates the integration of digital technologies within manufacturing and other sectors such as retail, logistics, oil and gas, and infrastructure (Oztemel et Gursev, 2020). Particularly, the fusion of IoT into Industry 4.0 gave birth to the concept of IIoT. At the heart of this industrial revolution is the notion of Cyber Physical Systems (CPS) in which IoT devices play an integral role, enabling the formation of "smart factories", where machines and production systems are interconnected and can communicate with each other. This intercommunication allows for the optimization of production processes, leading to cost reductions, efficiency improvements, and data-driven decision-making (I. Khan et Javaid, 2022).

The IIoT is a specialized segment of the broader IoT that focuses on the interconnection and automated exchange of information between machines (M2M communication), and it is particularly tailored for industrial processes and applications. It focuses on the manufacturing industry by connecting all the industrial assets, including machines and control systems, with information systems and business processes (Alabadi, Habbal et Wei, 2022). The IIoT shares common communication requirements with the general IoT, such as scalability and compatibility with low-cost, resource-constrained devices (limited storage and processing capabilities, low energy consumption requirements). However, it has additional specific needs due to the complexity of the systems it manages, such as availability, security and privacy, and superior QoS in terms of latency, throughput, reliability. Unlike IoT, which emphasizes novel communication standards for a flexible connection of devices, IIoT focuses on integrating and interconnecting formerly isolated plants, machines, and work areas for efficient production and intelligent services (Sisinni et al., 2018).

IIoT demands a fixed, well-structured network for high-stakes machine-to-machine links that meet stringent timeliness and reliability requirements. In applications like process automation, IIoT necessitates millisecond-level bounded delays and 99.99% transmission reliability (Åkerberg, Gidlund et Björkman, 2011). This is challenging for the IIoT since the wireless medium is unreliable, exposed to interference within the ISM band that it shares with other wireless technologies (e.g., Wi-Fi), and radio links quality is prone to degradation over time.

As previously mentioned, the IEEE 802.15.4e amendment (std., 2012) proposed various MAC modes to meet the requirements of industrial applications, among which, TSCH (Time Slotted Channel Access). The main application domains of TSCH include industrial automation and process control, specifically energy production applications, water management systems, food production plants, pharmaceutical production places, and oil and gas process plants. This protocol is a great potential to meet industrial requirements for timeliness, reliability and energy efficiency. It applies time slotting by adopting a TDMA (Time Division Multiple Access) (Falconer, Adachi et Gudmundson, 1995) scheme to eliminate collisions due to simultaneous transmissions from competing nodes within the same interference range. In addition, it employs channel hopping to mitigate external interference and multi-path fading effects [1].

On the other hand, Digital Twin (DT) is recently emerging as a promising technology for network optimization (Almasan et al., 2022). In this regard, specifically, the focus is on the concept of Network Digital Twin (NDT). As per the definition of the IETF (Internet Engineering Task Force) in their latest draft on this concept (Zhou et al., 2023), an NDT "*is a virtual*

1. Multipath fading in wireless networks refers to the phenomenon where radio signals reach the receiving antenna by multiple paths due to reflection, diffraction, and scattering, causing signal strength variations and potential interference at the receiver. This can lead to issues like fluctuating signal quality or temporary loss of connectivity.

representation of the physical network that has the purpose of analyzing, diagnosing, emulating, and then controlling the physical network based on data, models and interfaces". It is specifically designed to help achieve the goal of autonomous/self-driven networks. The NDT can help satisfy the requirements of the IIoT by enabling simulation, monitoring, and optimization of such networks (Yang et al., 2023). By effectively mapping the physical IIoT with its virtual twin network, historical data can be collected (e.g., topology data, configuration data, metrics data, etc) and exploited to construct models of the network useful for analysis and diagnosis of IIoT behavior and support for intelligent decision making.

1.2 Research Motivation

TSCH protocol ensures time slotting by establishing a periodic slotframe, composed of a number of timeslots, which the nodes adhere to throughout network operation. Moreover, it allows channel hopping, where nodes oscillate/hop among various frequency channels following a predetermined sequence while transmitting data. However, the standard does not define how the nodes are supposed to communicate, i.e., on which timeslot and what channel a node is supposed to transmit/receive data. This raises a need for a schedule in order to profit from the full potential of TSCH. A schedule can be modeled by a matrix whose rows are the available channels and columns are the timeslots in the slotframe. It has not been defined in the standard since the design of a schedule depends on the application requirements.

TSCH scheduling has garnered considerable interest among researchers, and numerous scheduling strategies have been proposed, either in a centralized or decentralized mode (Urke, Kure et Ovsthus, 2022). In the centralized schedulers, slot reservations for all nodes are orchestrated by a central entity, such as a Path Computation Element (PCE) or Software Defined Network (SDN) controller, which possesses a holistic view of the network. Subsequently, scheduling directives are dispatched to network nodes.

Conversely, in the decentralized mode, schedules are built without the requirement for a central component. Nodes, in this scenario, can negotiate cell allocations with their neighbors (distributed schedulers) or make scheduling decisions locally based on already available data, such as cross-layer routing information or node addresses, eliminating the need for additional negotiation overhead (autonomous schedulers). The responsiveness of decentralized schedulers renders them more fitting for large-scale, mobile, and dynamic networks, due to their superior capability to adjust to network alterations. However, centralized scheduling algorithms lack this level of flexibility (De Guglielmo, Brienza et Anastasi, 2016), as any change in the network necessitates a complete recalculation of the schedule, followed by its dissemination to all nodes in the network for schedule updates. This process imposes significant overhead and results in a slower responsiveness.

In an industrial context, a mix of different applications can generate heterogeneous traffic conditions. This can include the simultaneous operation of periodic monitoring and process control applications. Consequently, in these varied traffic conditions, some nodes may be traffic-intensive, while others generate low-rate traffic. This scenario is especially evident in Industry 4.0 environments. For instance, certain sensors may generate low data rate traffic to monitor environmental parameters of a shop-floor such as temperature, humidity, or gas levels. Meanwhile, nodes integrated into cameras or scanners in automated quality inspection systems might generate vast amounts of data, thereby, requiring high-rate data transmission. A further example includes sensors transmitting high-resolution visual data to create a 3D digital twin of an industrial component (He, Guo et Zheng, 2018), operating concurrently with sensors that relay

diagnostic data regarding the component performance and status. Decentralized TSCH schedulers have the potential to satisfy the requirements of such scenarios due to their superior responsiveness to network dynamics as opposed to centralized schedulers (Accettura et al., 2015a ; Aijaz et Raza, 2017).

A more recent trend in TSCH scheduling is the adoption of Reinforcement Learning (RL) (Sutton et Barto, 2018) to build the schedule. It is either directly implemented on the IoT motes, or remotely when Deep RL (Arulkumaran et al., 2017) is intended. Implementing Deep RL algorithms is computationally expensive and cannot be incorporated directly in the IoT motes due to their constrained nature. In the first case, there is a need to model the network environment but wireless communication is inherently uncertain and hard to be modeled. Model-free RL techniques can learn directly from experience without a model of the environment, making them potentially useful for wireless environments. Q-Learning is one such technique and can be adopted in such scenarios. Applying Q-Learning in the context of TSCH networks, specifically for scheduling communication, brings several advantages :

- **Model-Free Nature** : At its core, Q-learning does not require a predefined model of the environment (Watkins et Dayan, 1992). In wireless communications, accurately modeling the environment can be difficult due to unpredictable factors such as interference, fading, and node mobility. Q-learning learns directly from interaction with the environment, adjusting its strategy based on observed outcomes rather than relying on a potentially inaccurate or incomplete model.
- **Adaptability** : Q-learning is designed to explore and adapt. In the dynamic world of wireless communication where the network topology, interference levels, and data traffic patterns can change over time, the adaptability of Q-learning is a significant asset (Savaglio et al., 2019). It can continuously update its Q-values (which essentially represent the expected future rewards of actions in states) as it receives new data, allowing it to adjust its scheduling decisions as the environment changes.
- **Handling Uncertainties** : Q-learning employs an exploration-exploitation trade-off (Watkins et Dayan, 1992). This means it sometimes tries new actions (exploration) to discover their consequences, even if they are not the currently known best actions. This feature is beneficial in wireless communications where the "best" action (e.g., the best channel or time slot for transmission) can change unpredictably.
- **Reward-based Learning** : In Q-learning, actions are chosen based on expected future rewards. This is well-suited for TSCH scheduling where the goal is often to optimize certain metrics (e.g., throughput, latency, or energy efficiency). The Q-learning agent can be trained to learn and prioritize actions that lead to higher rewards, thus, better performance metrics.

However, in a TSCH network using Q-learning for scheduling, each node can be seen as an individual agent (Q-learning agent) making decisions to optimize its own scheduling. This decentralized decision-making approach means that nodes independently assess their environment and determine the best action to take, such as when to transmit data. In such a multi-agent system, coordination becomes critical. Without proper coordination mechanisms, the independent decisions made by each node can conflict with those of others (H. Park, Haeyong Kim, S.-T. Kim et al., 2020). For example, two nodes might decide to transmit on the same channel at the same time, leading to a collision. This is part of the more general problem of non-stationarity in multi-agent systems. While each node seeks to maximize its own performance or reward, it is essential to ensure these individual objectives align with the overall network's efficiency and reliability goals. The challenge, then, is coordinating the actions of all nodes in a way that the collective decisions lead to optimal network performance.

Nonetheless, multi-agents coordination for TSCH scheduling using Q-learning brings its own set of challenges. Firstly, the decentralized nature means there's a lack of a global view, making it difficult for nodes to predict the actions of their peers. This can lead to decisions that are locally optimal but globally suboptimal. Secondly, the dynamic nature of wireless networks, with changing channel conditions and network topologies, adds to the unpredictability of agent decisions. Thirdly, the communication overhead for exchanging information among nodes to achieve coordination can be significant, potentially leading to increased energy expenditure and reducing the efficiency gains expected from the Q-learning approach. Lastly, ensuring real-time responsiveness while coordinating decisions across numerous nodes can be computationally intensive, especially for resource-constrained devices. Overall, finding a trade-off between effective coordination among network nodes (Q-learning agents) and maintaining network efficiency without increasing energy consumption is a significant challenge.

Despite the advantages offered by distributed RL for scheduling, the challenges outlined earlier might limit significantly their suitability for extremely heterogeneous networks present in Industry 4.0 environments. These networks are characterized by heterogeneous QoS requirements, varied traffic patterns, among others. Despite their great responsiveness, decentralized schedulers cannot ensure globally optimal schedules because they lack a global view of the network. Moreover, they are not the best suited for traffic isolation or supporting various QoS requirements for different traffic flows. In contrast, while centralized scheduling algorithms may be less responsive than decentralized ones, they often excel in managing heterogeneous networks with various QoS requirements. This is due to their access to a holistic view of the network enabling the generation of globally optimized schedules. Furthermore, it allows per-flow scheduling so that end-to-end reliability and latency constraints are efficiently met (Orozco-Santos, Sempere-Payá et al., 2022 ; Veisi, Montavont et Theoleyre, 2023). We believe that combining the advantages of centralized scheduling with Deep RL would further enhance network performance and satisfy the strict requirements of Industry 4.0 applications. In addition to the advantages provided by centralized scheduling, deep RL have the potential to : *i)* mitigate the coordination problem in decentralized RL scheduling ; *ii)* excel at handling complex non-linear relationships, making them especially valuable in the IIoT where networking interactions are often non-linear ; *iii)* enhance the scalability of centralized schedulers ; *iv)* generalize to a wide range of networking scenarios via transfer learning, thus, great adaptability is ensured ; *v)* provide better responsiveness than traditional centralized schedulers because of their generalization ability and the possibility of continuously learn scheduling decisions based on the current state of the network. However, integrating centralized deep RL scheduling in the IIoT introduces significant challenges, such as providing the model with up-to-date and relevant training data, modeling the IIoT environment accurately, setting up a feedback loop for the RL agent to learn from the environment, and ensuring that the deep RL model generalizes well to different and unseen scenarios in the IIoT.

Thus, there is a need for a novel networking architecture able to adopt deep RL for scheduling. Specifically, Network Digital Twins are promising candidates for this purpose as they can allow real-time mirroring of the network and enable intelligent network management services, including resource allocation. In fact, an NDT can provide a safe, flexible, and cost-effective environment for developing, training, and validating Deep RL TSCH scheduling solutions. It can address many of the challenges associated with this approach by offering a virtual platform that closely mirrors the real-world network. This virtual environment facilitates rapid prototyping, abundant data generation for training, and the ability to test diverse network scenarios. Moreover, it can enable a balanced approach between online and offline learning, ensuring robust model performance. However, the NDT concept is still at its infancy, especially for the IoT where many challenges are still unexplored in the literature.

1.3 Research Aim and Questions

The purpose of this thesis is to address the challenges of the IIoT by exploring TSCH scheduling, RL-based schedulers and NDT for the IIoT. Thus, we define the following research questions :

1. How to address the non-stationarity problem of decentralized RL-based TSCH scheduling without increased energy consumption while ensuring that network performance is not compromised ?
2. How effective are decentralized TSCH algorithms, including autonomous, distributed, and RL-based ones, in managing highly heterogeneous traffic intensities typical in Industry 4.0 environments ?
3. How can the NDT concept benefit the IIoT and how to enable these advantages ?
4. In what ways does the Software Defined Networking paradigm facilitate the creation of Network Digital Twins within the Industrial Internet of Things landscape ?
5. How to model the IIoT network in the NDT while ensuring the model explainability and fault-detection capabilities essential for mission-critical industrial settings ?

1.4 Research Contributions

First of all, we begin our research with a critical literature review on Reinforcement Learning TDMA-based MAC scheduling in the IIoT as an effort to delineate the current state of research and identify research gaps. We provide a comprehensive synthesis on the most relevant RL techniques used for TDMA scheduling (including TSCH) and subsequently compare between them in terms of several unified features such as the scheduler main objective, the RL method used, advantages, limitations, etc.

Secondly, we provide an overview of the most relevant decentralized TSCH schedulers along with a performance evaluation under highly heterogeneous traffic conditions. More particularly, we classify decentralized schedulers into autonomous, distributed and RL-based ones. The most relevant ones from each category are then evaluated under a highly heterogeneous traffic scenario, i.e., considering the simultaneous coexistence of very high and very low data rates in the network. Subsequently, creating their energy consumption and network performance profiles to get insights into their relevance to applications handling such divergent traffic conditions.

Thirdly, we introduce QL-TSCH-plus, a novel Q-learning distributed scheduling algorithm for TSCH networks. It is an enhancement of an existing autonomous Q-learning scheduler called QL-TSCH. QL-TSCH-plus adapts the action peeking mechanism used to address the non-stationarity issue in the network to a distributed scheme. More specifically, instead of continuously listening to neighboring nodes communication, QL-TSCH-plus allows nodes to broadcast the learned transmission slots for updating the Action Peeking Tables. Moreover, the broadcast transmission slots are used to efficiently allocate reception slots. This way, QL-TSCH-plus reduces energy consumption by up to 47% compared to QL-TSCH while maintaining reliability and timeliness, making it suitable for the IIoT.

Fourthly, we explore the integration of Digital Twins with SDN for the IIoT. The main motivation is to permit a closed-loop network management across the whole network lifecycle, from the design to the service phase. To do so, we propose a holistic digital twinning architecture for the IIoT where the network is integrated along with other industrial components of the system. We employ SDN paradigm, a manifestation of network softwarization, in our

architecture. Specifically, the SDN controller establishes the link between each Digital Twin of the industrial system and its corresponding physical entity. We validate the feasibility of the proposed architecture in the process of choosing the most suitable communication mechanism that satisfies the real-time requirements of a Flexible Production System.

Fifthly, we design an in-depth NDT architecture based on SDN to enable intelligent management of the IIoT. This way, new networking services such as predictive maintenance, network diagnosis, resource allocation, energy optimization with other intelligent services can be efficiently integrated and exploited in the network life-cycle. We validate the proposed architecture by providing a promising prototype implementation that should unleash the full potential of the network digital twin. The implemented prototype enables communication scheduling using TSCH by integrating a centralized scheduling algorithm. Moreover, it includes a routing management service to ensure the routing function of the network.

Sixly, we provide an NDT architecture for the IIoT based on the open-source tools, Eclipse Ditto and Hono. By taking advantage of Eclipse Hono which allows an efficient connectivity for the network devices and Eclipse Ditto for representing the devices states in a digital form and also providing easy access to these states for the NDT. Thus, intelligent network services, including sustainability features, network diagnostics, and resource allocation, can be seamlessly incorporated and managed throughout the network lifecycle. We validate the proposed architecture by providing a resource allocation case study where we evidence how TSCH mechanism is exploited in our architecture.

Finally, we explore the construction of an NDT using Petri-nets with a particular focus on the IIoT, enabling data-driven Petri-nets for the IIoT. We present an architecture based on open-source tools as a framework to encourage researchers to investigate this research direction. Furthermore, we validate the proposed architecture through a case study involving a small-scale network modeled with Timed Petri-Nets. The results demonstrate the model's effectiveness in executing what-if scenarios based on the network's operational parameters to predict the Packet Delivery Ratio (PDR) and enable real-time fault detection. Lastly, we conduct a study on the IIoT observability by the NDT to address the challenge of virtual-real systems mapping in such a context.

1.5 Thesis Structure

This thesis comprises 6 chapters. After this introductory chapter, the following is a description of each chapter :

- **Chapter 2** : provides a comprehensive background on 6TiSCH protocol stack with a particular focus on TSCH protocol. Also, it presents an overview on some well-known TSCH scheduling algorithms categorized into centralized and decentralized ones. Then, it delineates the current state of research on RL-based TDMA MAC scheduling. Furthermore, it provides a comprehensive state-of-the-art on NDTs in terms of architecture and network modeling techniques for the NDT.
- **Chapter 3** : presents in the first part QL-TSCH-plus, our novel distributed scheduling algorithm for TSCH networks suitable for the IIoT. Secondly, it conducts an evaluation of decentralized TSCH schedulers including autonomous, distributed and RL-based ones under highly heterogeneous traffic conditions.
- **Chapter 4** : delves into the integration of DTs with SDN for the IIoT by proposing a holistic NDT architecture leveraging SDN to enable closed-loop network management across the entire network life-cycle. Moreover, it presents the NDT core architecture des-

cribing its main internal components and their interactions. The proposed architecture is validated through its application to the early design stage of an industrial project with real-time requirements.

— **Chapter 5** : proposes to leverage Petri-nets as a formal network modeling technique for the IIoT within the NDT context. An NDT architecture for the IIoT based on open source tools (Eclipse Hono and Eclipse Ditto) is presented to integrate Petri-nets. It conducts a case study to validate the proposed architecture and enable PDR prediction and real-time fault detection through this metric based on the IIoT operational parameters.

— **Chapter 6** : concludes by summarizing this thesis and opening up a set of perspectives.

Chapter 2

Background and Literature Review

TSCH has been proposed to alleviate the challenges and respond to the strict requirements of industrial applications. To enable the integration of IPv6 over TSCH, 6TiSCH Working Group (6TiSCH WG) defined a protocol stack composed of standardized protocols for each layer of the IoT. In this chapter, we bootstrap our work with an overview of 6TiSCH protocol stack. Adopting a bottom-up approach, we delve into the protocols at each layer, with a particular focus on TSCH protocol. Furthermore, we cover a set of TSCH scheduling algorithms including centralized, autonomous and distributed ones.

As highlighted in the preceding chapter, RL offers numerous benefits to TSCH scheduling, primarily because of its continuous interactions with the network environment. This not only ensures adaptability but also effectively manages the uncertainties inherent in wireless networks. To examine the current state-of-the-art of this research direction, we present a detailed review of the key RL-based techniques used in TDMA scheduling, including TSCH. Then, we compare them based on several criteria, such as their main objectives, the RL methods they employ, their strengths, and limitations. Furthermore, the need for adopting the DT paradigm is made evident to integrate Deep RL-based TSCH scheduling. Thus, we first define the DT concept before delving deeper into Network Digital Twining.

NDTs enable real-time network mirroring, paving the way for a new era of network management. However, it is still a new concept and brings its own set of challenges. For this reason, we tackle the research propositions in this context while focusing on the architectural aspect along with network modeling techniques for the NDT.

2.1 6TiSCH Protocol Stack

The IETF IPv6 over the TSCH mode of IEEE802.15.4-2015 Working Group has standardized a set of protocols to enable low power industrial-grade IPv6 networks by proposing a protocol stack integrated in TSCH mode (Thubert, 2021). 6TiSCH enables multi-hop topologies with RPL routing protocol (Routing Protocol for Low-Power and Lossy Networks) and supports IPv6 through 6LoWPAN. Moreover, it uses UDP (User Datagram Protocol) in the transport layer and the Constrained Application Protocol (CoAP) in the application layer.

Figure 2.1 depicts the 6TiSCH protocol stack consisting of five layers, we discuss each layer in the following sub-sections through a bottom-up approach.

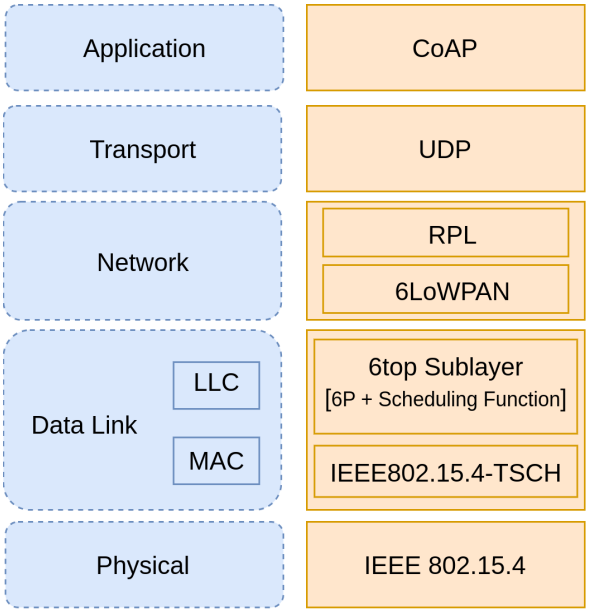


FIGURE 2.1 – 6TiSCH protocol stack

2.1.1 Physical Layer

The 6TiSCH protocol stack's foundation at the physical layer is the IEEE 802.15.4 standard (IEEE, 2020), which has been specifically designed for LR-WPAN. Here are some elaborated details :

1. *Frequency Bands* : While IEEE 802.15.4 caters to multiple frequency bands, 6TiSCH often uses the 2.4 GHz ISM band. However, its flexibility allows the protocol to operate in other sub-GHz bands as well, such as 868 MHz in Europe and 915 MHz in North America.
2. *Modulation and Data Rates* : For the commonly used 2.4 GHz band, the default modulation scheme is Offset Quadrature Phase Shift Keying (O-QPSK). This provides a data rate of 250 kbps, which is deemed sufficient for most low-power IoT applications.
3. *Channels* : In the 2.4 GHz band, IEEE 802.15.4 defines 16 separate channels, each with a width of 2 MHz. Additionally, only one channel is used when operating in the 868 MHz band while 10 channels are used in the 915 MHz band.
4. *Transceiver Operations* : At the physical layer, the tasks include turning the radio transceiver on and off, selecting the operating frequency, and handling the actual transmission and reception of raw symbol sequences. Modulation and demodulation of these symbols to convert them into meaningful data frames also occur at this layer.
5. *Energy Efficiency* : Given the primary deployment of 6TiSCH in environments with battery-operated devices, the IEEE 802.15.4's physical layer has been optimized for low power consumption. It supports features like clear channel assessment and link quality indication, ensuring reliable communication with minimal energy wastage.

By leveraging the robust foundation of IEEE 802.15.4's physical layer specifications, 6TiSCH ensures that IoT devices can maintain reliable wireless communication while operating under energy constraints.

2.1.2 Data Link Layer

The Data Link layer is divided into two sub-layers : MAC and Logical Link Control (LLC). The former manages the nodes access to the physical medium either in a contention-based or contention-free fashion. The MAC sub-layer of the 6TiSCH protocol stack is founded upon TSCH mode from the IEEE 802.15.4-2015 standard, which is a contention-free protocol. On the other hand, the LLC sub-layer acts as an intermediary layer between the MAC and the networking layer. In 6TiSCH, 6top Protocol (6P) is implemented as the LLC sub-layer by defining the framework to dynamically manage and schedule communication slots. Through 6top, devices can negotiate with their neighbors to add, delete, or relocate TSCH slots, adapting to the ever-changing needs of the network and ensuring efficient resource utilization. However, 6P doesn't inherently dictate how the cells should be allocated or used. This is where the Scheduling Function (SF) comes into play, it is responsible for deciding when and how to allocate or deallocate communication cells.

IEEE 802.15.4-TSCH

TSCH ensures the MAC function of 6TiSCH. First, it was proposed in (STD., 2012) and then integrated in IEEE802.15.4 standard in 2015. It is tailored for process automation, spanning sectors like oil and gas, green energy production, and climate control, among others (IEEE, 2020). At its core, TSCH merges the benefits of time-slotted access with multi-channel capabilities, enriched by channel hopping. This time-slot technique boosts the network throughput by averting collisions between nodes and guarantees predictable latency. Using multiple channels means nodes can communicate simultaneously by picking different channel offsets, increasing the overall network capacity. Channel hopping, on the other hand, mitigates the effects of interference and multi-path fading. As a result, TSCH offers a robust network with enhanced capacity, reliable communications, and predictable latency. What's more, its energy efficiency is commendable due to the time-slotted approach. Its flexible design can adapt to various network topologies, from simple star configurations to complex mesh setups, making it ideal for multi-hop networks where efficient resource use is crucial.

Slotframe Structure. In TSCH, nodes align to a recurring slotframe, which is broken down into several timeslots of equal length. The timeslot length should be sufficient for a data packet and its acknowledgment to be exchanged between a transmitter and a receiver. Moreover, a node may transmit, receive a packet or switch off its radio (sleep to save energy) at each timeslot. Nodes gather synchronization, channel hopping information, and details on timeslots and slotframes from Enhanced Beacons (EBs). These EBs are periodically broadcasted by nodes who already joined the TSCH network to signal its presence. After joining the network upon receiving an EB, the slotframe is repeated during the whole networking operation and synchronized based on nodes shared notion of time.

Synchronization. Due to factors like manufacturing variances, temperature fluctuations, and supply voltage differences, the clocks in different nodes can vary slightly in frequency, leading to a phenomenon called "clock drift". As a result, nodes need to realign themselves periodically. Each node is paired with a time-source neighbor for consistent synchronization, even though the specifics of selecting this neighbor aren't explicitly laid out in the 802.15.4 standard. Nodes can use either *Frame-based* or *ACK-based* synchronization for this purpose. In *Frame-based synchronization*, whenever a node receives a data packet from its time-source neighbor, it marks

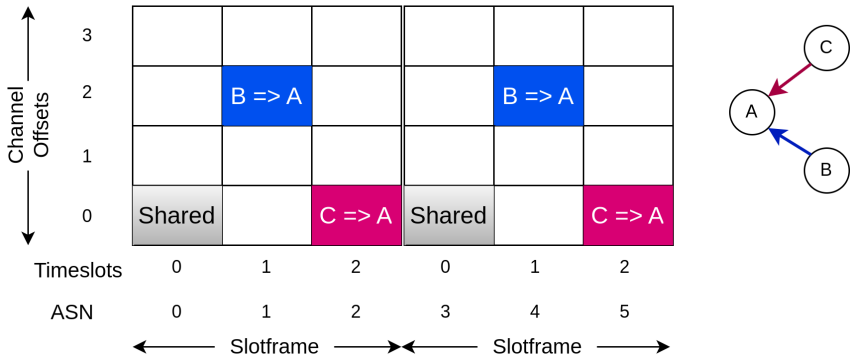


FIGURE 2.2 – Example of a TSCH schedule for a simple network of 3 nodes (A, B and C).

the reception start time. Knowing the TsTxOffset (the waiting time inside a timeslot before transmitting a packet), the node adjusts its timeslot boundaries to align with its time-source. In *ACK-based synchronization*, when a node transmits a packet to its time-source neighbor, the latter notes the reception start time and includes this timestamp in the acknowledgment. The sending node then utilizes this timestamp to re-calibrate its clock.

Schedule. Nodes in TSCH follow a schedule which dictates when a node will transmit, receive of turn off its radio. This schedule can be modeled by a matrix whose rows are the available channels and columns are the timeslots in a slotframe, as depicted in Figure 2.2. Each cell of the matrix represents a specific link having as coordinates (Channel Offset, Timeslot). We differentiate two types of cells in TSCH : i) *Shared cell* that is shared between multiple transmitters (cell allocated at timeslot 0, channel offset 0 in Figure 2.2), when an acknowledgement is not received for a packet, a specific CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance) algorithm is executed by selecting a random backoff before the next retry (more details are provided in TSCH CSMA-CA paragraph below) ; ii) *Dedicated cell* is allocated to a single node for transmission/reception (cells (1, 2) and (2, 0) in Figure 2.2).

Channel Hopping. A defining feature of TSCH is its use of multi-channel communication, which leverages channel hopping to combat noise and interference. At the beginning of each timeslot, the channel offset is translated into a frequency (physical channel) in which two nodes can communicate as follows :

$$f = F[(ASN + channelOffset)\%N_{channels}] \tag{2.1}$$

where ASN is the total number of timeslots that have elapsed since the start of network service, incremented in each timeslot, *channelOffset* is the channel offset of the current cell and $N_{channels}$ is the number of available channels. Function F can be implemented as a look-up table, usually it is defined as the hopping sequence specified in TSCH. For example if 4 channels are used, the hopping sequence can be $\{15, 20, 25, 26\}$. It should be mentioned that Equation (2.1) can return different frequencies for the same link in different timeslots, which ensures the channel hopping mechanism. Over time, this ensures that every available channel is utilized for link communications, effectively minimizing the impact of external interference (DE GUGLIELMO, BRIENZA et ANASTASI, 2016).

TSCH CSMA-CA. When using shared links, multiple transmitters can send data simultaneously, leading to potential collisions and transmission failures. The standard outlines a CSMA-CA retransmission method to minimize the likelihood of repeated collisions. Here's how it works : If a transmitter node t has a data frame for a receiving node r, it waits until the next available link (either dedicated or shared) for (t, r) becomes accessible. If the transmission on a shared link is unsuccessful (indicated by no acknowledgment), it's probable that a collision took place. The node t then executes the CSMA-CA algorithm to avoid subsequent collisions. The procedure node t follows is :

1. Initialize state variables : the number of attempted retransmissions for the current frame ($NB = 0$) and the backoff exponent ($BE = macMinBE$).
2. Generate a random number w within the range $[0, 2^B E - 1]$.
3. Delay the frame retransmission for w shared links directed to r or until a dedicated link to r is found.
4. For retransmission on a shared link : if successful (acknowledgment received), BE resets to $macMinBE$ and the process ends. If not, the state variables update : $NB = NB + 1$, $BE = min(BE + 1, macMaxBE)$. If NB exceeds the maximum retry limit ($macMaxFrameRetries$), the frame gets discarded, and the algorithm revisits step 2.
5. For a successful retransmission on a dedicated link, BE resets to $macMinBE$, but if there are more frames awaiting transmission to the same receiver, BE retains its current value.

This mechanism ensures that in the event of transmission failures, the sender node makes informed decisions on when and how to retransmit, optimizing the use of shared links and minimizing collisions.

The following are the main differences between TSCH CSMA-CA and the original 802.15.4 CSMA-CA :

— *Backoff Mechanism* : In the 802.15.4 CSMA-CA, nodes with packets ready to send wait for a random backoff time to prevent simultaneous transmissions and potential collisions. On the other hand, TSCH CSMA-CA initiates the backoff only after a collision has happened, aiming to prevent subsequent collisions.

— *Backoff Unit Duration* : Both algorithms define a backoff unit, where nodes wait for a random number of backoff units before attempting a retransmission. In the 802.15.4 CSMA-CA, this duration is 320 μs. However, TSCH defines it as a shared slot, ensuring collisions only if multiple nodes access the same slot simultaneously. This differs from 802.15.4 CSMA-CA, where packets might collide even if they are transmitted at different times.

— *Clear Channel Assessment (CCA)* : In the 802.15.4 CSMA-CA, nodes conduct a CCA to verify the channel's condition before sending a packet. This helps in avoiding collisions with ongoing transmissions. In TSCH, CCAs are not primarily for collision prevention since all nodes are synchronized together, making simultaneous transmissions unlikely. Instead, it's to detect strong external interference. Additionally, TSCH makes CCAs optional.

— *Packet Dropping* : In the 802.15.4 CSMA-CA, packets are discarded by the sending node if the channel is continually occupied for macMaxCSMABackoffs times. In contrast, TSCH CSMA-CA drops a packet only if it exceeds the max number of retransmissions, indicated by the macMaxFrameRetries parameter.

6P and SF

6P is defined in RFC8480 (Q. Wang, Vilajosana et Watteyne, 2018). It is a cornerstone component of the 6TiSCH architecture, operates within the MAC sublayer to govern dynamic cell management in a TSCH context. 6P's primary function is to provide mechanisms that allow nodes to collaborate and orchestrate the allocation, addition, or deletion of TSCH cells, based on evolving network conditions and application requirements. Crucially, the protocol facilitates bilateral negotiations between neighboring nodes to determine optimal cell allocation, optimizing link-layer resource utilization and mitigating potential contention.

6P defines two types of cells, *Soft Cells* which are dynamically managed by the 6top protocol. Their allocation and deallocation are a direct result of the 6P commands, making them flexible and adaptive to the current network conditions and demands. Soft cells are typically used to manage data flows that might vary over time and managed by SFs. Contrary to soft cells, *Hard Cells* have a fixed schedule and are preconfigured, meaning their timeslots and channel offsets are static and aren't subject to dynamic adjustments by the 6top protocol. These cells are not managed by 6P but can be used when hard-coding a schedule as in the Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration (Vilajosana, Pister et Watteyne, 2017).

The main commands defined in 6P are :

— ADD : Used to add one or more cells. The receiving node either confirms the addition or denies the request based on its local policy or available resources.
— DELETE : Enables a node to remove specific cells. The receiving node confirms or denies based on its current scheduling.
— RELOCATE : A command that signifies shifting of a cell from one timeslot or channel offset to another. It's vital for optimizing the distribution of cells and ensuring effective channel utilization.
— LIST : Allows nodes to request a list of scheduled cells from its neighbors. It's instrumental in understanding the current scheduling status.
— CLEAR : This command instructs the receiving node to remove all the scheduled cells between the sender and the receiver. It's a kind of "reset" command for the cell schedule between two nodes.

These commands are executed through 6P transactions, which can be either two-step or three-step message exchanges between negotiating nodes. A two-step DELETE transaction between a transmitter A and a receiver B proceeds as follows :

1. Step 1 (Request) : Node A sends a 6P DELETE command to Node B, indicating the cells it wishes to delete.
2. Step 2 (Response) : Node B processes this request and then replies with a 6P Response indicating the successful deletion or any encountered error.

In order to elaborate the concept of three-step transactions, let's say Node A wants to add new soft cells for communication with Node B, but it doesn't have a specific cell in mind. It wants Node B to suggest some available cells. Figure 2.3 illustrates the three-step transaction associated to this example.

— Step 1 (Request) : Node A sends a 6P ADD Request command to Node B, specifying the number of cells it wants but not specifying the exact cells.
— Step 2 (Follow-Up) : Node B processes the request, identifies available cells that can be allocated ((1,1), (4,3) and (3,2)), and replies with a 6P Response listing the proposed cells for Node A use.

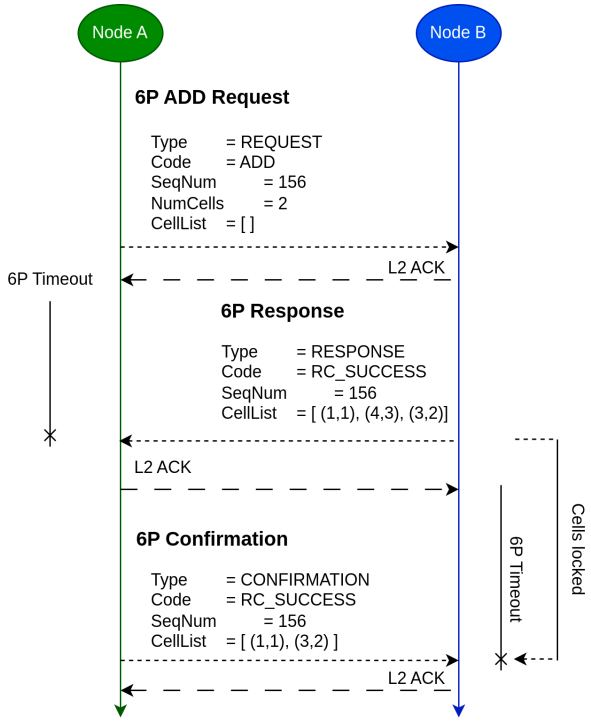


FIGURE 2.3 – 6P three-step transaction

— Step 3 (Confirmation) : Node A examines the proposed cells, selects two our of the three proposed cells by node B ((1,1) and (3,2). Then confirms the allocation by sending a 6P Confirmation to Node B, thereby finalizing the addition of the new cells.

These examples highlight how the 6top protocol accommodates different operational needs, ensuring flexibility and collaboration between nodes in the 6TiSCH network.

The cells allocation policy is defined by the SF which launches the 6P transactions for setting up and maintaining the communication schedule according to application needs and network traffic. Since the scheduling in this case requires negotiating cells allocation, it is distributed. Many distributed SF have been proposed in the literature but only one is under standardization by the IETF, the 6TiSCH Minimal Scheduling Function (MSF) defined in RFC9033 (CHANG, VUČINIĆ, VILAJOSANA, DUQUENNOY et al., 2021). We will cover it later along with other well-known distributed SFs in Section 2.2.2.

2.1.3 Network

6LoWPAN

The rise of the IoT demands a vast number of unique addresses to connect billions of devices to the internet. IPv4 limited address space cannot accommodate this surge. Therefore, IPv6, with its expansive address capacity, is the best choice for IoT. Moreover, IPv6 ensures direct communication, improved security, and easier device setup, positioning it as a fundamental enabler for IoT growth. While IPv6 offers a vast addressing space, its headers, designed for the traditional internet, are relatively large (40 bytes) and inefficient for LLNs where the MTU is limited to 127 bytes. These constraints motivated the introduction of 6LoWPAN, standing for "IPv6 over Low-Power Wireless Personal Area Networks". It provides a mechanism to compress IPv6

headers, making IPv6 packets more suitable for transmission over LLNs. A closer examination reveals several significant features and components :

1. *Header Compression (6LoWPAN HC)* : Arguably the most vital aspect of 6LoWPAN is its ability to compress the typically 40-byte IPv6 headers. This is achieved through 6LoWPAN HC, which reduces the header size considerably. Given the limited bandwidth of many IoT networks, this compression is essential, making data transmission more efficient by minimizing header overhead.
2. *Fragmentation and Reassembly* : Since IoT devices, especially those adhering to the IEEE 802.15.4 standard, have a maximum packet size constraint (typically up to 127 bytes), there arises a challenge when transmitting larger IPv6 packets. 6LoWPAN addresses this by fragmenting these larger packets at the source and reassembling them at the destination. This ensures that data can be reliably transmitted over the network without violating packet size restrictions.
3. *Routing Capabilities and 6LoWPAN Routing Header (6LoRH)* : Beyond compression and fragmentation, 6LoWPAN introduces enhanced routing capabilities. The 6LoWPAN Routing Header is used to facilitate this, allowing the protocol to support various topologies, from star to mesh configurations. This header is especially beneficial in multi-hop environments, offering devices flexible routing options.
4. *Mesh Networking* : 6LoWPAN's design permits a mesh-under routing approach. By allowing IoT devices to form mesh network topologies, multiple paths for packet routing are provided, ensuring increased resilience and adaptability within the network.
5. *Adaptability to Various Network Scenarios* : 6LoWPAN isn't a one-size-fits-all solution. Instead, it's modular, meaning that specific components (like certain header compression mechanisms or routing strategies) can be implemented based on the unique needs of the IoT environment.

RPL

RPL, or the Routing Protocol for Low-Power and Lossy Networks, is a specialized routing protocol standardized by the IETF under RFC 6550 (Thubert et Winter, 2021) to serve the unique challenges presented by IoT and other low-power, loss-prone environments. The fundamental principle behind RPL is the creation of a Directed Acyclic Graph (DODAG), which is rooted at a specific node, typically a border router that provides connectivity to other networks.

1. *DODAG Structure* : Within this structure, data is routed upwards to the root before being directed downwards to its intended destination. This hierarchical approach ensures efficient multi-hop routing, even in environments where network links are unstable.
2. *Operational Modes* : RPL operates in two modes :
   — Storing Mode : Nodes within the network maintain their own local routing tables. This distributed approach allows for efficient downwards routing from any node to a lower one in the DODAG hierarchy.
   — Non-Storing Mode : Contrarily, in this mode, only the DODAG root possesses a full view of the network topology. When routing, the root appends a source route to packets, detailing the path the packet should take. Given the memory constraints of many IoT devices, this mode is often more practical.
3. *ICMPv6 Messages* : RPL operation leverages four specialized ICMPv6 messages to manage and maintain the DODAG :

— DIS (DODAG Information Solicitation) : Utilized by nodes to request routing data.
— DIO (DODAG Information Object) : Employed to construct the DODAG from the root, allowing nodes to disseminate routing information.
— DAO (Destination Advertisement Object) : Enables nodes to ascertain and establish reachability paths back along the DODAG.
— DAO-ACK : Acts as an acknowledgment for DAO messages, ensuring reliable route registrations.

4. *Joining and Maintenance* : A node aiming to join the network sends a multicast DIS to solicit DIO messages. Once it receives a DIO, it decides whether to join that particular DODAG instance. If it joins, it then selects a parent based on the chosen objective function and advertises its position using a DAO message sent to the root. Continuous link quality monitoring and parent re-selection ensure that the network topology remains efficient, adapting to changing conditions in real-time.

5. *Objective Function* : While RPL offers a general approach to distance vector routing, it can be tailored for specific needs through Objective Functions (OFs). These OFs adapt the standard RPL operation to particular scenarios, taking into account metrics like latency, delivery ratio, or redundancy requirements. IETF standardized two OFs : OF0 (Thubert, 2012) and MRHOF (Gnawali et Levis, 2012), each catering to different metrics and constraints. For instance, 6TiSCH uses OF0 with the ETX (Expected Transmission Count) metric. By allowing different objective functions, RPL provides adaptability for diverse use cases.

In summary, RPL is a dynamic, adaptable, and efficient routing protocol for the challenging environments typical of IoT deployments. Its design emphasizes flexibility, allowing it to serve a wide range of application scenarios including industrial ones.

2.1.4 Transport and Application

The 6TiSCH transport layer utilizes UDP. It offers a lightweight, connectionless communication scheme that is apt for IoT settings, where swift data transmission often takes precedence over guaranteed delivery.

Transitioning to the application layer, CoAP is the protocol of choice in the 6TiSCH stack. Designed specifically for resource-constrained devices and networks, CoAP is a web transfer protocol. It is standardized in RFC7252 (Shelby, Hartke et Bormann, 2014) It enables devices to interact with web resources efficiently, making it a perfect fit for the limited capacities of IoT devices. CoAP runs over UDP, and when security is essential, it can operate over Datagram Transport Layer Security (DTLS). It mirrors HTTP (HyperText Transfer Protocol) request/response model, facilitating methods like GET, POST, PUT, and DELETE, but differentiates by employing a compact binary header format, optimizing it for bandwidth-constrained networks. Instead of constant polling, CoAP observe relationships allow devices to be automatically notified of changes. In fact, in traditional web systems, if a device wants to know if something has changed on the server, it would have to repeatedly ask or "poll" the server. This is like constantly asking someone, "Are we there yet ?" on a long car ride. This constant checking consumes a lot of resources and energy, which is not suitable for contrained IoT devices. CoAP, however, offers a smarter alternative with its "observe" feature. Additionally, CoAP handles large payloads by splitting them into manageable blocks for transmission. To enhance network efficiency, CoAP supports proxies that can cache responses, benefitting situations where multiple devices request

identical data. Overall, CoAP combines the functionalities of web protocols like HTTP with design considerations unique to the constrained environments of IoT.

In essence, the 6TiSCH stack's selection of UDP and CoAP underpins a balance between simplicity and performance, catering to the unique demands of IoT communications.

2.2 TSCH scheduling

Scheduling communication in TSCH has gained a huge interest from researchers since its proposition in the 802.15.4e-2012 amendment (STD., 2012). This latter left the schedule specification an open research area since it depends on the applications requirements. Various TSCH scheduling algorithms have been proposed in the literature since then which can be coarsely classified into centralized or decentralized strategies (URKE, KURE et OVSTHUS, 2022). We start by discussing some existing centralized schedulers before detailing well-known decentralized schedulers that we further classify into autonomous and distributed ones.

2.2.1 Centralized TSCH scheduling

Typically, a centralized approach employs a single PCE to generate and distribute schedules using a specific scheduling algorithm. Moreover, an SDN based architecture can be adopted for centralized scheduling by having an SDN controller instead of the PCE (OROZCO-SANTOS, SEMPERE-PAYÁ et al., 2021). While this setup allows for highly optimized schedules based on detailed network information, such as node capabilities, wireless link properties, and physical and routing topologies, it faces challenges. Specifically, adapting to frequent changes in network conditions—such as mobile nodes, fluctuating radio environments, or evolving application requirements—may introduce significant overhead and delay timely accommodations (KARAAGAC, MOERMAN et HOEBEKE, 2018).

TASA (PALATTELLA, ACCETTURA et al., 2012) is a centralized, multi-hop scheduling algorithm designed for tree-like network topologies where each node has a singular parent and network traffic is forwarded from leaf nodes towards the sink. The algorithm prioritizes nodes with higher traffic loads and aims to create compact schedules, minimizing the last cell used in the slotframe. It employs two key algorithms : matching and coloring. The matching algorithm identifies a set of eligible links that can be scheduled simultaneously within a single time slot. The coloring algorithm, on the other hand, assigns distinct channel offsets to each selected link in the matching, thereby minimizing channel conflicts. In this manner, TASA allocates bandwidth primarily to the most constrained nodes while effectively utilizing different channels to resolve link conflicts.

MODESA (Multichannel Optimized DElay time Slot Assignment), proposed in SOUA, MINET et LIVOLANT, 2012, considers the same assumptions as TASA but uses several radio interfaces at the sink level. But unlike its predecessor, MODESA reduces queue congestion by first scheduling the nodes that have more packets in their buffer. The algorithm builds the scheduling slot by slot by applying the following six rules :

1. Each node is assigned a dynamic priority equal to $remPckt * parentRcv$ where : i) *remPckt* is the number of packets a node has during the current iteration ,ii) *parentRcv* is the total number of packets the parent node must receive during a cycle.
2. Nodes compete for the current time slot if and only if they have data to transmit.
3. A node and its parent must have an available interface.

4. For any time slot, the first node scheduled is the node with the highest priority among all nodes with data to transmit. If two nodes have the same priority, MODESA selects the node with the smallest identifier. The selected node is set to the first channel c.
5. Any node can be scheduled in the current time slot on channel c if it does not interfere with nodes already scheduled on channel c in this slot.
6. Conflicting nodes that interfere with nodes already scheduled in this slot are scheduled on a different channel.

AMUS (Adaptive MUlti-hop Scheduling), as proposed in Jin et al., 2016, utilizes CoAP to gather essential scheduling information. It employs a unique approach that reserves communication resources for each set of end-to-end links, enabling effective multi-hop scheduling. The algorithm is sensitive to factors such as interference and collisions and optimizes resource allocation based on traffic demands. This includes aggregating multi-hop traffic at relay nodes. AMUS also introduces an innovative mechanism to manage retransmissions efficiently. Traditional scheduling algorithms often suffer from delays when packets must be retransmitted due to poor link quality ; these retransmissions consume additional time slots, which can impact subsequent packets in the slotframe. To mitigate this, AMUS deploys provisional scheduling that pre-allocates spare cells for active links. This enhances communication reliability and significantly reduces latency.

These algorithms are designed to be centralized where the schedule is built in a PCE. Recently, the rise of SDN solutions for WSNs such as SDN-WISE (Software Defined Networking solution for WIreless SEnsor Networks) (Galluccio et al., 2015) is paving the way for highly efficient centralized scheduling. Specifically, SDN WISE-TSCH has been proposed in (Orozco-Santos, Sempere-Payá et al., 2021) by integrating TSCH scheduling in SDN-WISE framework. It allows the differentiation, isolation and logical segmentation of data flows, by means of the allocation of TSCH resources. This guarantees that QoS requirements are met for each data flow independently. A more recent approach in this context is SDN-TSCH (Veisi, Montavont et Theoleyre, 2023), an SDN solution designed for scheduled wireless networks with traffic isolation, the SDN controller sets up a collision-free control plane. This enables devices to connect reliably to the controller even over lossy wireless links. An innovative link quality estimation approach is proposed, which doesn't require the generation of additional control packets to create an accurate network topology. Utilizing this quality estimation, the scheduler allocates cells specifically for EBs to avoid collisions. Leveraging this precise information, the SDN controller then calculates a compact schedule that meets end-to-end constraints.

2.2.2 Decentralized TSCH scheduling

Autonomous Schedulers

Orchestra (Duquennoy et al., 2015) is one of the most popular autonomous scheduling algorithms. It builds schedules for each traffic plane (MAC, routing, and application) and maintains them without any overhead communication, mainly using a hash function of the node's ID or its MAC address. It calculates the schedules locally based on the existing topology from the upper layer (exploiting RPL). It has been exclusively developed for TSCH+RPL networks which operate with low-power IPv6 networks. Despite being the first fully autonomous scheduler proposed in the literature, Orchestra is not traffic adaptive since the number of transmission slots does not depend on the traffic volume flowing in the network. Also, collisions may still occur during some of the frames since the cell is allocated only according to a hash of the MAC address. In (Duquennoy et al., 2015), Orchestra is compared with MSF (Chang, Vučinić, Vilajosana, Duquennoy et al., 2021), ContikiMAC (Adam Dunkels, 2012) and Always-on MAC (CSMA)

mode of Contiki-OS (A. Dunkels, Gronvall et Voigt, 2004). The simulation setup is simple and can evaluate the protocols for general-purpose applications. There are two-directional types of traffic : *sink-to-node* and *node-to-sink*. Packets are generated every 60 seconds and a real testbed is deployed to test the protocols. The results showed that Orchestra can perform as well as the other protocols while keeping the network synchronized and at low energy consumption. However, the authors did not test the protocols with different types of traffic flows and they only consider a relatively low data rate (1 packet per minute). Currently, Orchestra has three versions : Sender-Based (SB), Receiver-Based (RB) Storing, and RB Non-storing.

Another popular autonomous scheduler called "Autonomous Link-based Cell Scheduling" (ALICE) is proposed in S. Kim, H.-S. Kim et C. Kim, 2019 and widely investigated and studied in Elsts et al., 2020 with some amendments (node-based channel allocation). ALICE is an autonomous scheduling protocol that deals with resources as links, it allocates cells (timeslots) for a one-directional link between two neighbors (Link-Based scheduling). In fact, there are one or more Rx slots and one or more Tx slots for each link in LB scheduling contrarily to NB scheduling where a node selects a slot for himself based, mostly, on the hash of its MAC address. Like any other autonomous scheduler, ALICE takes advantage of the existing RPL routing topology information. Unlike Orchestra, it minimizes collisions since the hash function used for allocating cells exploits time as an input, represented by the Absolute Slotframe Number, in addition to the link node IDs and the direction of traffic. The hash output in this case has a bigger chance of being different for each slotframe compared to Orchestra's hash output. This way, even if collisions occur in a slotframe, they are almost eliminated in the next slotframe upon recalculating the schedule thanks to the ASN-based scheduling adopted by ALICE. Moreover, the upstream and downstream traffic are differentiated in ALICE and they receive separate resources. The scheduler performance evaluation conducted in S. Kim, H.-S. Kim et C. Kim, 2019 ; Elsts et al., 2020 prove that ALICE outperforms Orchestra in all aspects : reliability, throughput, latency, routing stability, and energy consumption. However, it still lacks the adaptive feature since cells are not allocated according to the traffic rate in the network.

To overcome the main limitations of Orchestra and ALICE, i.e., the absence of the traffic adaptive feature, Jeong et al., 2020 proposed an autonomous scheduler called "On-Demand TSCH Scheduling with Traffic Awareness algorithm (OST)". This protocol allocates resources based on the nodes traffic load. One of its best features is that it can dynamically adjust the frequency of timeslots following the demand in real time. This allows nodes to send their long-queued data in a short time. In this paper, OST is tested against Orchestra and ALICE on a real testbed consisting of 72 nodes (FIT/IoT-LAB). They also generated traffic in two directions (upstream and downstream) but with a low data rate reaching 5 packets/s at maximum for the whole network. The results showed that OST can increase reliability by 60% and energy efficiency by 52% compared to the state-of-the-art protocols (ALICE, Orchestra).

Distributed Schedulers

There is a scheduling function that is being standardized by IETF and developed by the 6TiSCH Working Group, called MSF, proposed in RFC9033 (Chang, Vučinić, Vilajosana, Duquennoy et al., 2021). This protocol is considered a distributed approach and is designed for generic use cases. It is implemented on top of 6P and considers three types of cells : minimal, autonomous, and negotiated cells. The minimal cells implement the 6TiSCH minimal configuration that is used for the bootstrap traffic, including a single shared slot for exchanging EB packets and RPL signaling packets. The autonomous cells are dedicated to unicast communication. Each node schedules permanently an AutoRxCell (autonomous reception cell) for reception. A node

also adds/removes AutoTxCells on demand when there is a frame to be sent and no negotiated cell is allocated for that frame. Finally, the negotiated cells are allocated for a node with its selected parent and they have an adaptive nature. A node can add/delete/relocate negotiated cells with its parent either to match the traffic conditions, to handle switching its parent, or to handle a schedule collision. MSF is regarded as RPL-dependent, which means that an error in RPL can result in network failure. Furthermore, the main drawback of distributed schedulers, and therefore of MSF, is the additional overhead created for negotiating cells allocation between the network nodes. Chang, Vučinić, Vilajosana, Dujovne et al., 2020 evaluate the performance of MSF and state that with the right configuration, MSF can keep up with other state-of-the-art autonomous scheduling protocols. In addition, Righetti et al., 2020 compares MSF with On-The-Fly (OTF) (Palattella, Watteyne et al., 2016) protocol and states that MSF suffers in performance due to the failures in the 6TiSCH stack (mainly RPL configuration).

Decentralized Traffic Aware Scheduling (DeTAS) (Accettura et al., 2015a) is one of the first proposed distributed schedulers for TSCH. It uses micro-schedules to build the global schedule where each micro-schedule uses a dedicated set of channels. In fact, each node demands bandwidth from its parent which in turn calculates the number of packets that it will receive from its children and its own local traffic, then transmits this information recursively until it reaches the sink node. This latter starts the cells allocation by scheduling the timeslots to receive the traffic from each child. On the other hand, the allocation of transmission slots in a node is done in such a way as to be synchronized with the reception slot of its parent. In particular, the transmission and reception cells are alternated along the path to the sink, i.e, if a type of cell is scheduled in an even slot the other type should be scheduled in the next odd slot (Accettura et al., 2015b). In simple words, a packet received during a slot is transmitted to the parent in the next slot. This allows to overcome the hidden terminal problem, avoids buffer overflow and keeps queue utilization as low as possible. Simulation results of DeTAS show better queue management compared to TASA (Palattella, Accettura et al., 2012), which is considered as its centralized version, making it resilient to traffic load increment for all nodes in the network. In contrast, its main drawback is that if a packet is lost due to poor radio link quality, all the subsequent slots scheduled on the parent nodes to forward this packet are wasted. In addition, the same limitation as its centralized version (TASA) is present, i.e., when the network topology or traffic changes, the schedule procedure has to start over to still match the latency requirements. Moreover, collisions may occur if other links exist in addition to the convergecast links (Soua, Minet et Livolant, 2014).

Soua, Minet et Livolant, 2015 proposed a distributed interference and traffic-aware scheduler for TSCH called DiSCA (Distributed Scheduling for Convergecast in multichannel WSNs Algorithm). It tries to minimize the number of timeslots needed to transmit all the packets to the sink. In fact, the algorithm operates iteratively by allocating at step i the i^{th} transmission of each node. The priority in terms of scheduling is given to the node which has the highest number of packets to transmit. Each iteration results in a micro-schedule that can overlap to reduce the total number of slots. Moreover, a transmitting node notifies all its conflicting nodes upon picking the first available pair of timeslot and channel offset. DiSCA takes into account the case where the sink is equipped with multiple radio interfaces and considers different traffic loads generated by the nodes : experiments have been conducted considering that each node generates a number of packets randomly drawn in $[1, 5]$ per slotframe. According to the simulation results in the original paper, DiSCA outperforms Wave (Soua, Minet et Livolant, 2014) which was proposed previously by the same authors, in terms of the number of required slots in different configuration setups. Also, it gives a close performance to the optimal schedule with reduced control messages. In contrast, the drawback of such an approach resides in notifying the reser-

vations to the interfering nodes which is a complicated task in practice. The set of conflicting nodes has to be precisely identified and estimating the interfering nodes is still an open challenge. Furthermore, no mechanism tries further to detect and solve the inconsistencies (collisions).

2.3 RL-based TDMA MAC schedulers

RL is nowadays applied in various application domains, underpinning its versatility and robustness. It has showcased its transformative potential in several key domains, demonstrating its capability to revolutionize both recreational and critical real-world applications. In the realm of robotics, RL has empowered machines to autonomously learn complex tasks, from basic object manipulation to dynamic locomotion, adapting to varied environments without explicit programming (M. A.-M. Khan et al., 2020). This capacity for on-the-spot adaptation and learning is further proved in the gaming industry, where RL agents have not only mastered classic board games (Ghory, 2004) like Chess and Go but have also conquered intricate video games (Lample et Chaplot, 2017), often outperforming human experts. Meanwhile, in the automotive sector, autonomous vehicles leverage RL for intricate decision-making processes, path planning, and obstacle detection, ensuring safe navigation in unpredictable road scenarios (Aradi, 2020). The successes in these areas underscore RL proficiency in tackling problems that demand real-time decision-making and adaptive learning (Sutton et Barto, 2018).

Its adaptive nature make it a suitable technique for uncertain environments, such as those found in wireless networks. Several innovative proposals already exploited RL in TDMA-based networks and they demonstrated a huge potential for increased network performance. In this section, we start by giving an overview of RL. Then, we focus on detailing the RL-based TDMA MAC scheduling techniques since they are highly related with our thesis. We categorize these techniques based on their engagement with communication channels : those that incorporate multi-channel communication within the algorithm and those limited to single-channel utilization. We proceed by providing a description in terms of objectives, how RL is used in the proposed method and the performance results for each method.

2.3.1 Reinforcement Learning

RL is defined in Sutton et Barto, 2018 as a machine learning technique based on learning from interactions with the environment. This learning process is goal-directed where an agent must discover which actions yield the most reward by trying them. The interaction between a learning agent and its environment is defined in terms of states, actions and rewards using the formal framework of Markov Decision Processes. An agent in state s takes an action a that takes it to another state s' and then receives reward r from its environment, as depicted in Figure 2.4. This learning process is repeated enough times for the algorithm to converge and results in an optimal policy π^*. This latter maps the states to the optimal actions to ensure that the design goal of the algorithm is achieved. RL methods can be classified in three categories according to (Heidrich-Meisner et al., 2007) :

— *Critic-only methods or learning based on value functions* : their idea is to first find the optimal value function and then derive an optimal policy.
— *Actor-only methods* : here the optimal policy is directly searched in the policy space.
— *Actor-critic methods* : which are a combination of the above approaches, the policy (actor) and the value functions (critic) are represented and improved separately. The critic measures the performance of the actor and decides when it should be improved.

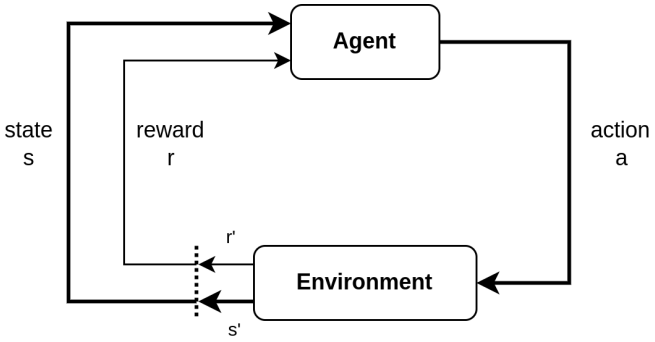


FIGURE 2.4 – Reinforcement learning principle

The most known critic-only method is Q-learning which is a model-free reinforcement learning algorithm, detailed in (WATKINS et DAYAN, 1992). A QL agent takes an action to maximize the value function, $Q(s, a)$, of the state-action pair. $Q(s, a)$ represents the expected total discounted returns from the action taken in state s :

$$Q(s, a) \leftarrow \alpha\ Q(s, a) + (1 - \alpha)(r + \gamma\ max_a\ Q(s', a))$$

where α is the learning rate, γ is the discounting factor that sets the preference either towards immediate rewards or to long-term rewards, r is the received reward from executing action a in state s that leads to state s'. Further, $max_a Q(s', a)$ represents the largest Q-value that can be obtained from the actions that can be taken in the next state.

RL can be combined with deep learning to deal with the problem of high-dimensional state and action spaces, resulting in Deep RL as stated in (ARULKUMARAN et al., 2017). It is generally based on training Deep Neural Networks (DNN) to approximate the optimal policy and/or the optimal value functions. Furthermore, DNNs could be used to model the environment in which RL agents will be trained.

2.3.2 Single-channel Utilization Algorithms

Z. LIU et ELHANANY, 2006 introduced RL-MAC as a reinforcement learning based MAC protocol for WSNs. It was designed to optimize the nodes' radio on-off function in order to minimize energy consumed by the sensor nodes. The main particularity of the proposed algorithm is that it adapts to the traffic generation pattern of the node and also of its neighboring nodes. The proposed learning scheme uses Q-learning with a reward function that tries to find a trade-off between minimizing energy consumption and ensuring an acceptable throughput. Precisely, the state s is represented by the number of packets queued for transmition at the beginning of the slotframe. The action is the reserved active time for the node. The reward function is a combination of two components, the first one reflects on the internal state of a node at timeslot t which aims to maximise the energy efficiency. The second component reflects on the state of other nodes as perceived at timeslot $t + 1$ which tries to minimize the number of missed packets. An ϵ-greedy method is followed in the learning algorithm to ensure a balance between exploitation and exploration. Performance evaluation of RL-MAC shows that it outperforms in terms of energy efficiency and data throughput S-MAC proposed in W. YE, HEIDEMANN et ESTRIN, 2004, and T-MAC proposed in DAM et LANGENDOEN, 2003, two MAC protocols designed to reduce energy waste caused by idle listening by managing the nodes' duty cycle. Also, latency is reduced considerably in RL-MAC compared to S-MAC especially when traffic load is heavy.

Finally, we can say that despite the fact that the proposed protocol is one of the first works including RL for scheduling nodes' radio on-off, it is outdated (2006) and does not fit the strict requirements of nowadays IIoT applications.

Always in the context of energy-saving directed protocols, Mihail Emilov Mihaylov et al., 2011 proposed a self-organizing reinforcement learning approach for scheduling the wake-up cycles of nodes in a wireless sensor network. It allows to adapt the use of sensor resources to the applications requirements in terms of latency, data rate and lifetime. Concretely, each node will learn to stay awake during the periods where it needs to communicate with its parents/children nodes (nodes that belong to the same *coalition*), this behaviour is called synchronization. At the same time, the node learns to stay asleep when neighboring nodes on the same hop are communicating (nodes in another coalition). In other words, the node desynchronizes with the neighboring nodes that are not in its coalition to avoid radio interferences and packet loss. The algorithm uses a value iteration approach similar to Q-learning with an implicit exploration strategy. The action space contains the timeslot numbers within the slotframe, an agent selects a slot when its radio will be switched on for the duration of the duty cycle which is fixed by the user. Each agent stores a "quality value" for each timeslot which is updated every time an event (overheard, sent or received packets, idle listening) occurs during that slot. The node will stay awake for those consecutive timeslots (of a length equal to the duty cycle) that have the highest sum of Q-values. Evaluating this protocol in different topologies has showed that it provides much lower end-to-end latency compared to S-MAC. However, various shortcomings can be addressed in the proposed protocol. For example, the duty cycle is fixed and equal for all nodes in the network which means that it is not traffic-adaptive. In addition, communications between active nodes on the same routing branch can collide since they are synchronized. These drawbacks has been solved in a subsequent extension of the algorithm called DESYDE, proposed in Mihail Mihaylov et al., 2011.

Savaglio et al., 2019 proposed a Q-learning MAC protocol (QL-MAC) which self-adjusts the node's duty-cycle to minimize energy consumption without impacting the other network parameters. It optimizes the sleeping and active periods of the nodes based on traffic predictions and transmission state of neighboring nodes. This is ensured by applying a Q-learning scheme where each slot is assigned a Q-value that is updated based on the actions of a node or the state of neighbor nodes. A node decides whether it should stay active or in sleep mode during each time slot, so the action space depends on the number of timeslots in a frame. The reward function is crafted carefully to take into consideration the state of neighboring nodes in addition to node state. Performance results show that QL-MAC reduces considerably energy expenditure with a minimal negative impact on the PDR compared to CSMA/CA. However, the authors did not provide insights on the impact of the learning algorithm on latency which allows to say that there is no guaranties of low latency.

The previous works are protocols developed from scratch and do not rely on a standardized stable protocol, such as TSCH. Based on TSCH MAC protocol, Nguyen-Duy et al., 2019 presented RL-TSCH, a reinforcement learning solution for scheduling TSCH nodes. The algorithm schedules the process of turn on/off nodes radio at each beginning of timeslot based on the current state and the previous state of the node. It takes into consideration the number of packets in the transmission buffer along with the remaining energy in every node. The learning agent determines the number of active and non-active timeslots at the beginning of each slotframe. Thus, the node behaves as in MSA (Minimal Scheduling Algorithm) [2] during the active times-

2. MSA turns on the node radio at the start of each time slot for a specified period of time to check for packets. If packets are present, it sends or receives them ; if not, it turns the radio off after the specified time.

lots and turns off its radio during the non-active timeslots. The algorithm applies Q-learning with an action space consisting in choosing the number of active timeslots. The reward function is designed to minimize energy consumption and ensure a high throughput and reliability. Evaluating the proposed algorithm in different small-scale topologies showed that it reduces the energy consumed to about one third compared to MSA, achieves a similar PDR but the latency is much higher to that obtained with MSA. Based on the obtained results, it is logical to say that the proposed algorithm does not fit for low-latency applications but may be well suited for applications with strict energy requirements.

Another work based on TSCH protocol is the one presented by H. Park, Haeyong Kim, S.-T. Kim et al., 2020. The authors introduced a multi-agent reinforcement learning based scheduler for TSCH, called QL-TSCH. The proposed algorithm reduces collisions while allowing contention, so multiple links can be scheduled in the same timeslot. This has the effect of increasing network throughput while allowing low energy consumption, thus the algorithm fits for high-density and high-traffic applications. In fact, each node acts as a Q-learning agent that learns the transmission slot with the lowest transmission failure rate and transmits only in that slot. During the learning phase, an agent performs a transmission with a probability $P_{exploration}$ in a slot chosen by an Action Peeking (AP) mechanism that selects the least active timeslot. Otherwise, the action (the timeslot) with the highest Q-value is selected. The chosen timeslot is scheduled as a transmission slot and the remaining slots are scheduled as listening slots. Rewards are assigned based on the transmission success or failure, a null value when a transmission succeeds and a negative value is assigned if it fails. This algorithm addresses the non-stationarity problem of the multi-agent system that may interrupt the convergence by including the aforementioned AP mechanism. This latter consists in a node observing the activity of other neighboring nodes during its listening slots, the node concludes that a timeslot is already reserved whenever a nearby communication is detected during that slot. Performance evaluation of QL-TSCH has been done in a large-scale network of 99 nodes by considering the PDR and the end-to-end packet delay in three industrial scenarios, compared with Orchestra and FTA scheduler proposed in H. Park, Haeyong Kim, K. T. Kim et al., 2019. Results show that QL-TSCH outperforms both Orchestra and FTA in terms of PDR, a better end-to-end packet delay compared to Orchestra but FTA has the best performance regarding this latter metric. In spite of all the good results that QL-TSCH achieved, its energy expenditure has not been evaluated. This leaves an open future research perspective.

2.3.3 Multi-channel Utilisation Algorithms

Phung, Lemmens et al., 2013 proposed a multichannel protocol for data gathering WSNs with a reinforcement learning based scheduling algorithm. The aim of the algorithm is to minimize energy consumption caused by collision, idle listening and deafness problem in WSNs. It addresses the joint problem of route selection and transmission scheduling and solves it in a fully distributed manner without a need for a coordination between the nodes. In other words, it makes nodes learn not only to which parent but also on which channel they should forward their data. Concretely, in each slot a node executes an action from the set of available actions (listen on its own channel for reception or transmit to one of its parents default channels) and keeps track of the probability of successfully performing each action in that slot. Such a probability is updated for the selected action in a given slot and will be the basis for choosing the best actions in the scheduling phase. The trade-off between exploitation and exploration is ensured by applying a "*win-stay lose-shift*" policy. In fact, a successful action will be repeated in the same slot in the next frame while a failed action leads to choose randomly another action from the action space in the next frame. The algorithm is traffic-adaptive because a node only contends for channel access when it has

packets in its queue. Evaluation results of the proposed protocol show that it outperforms a frequency-hopping protocol called McMAC, proposed in Hoi-Sheung et al., 2007, in terms of PDR, end-to-end latency and it provides 9 times better energy efficiency.

Phung, Huong et al., 2018 introduced a scheduler for TSCH networks supporting multiple QoS objectives. It is based on a trial-and-error process where each node acts as an autonomous agent learning from feedback from its environment and the interaction with the other nodes. In fact, two RPL instances are considered in the design of the learning scheme, one is reserved for delay-sensitive data and the other one for regular data. A node either listens for data coming from the children nodes, transmits data to the parent of delay-sensitive instance or transmits to the parent of regular instance. The reward function is a combination of rewards for the objective of reliability and energy and rewards for the objective of low latency. Based on the received reward signal, the action to execute in the next slotframe would be the same as in the current one if the reward equals 1 or another action is selected if the reward equals 0. The learning algorithm keeps track of the probabilities of successfully performing each action in each slot, the same way as the precedently described algorithm. The probabilities are updated for each executed action and the allocation process exploits the final obtained probabilities to choose the best actions for each node during the slotframe. The scheduler has been evaluated with two RPL instances where one is for delay-sensitive data (bounded latency setup at 2 timeslots) and the other is for regular data. Results show that the proposed scheduler provides much lower data delivery latency than Orchestra (one order of magnitude lower) while keeping a similar energy consumption. This proves that the scheduler fits for delay-sensitive applications requiring low latency.

In order to consider the more general problem of satisfying packets' deadlines in delay-sensitive environments, Chilukuri, Piao et al., 2021 proposed RLSchedule as a reinforcement learning based TDMA slot scheduler for networks with strict time constraints. It has the objective of finding a schedule where the least possible number of packets miss the deadlines by the least amount of time. The paper identifies a set of node features that will be the basis for network state representation. This enables the RL scheme to take scheduling decisions based on an up-to-date dynamic network status and not on the same static criterion every time. The proposed framework follows a centralized approach where a controller gathers information about the most frequently seen network scenarios and sends this knowledge to a server. Deep RL with PPO (Proximal Policy Optimization) is applied to learn the optimal policy which will be sent to the centralized controller. This latter exploits the received policy to build schedules for any scenario it sees. The action space consists in choosing the first M (the set of available channels) non-conflicting transmissions based on a proven baseline heuristic. The used heuristics are the following five : a) Deadline Monotonic (DM), b) Earliest Deadline First (EDF), c) Proportional Deadline (PD), d) Earliest Proportional Deadline (EPD) and e) Least Laxity First (LLF). A sixth possible action consists in choosing the top M non-conflicting transmissions with the minimum set of features. The reward function is designed in such a way to minimize the number of packets missing their deadlines and even if they exist, the time by which they miss their deadlines is minimized. Performance evaluation of RLSchedule shows that it provides much lower packet delay and it has the least percentage of missed packets compared to the other scheduling heuristics (DM, EDF, PD, EPD, LLF). The obtained results confirm that RLSchedule is well suited for time-constrained networks.

Chilukuri et Pesch, 2021 presented RECCE, a scheduler that follows the same principles of RLSchedule but considers a more general problem by including routing in the learning scheme. In fact, RLSchedule considers the best (shortest) routes following Dijsktra algorithm while RECCE explores and learns multiple routes and schedules to deliver more packets within the deadline.

Experimentation results showed that by exploring different routing paths and choosing the one (not necessarily the shortest) with the minimum delay allows RECCE to meet the scheduling goal better than RLSchedule.

2.3.4 Summary

Table 2.1 summarizes the discussed techniques in a comparative way, provides details on the experiments done for each and also their advantages and drawbacks.

We notice that some of the proposed RL-based schedulers aim to adjust the nodes duty cycling or the nodes wake-up cycles to reduce energy consumption (Z. Liu et Elhanany, 2006 ; Mihail Emilov Mihaylov et al., 2011 ; Savaglio et al., 2019 ; Nguyen-Duy et al., 2019). Notably, among these, only the RL-TSCH method utilizes TSCH. Nevertheless, the intrinsic design of TSCH is intended for precise scheduling where specific transmission/reception slots are allocated to fulfill application demands. This contrasts with the broader concept of self-adjusting duty cycle techniques. However, QL-TSCH (H. Park, Haeyong Kim, S.-T. Kim et al., 2020) is more elaborated, carefully designed for TSCH networks, and extensively tested as opposed to the previous approaches. It learns a transmission slot among all the available timeslots with the aim of reducing collisions while allowing contention. Nevertheless, its energy consumption has not been assessed. Intuitively, it should be high since the algorithm designates all slots, except for the transmission slot, as reception slots, constantly listening for communication between neighboring nodes. Moreover, the algorithm proposed in Phung, Huong et al., 2018 exhibits a similar oversight, as its energy consumption assessment appears to have been somewhat overlooked.

Deep RL-based methods such as RLSchedule (Chilukuri, Piao et al., 2021) and RECCE (Chilukuri et Pesch, 2021) design nodes features to model the network state. Examples of nodes features include the number of transmissions of a node at a specific timeslot, the minimum remaining time of a transmission queued at the node, the maximum number of remaining hops of a transmission at the node, among others. Moreover, the Deep RL scheduling algorithm is trained offline in a remote server based on a set of predefined network scenarios and results in a final policy that will be stored at a controller. Moreover, the proposed algorithms generated all of these features and network scenarios offline using probabilistic models.

The deployment of such models in an actual network necessitates the adoption of innovative networking architectures capable of real-time status monitoring and historical data retention throughout the network's life cycle. Additionally, the architecture must facilitate the transmission of scheduling commands to IoT nodes. These functionalities can be achieved through an NDT, which serves as a real-time digital mirror of the IIoT, maintaining an up-to-date model throughout its entire operational life cycle. This way, the deep RL scheduling algorithm can extract the necessary features from the NDT and can also take advantage of the historical data stored in the digital twin for the training phase.

TABLE 2.1 – General Comparison of RL-based Schedulers

Scheduler	Objective	Class	RL	Simulation	Testbed	Compared with	Advantages	Limitations
RL-MAC (Z. LIU et ELHANANY, 2006)	Energy saving	Single-channel	Q-Learning	NS-2 simulation	No	S-MAC and T-MAC (star with 5 nodes, linear with 10 nodes, mesh topologies with 100 nodes)	increased throughput and energy saving.	- synchronization problems - no real test-bed experiment - old proposal and compared with outdated protocols.
(De)synchronicity (Mihail Emilov MIHAYLOV et al., 2011)	Low latency	Single-channel	Q-Learning	OMNet++ simulation	No	S-MAC (Topology : Line with 4 nodes, Single-hop mesh with 6 nodes, Grid with 4x4 nodes)	- low end-to-end latency. - reduced energy consumption	- not well suited for irregular data traffic. - fixed duty cycle for all nodes - coordination problem.
QL-MAC (SAVAGLIO et al., 2019)	Energy saving	Single-channel	Q-Learning	Cooja simulation	TelosB motes with TinyOS	CSMA/CA (100% and 60% duty cycles), Topology : mesh with 7 nodes, grid with 4x4, 7x7, 10x10 nodes	- adaptable to topology changes . - reduced energy consumption	- no insurance of low latency - tested only against CSMA/CA
RL-TSCH (NGUYEN-DUY et al., 2019)	Energy saving	Single-channel	Q-Learning	Cooja simulation (Zoletia Z1 sensor nodes)	Nodes running Contiki OS	TSCH Minimal Scheduling Function (In simulation : 4-8 nodes in star, linear and mixed topologies. In real test-bed : 3 nodes)	- reduced energy consumption - traffic-adaptive	- no extensive experimentations - not suited for low latency applications. - tested only against the Minimal Scheduling Algorithm (MSA)
QL-TSCH (H. PARK, Haeyong KIM, S.-T. KIM et al., 2020)	High throughput	Single-channel	Q-Learning	No	ARM Cortex M4	Orchestra and FTA scheduler (Mesh topology with 99 nodes)	- fit for high-traffic, high-density large scale networks. - low latency and collision rate.	- no garanties of recuced energy consumption.
PHUNG, LEMMENS et al., 2013	Energy saving	Multi-channel	General RL	Matlab simulation	No	Mc-MAC (grid and random topologies of 25 nodes)	-traffic-adaptive - low energy consumption	- synchronization problems. - experimentations done in a matlab simulation.
PHUNG, HUONG et al., 2018	Low latency	Multi-channel	Trial-and-error process	Matlab simulation	No	Orchestra (grid topology of 16 nodes)	- traffic prioritization included - low data delivery latency - fit for delay-sensitive applications	- no garanties of reduced energy consumption. - experimentations done in a matlab simulation.
RL-Schedule (CHILUKURI, PIAO et al., 2021)	Satisfying deadlines requirements	Multi-channel	Deep RL - PPO	Home-made simulator	No	Scheduling heuristics (DM, EDF, PD, EPO, LLF,CFLLF, Random)	- Satisfies deadlines for time-constrained applications. - scheduling based on up-to-date network status - topology adaptive.	- no real test-bed experimentations - home-made custom simulator - need for a centralized controller
RECCE (CHILUKURI et PESCH, 2021)	Satisfying deadlines requirements	Multi-channel	Deep RL - PPO	Home-made simulator	No	Scheduling heuristics (DM, EDF, PD, EPO, LLF,CFLLF, Random)	- routing included in the learning scheme → increased performance relatively to CHILUKURI, PIAO et al., 2021	similar to CHILUKURI, PIAO et al., 2021

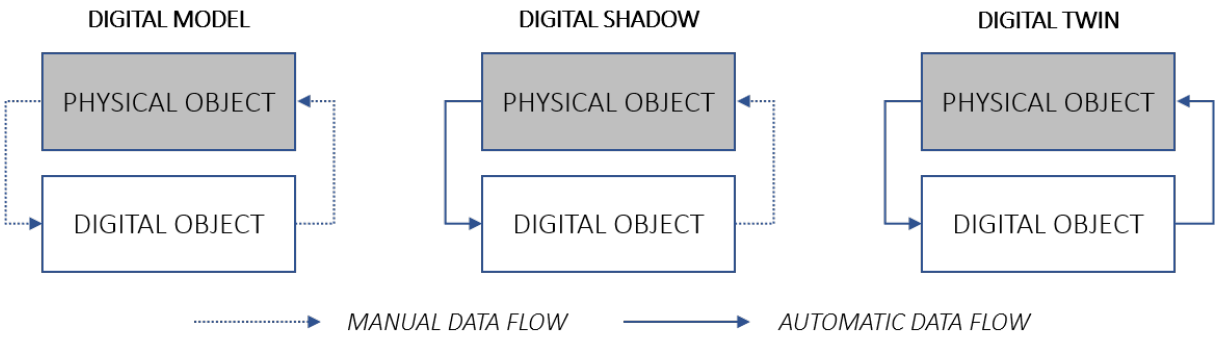


FIGURE 2.5 – Differences between Digital Model, Digital Shadow and Digital Twin.

The content of this section has been published as a survey on RL-based TDMA MAC schedulers in the IIoT at the 6th IFAC Symposium on Telematics Applications (KHERBACHE, SOBIROV et al., 2022).

2.4 Digital Twins

A *Digital Twin* can be viewed as a machine that is emulating or "twinning" the life of a physical entity (BARRICELLI, CASIRAGHI et FOGLI, 2019). A DT is more than just a simple simulation or a static model, it is a continuously evolving model that is always aware of the events happening in its physical twin as it follows its lifecycle to supervise and optimize its functions. The synchronization between the DT and its physical counterpart is possible thanks to the real-time data uploading ensured by IoT devices and sensor technology, while big data storage capabilities allow keeping historical data that can be useful for the Digital Twin. Artificial Intelligence (AI) algorithms, for instance, can be used to predict future states of the physical twin. A DT can also simulate new configurations in order to apply preventive maintenance operations (BARRICELLI, CASIRAGHI et FOGLI, 2019). Theses properties result in reducing costs and resources in many industries such as manufacturing and healthcare systems.

DT as a concept includes a real space, a virtual space and information/data exchange between the two spaces. It is important to highlight that in the absence of the two-way connection between the virtual and the real space, it is called Digital Model. Moreover, in the absence of automatic data flow from the virtual space to the physical space, we talk about Digital Shadow. Figure 2.5 illustrates the difference between the three concepts.

DT was first introduced in 2002 by Professor Grieves in the Product Lifecycle Management (PLM) course of the University of Michigan (WU et al., 2020). The concept went through a slow-going development phase from 2003 to 2011 where only few articles were published (TAO et al., 2019). Right after 2011, the availability of low-cost sensors and communication technologies in the era of the IoT, the emergence of big data analytics and simulations technologies, the remarkable advance in Machine Learning (ML) along with powerfull computation infrastructure, triggered the rise of DT technology. It gained widespread popularity among researchers in both academia and industry when NASA gave a formalized definition of DT and how it would ameliorate performance in the astronautics and aerospace field (GLAESSGEN et STARGEL, p. d.). In 2014, Grieves published the first white paper extending the DT concept from one conceptual idea to various practical applications (GRIEVES, 2014). Since then, numerous DT applications have appeared in various domains, in manufacturing (CUNHA et al., 2021 ; MANDOLLA et al., 2019), health-

care (Bruynseels, Santoni de Sio et Hoven, 2018), maritime and shipping (Arrichiello et Gualeni, 2020 ; Authority, 2018), city management (Fan et al., 2021), among others. In a 2023 report (Gartner, 2023), Gartner predicted that the digital twin market will cross the chasm in 2026 to reach 183 billion dollars in revenue by 2031.

2.5 Network Digital Twins (NDT)

The networking industry has shown great interest in DTs due to their immense potential in improving the field (Almasan et al., 2022). In this regard, specifically, the focus has been on NDTs. An NDT is defined as a virtual representation of the physical network that has the purpose of analyzing, diagnosing, emulating, and then controlling the physical network based on data, models and interfaces (Zhou et al., 2023). This differs from traditional network simulators with the integration of a two-way connection between the physical network and its digital replica. The physical network sends telemetry to the NDT to offer insights into its current state, thus, ensuring that a real-time network mirror is continuously maintained. On the other hand, the NDT sends back control commands to regulate the behavior of the physical network.

In this Section we first survey existing NDT architectures as found in the literature. We then transition into a discussion of network modeling techniques that are especially relevant for the development and application of NDTs.

2.5.1 NDT architectures

Several architectures ranging from generic to targeted for specific network types have been proposed in the literature. The IETF group is currently engaged in standardizing the NDT concept. In their latest draft (Zhou et al., 2023) they outline the basic concepts and building blocks of an NDT. They argue that NDT brings several advantages to network management including reduced network operation cost, optimized decision making through predictive maintenance, and increased safety when testing new network capabilities. A reference architecture for the NDT is proposed in this draft aiming to serve future developments in such a context. As illustrated in Figure 2.6, it is composed of three layers : Application layer, Digital Twin layer, and Real Network layer. The latter represents the physical network which forwards data regarding its status to the digital twin layer, while in return, receives control messages from it aiming to regulate its operation. The Digital Twin layer includes :

- *Data Repository subsystem* : it collects and stores diverse network data to construct real-time operational models of network elements via the twin southbound interface. It also offers data services like quick retrieval and batch service, along with unified interfaces for the Service Mapping Models subsystem.
- *Service Mapping Models* : handle data modeling, enhancing network agility and service programmability. It features two types of data models : basic and functional. Basic models provide real-time, accurate representations of the network based on elements like configuration, operational state, and topology. Functional models cater to specific needs like network analysis, fault diagnosis, and quality assurance, and can be tailored for various network types and life-cycle stages.
- *Digital Twin Network Management* : ensures the management function of the NDT, it tracks life-cycle activities of the twin entities, oversees their performance and resource use, and also manages individual models. The system provides visualization and control over various aspects such as network topology, model governance, and security measures.

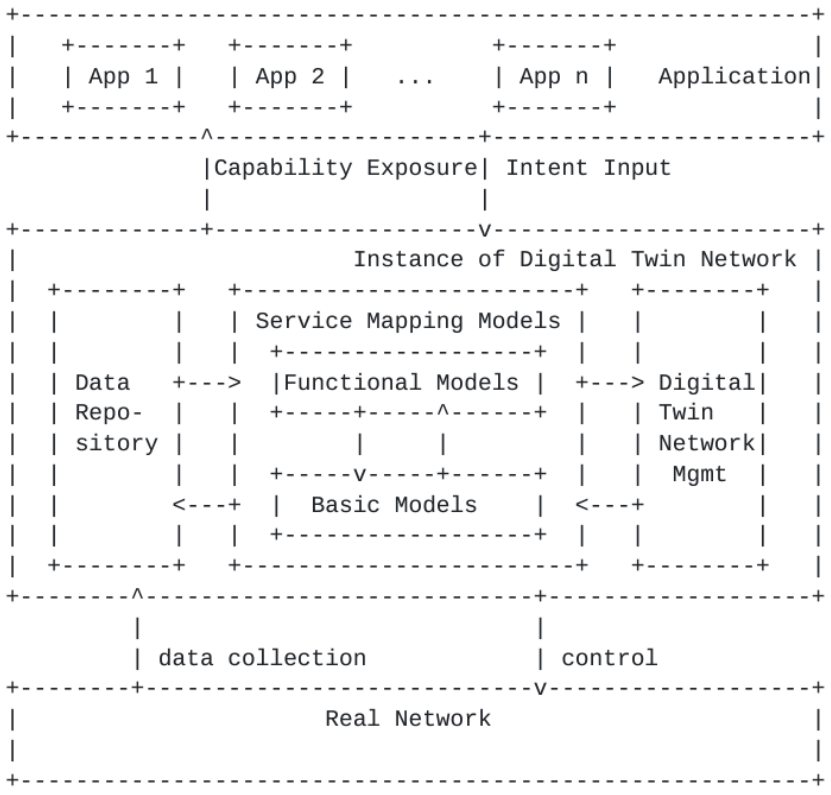


FIGURE 2.6 – Reference Architecture of Network Digital Twin , by ZHOU et al., 2023

The Application layer represents various network applications making requests that need to be satisfied by the NDT.

ALMASAN et al., 2022 provide a general architecture of the NDT, illustrated in Figure 2.7. The central component of the architecture is the Digital Twin, which serves as a virtual replica of the physical network. Using network state parameters like traffic patterns, topology, and scheduling policies as inputs, the DT generates various network metrics such as utilization rates, delays, or anomalies. Because the NDT is an isolated, accurate representation of the actual network, operators can safely test various scenarios without risking service interruptions (what-if scenarios). The NDT outputs are versatile and can be tailored to specific applications, ranging from time-series data to global network metrics. Examples of network applications can be "troubleshooting", where network operators can use the NDT to recreate previous network conditions in order to identify the underlying cause of service disruptions. Moreover, anomaly detection can be enabled by the NDT which alerts of the anomaly and identifies a set of potential root causes for it.

Figure 2.8 depicts the four-layer architecture for the NDT proposed in ZHU et al., 2021. It consists of a Physical network layer, Data lake layer, DT layer and Network application layer. The data lake layer collects, stores and pre-processes the collected data to provide knowledge and relation extraction to the DT layer for NDT model construction. The DT layer consists of : i) physical entity modeling that completes the modeling of the different networking components, ii) requirements modeling that is used to develop a multitude of scenario models such as network resource prediction, anomaly detection, automatic operation through AI algorithms, iii) Twin management & control center which orchestrates the DT layer to satisfy the intended

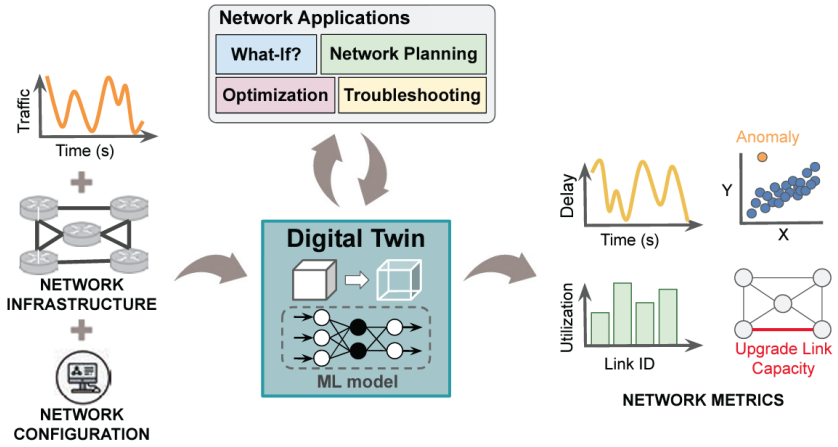


FIGURE 2.7 – General Network Digital Twin architecture, by ALMASAN et al., 2022

requirements.

In the context of Vehicular Networks, ZHAO et al., 2020 propose an Intelligent Digital Twin (IDT) for Software-Defined Vehicular Networks (SDVN) to enable intelligent and adaptive routing in vehicular networks. By having a DT that replicates the vehicular network in real-time, a report is sent to it when a routing error occurs. The IDT verifies different routing schemes before choosing the one with the best performance to deploy in the physical network. Figure 2.9 illustrates the IDT-SDVN architecture which is composed of : i) the real-world/physical SDVN, ii) the SDVN controller : is deployed in the MEC (Mobile Edge Computing) server and its role is to calculate the routing requests and scheduling vehicles, ii) the IDT : is deployed in the MEC and represents the virtual replica of the vehicular network, it models and verifies the instant learned functional models such as verifying improved routing schemes before applying them to the real network.

An Application-driven Digital Twin Networking (ADTN) middleware is proposed in BELLAVISTA et al., 2021 to facilitate the interaction with heterogeneous distributed industrial devices and the dynamic management of network resources by adopting an application-level approach. Figure 2.10 depicts a structured, multi-layered representation of a distributed IIoT setup, featuring various decentralized edge locations. These locations are overseen by single or multiple interconnected nodes that are tasked with managing local applications and DTs. The architecture is organized into three main layers. At the foundation is the Physical Devices layer, featuring a variety of sensors, controllers, and actuators. Above this is the Edge layer, which is split into DTs and an Edge Network Manager. DTs serve as virtual duplicates of the physical devices, while the Edge Network Manager executes local intelligence for fast decision-making, security, and compliance with access policies. It also enhances QoS through dynamic network control. The topmost Control Room layer, which is closely integrated with the Edge layer, provides a global view of the system. It also dynamically optimizes communication between DTs based on specific needs and QoS requirements.

The proposed ADTN middleware is composed primarily of the Simple Digital Twin (SDT) on the edge and the Composed Digital Twin (CDT) on the control room. The SDT is a software agent at the edge layer that mirrors a physical IoT device by digitalizing its features and functions. It maintains synchronized communication with the device, facilitating standardized interoperability across devices, services, and applications. Additionally, the SDT can augment the device's original capabilities, including supported protocols and packet processing. Alongside SDTs on the

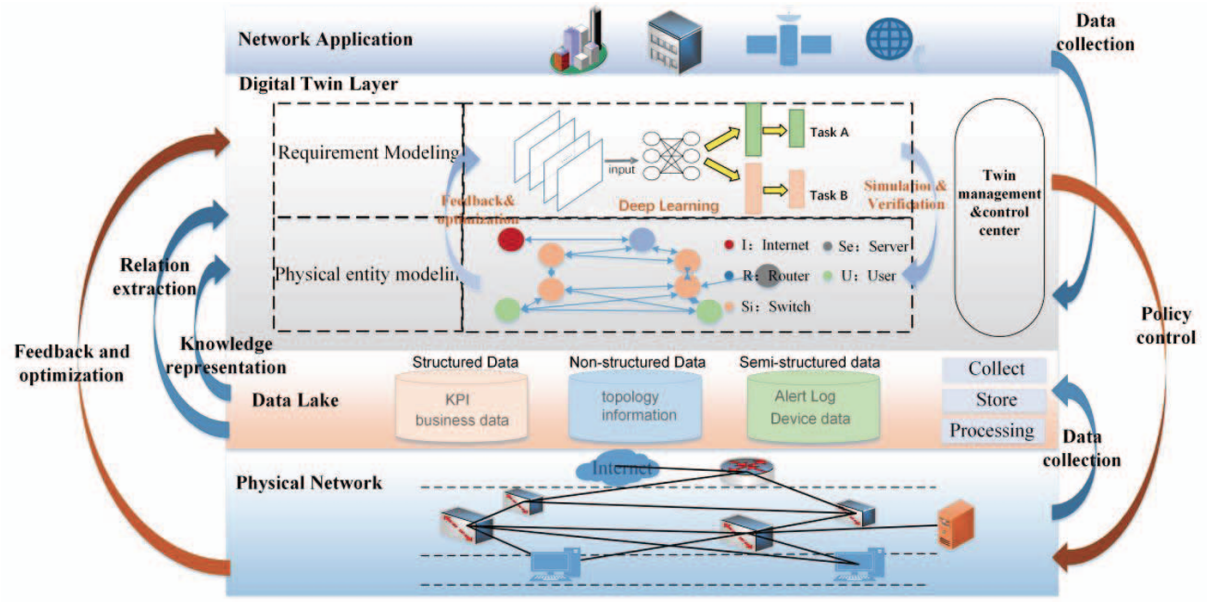


FIGURE 2.8 – The architecture of Network Digital Twin, by ZHU et al., 2021

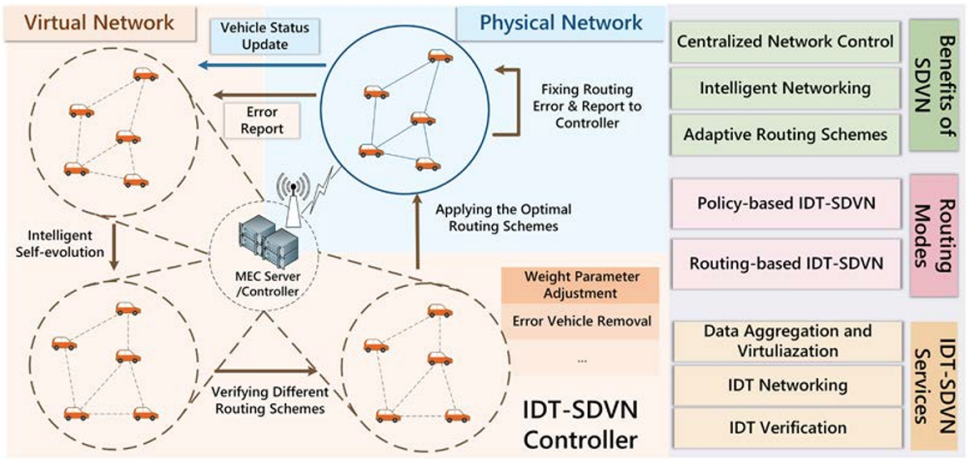


FIGURE 2.9 – Intelligent Digital Twin-Based Software-Defined Vehicular Networks architecture (ZHAO et al., 2020)

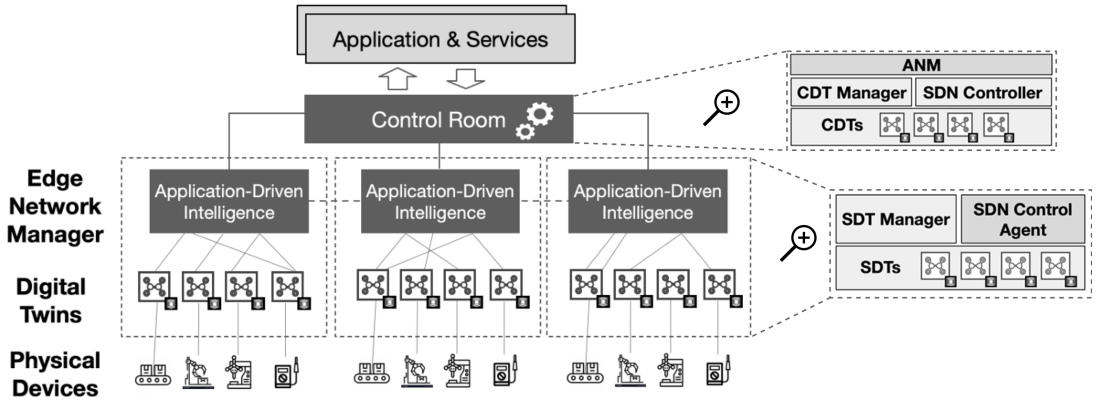


FIGURE 2.10 – High-level scheme of a distributed, edge-based and intelligent IoT architecture with a zoom into the ADTN middleware components, reproduced from (BELLAVISTA et al., 2021)

edge layer, the SDT-Manager (SDT-M) is responsible for creating, configuring, and managing the life cycle of SDTs. It advertises SDTs and receives configuration commands through remote interactions with the CDT Manager. Next to the SDT-M, the SDN Control Agent (S-CA) sends edge node characteristics and status to the Control Room. It also receives traffic engineering rules aimed at optimizing packet management. In the control room, we find the CDT which is a digital entity capable of aggregating multiple SDTs. This allows for the efficient modeling of complex, distributed applications and behaviors. The CDT Manager (CDT-M) is responsible for orchestrating the creation and management of CDTs. It communicates with S-CAs to gather information on physical devices and adjusts CDT configurations based on detected context variations. Alongside this, the SDN Controller (S-Ctrl) collects network data from S-CAs to generate network topology and distributes traffic engineering rules based on CDTs QoS requirements. It interacts with both active and intermediary edge nodes. Finally, on the top level of the control room, the Application-driven Network Manager (ANM) is regarded as the entry point to the entire system, ANM allows technicians to manage CDTs and network topology. It interfaces with both CDT-M for CDT status and QoS, and S-Ctrl for network resource tuning and traffic management to achieve the desired QoS requirements.

2.5.2 Network Modeling Techniques for the NDT

Due to being in the early stages of development, NDTs face various challenges that need to be addressed including scalability, interoperability, real-time requirements, data modeling difficulties, security risks, etc (ZHOU et al., 2023). Several works have started addressing the aforementionned challenges. We discuss here only those related to the modeling process of a network. Mainly, because simulation tools are delicately designed and tightly coupled, leading to high execution time, thus, increasing cost and computational power. The authors in ALMASAN et al., 2022 have experimentally demonstrated that deep learning models (Graph Neural Networks, Recurrent Neural Networks) offer very low execution cost compared to the traditional packet-level network simulator OMNet++. These approaches and other network modeling techniques are currently investigated in the research community to enable the full potential of NDT. Particularly to minimize the computational requirements and lower execution costs, thus, leading to faster network operations. This will allow to forecast performance and prevent network failure in advance (HUI et al., 2022).

In H. Hong et al., 2021, NetGraph is proposed as an NDT platform that enables intelligent network maintenance in DCNs (Data Center Networks). It exploits configuration and state data to represent the network using three models (device, network, and service) and provides functional blocks such as inventory search and models translation. Ontologies and knowledge graphs are used in Zhu et al., 2021 ; Ma et al., 2022 to model the network and the relationship between its components in the digital space. Real-time data from network elements is continuously fed into the knowledge graphs to maintain a current view of the network. In Ferriol-Galmés et al., 2022, Graph Neural Networks (GNNs) are used to model the network for accurately estimating delay, jitter and loss using real network data including various traffic loads, topologies, routing configurations and queue scheduling policies. The results show that GNNs effectively model the relational aspect in networks and can generalize well to scenarios unseen during training. H. Wang et al., 2022 uses the same modeling technique to enable efficient and autonomous management of network slicing in 5G networks. The proposed GNN-based model allows capturing the intricate interactions between multiple slices, investigating composite traffic generated by slices and generating accurate end-to-end slice latency predictions.

2.5.3 Summary

Based on the range of NDT architectures examined, it's evident that the approaches span a broad spectrum. While some propose generic architectures as outlined in (Zhou et al., 2023 ; Almasan et al., 2022 ; Zhu et al., 2021), others target more specialized use-cases, such as Software-Defined Vehicular Networks (Zhao et al., 2020) and industrial edge environments (Bellavista et al., 2021). It is important to maintain the research efforts and discussions on NDT architectures. Focusing on generic architectures aids in the standardization of the concept, while specialized architectures cater to the unique requirements and constraints of specific types of networks.

IIoT networks are known to be complex and hard-to-manage systems since they control constrained industrial applications where various requirements need to be satisfied. Reliability, strict real-time constraints and energy requirements are the most crucial ones among others. While NDT-related research is gaining a progressive interest recently, there is still a gap in providing a detailed architecture of a network digital twin for the IIoT given its constrained nature and complex characteristics that should be taken into consideration.

Moreover, it is clear that network modeling should be extensively investigated for the NDT by exploring all potential approaches. Although there has been substantial research employing data-driven techniques such as GNNs and ontologies/knowledge graphs to model network behavior, these approaches often lack the formal rigor and the capacity for real-time fault detection that are essential in mission-critical applications. In contrast, Petri-nets stand out as a formally rigorous approach, proven to be exceptionally effective in capturing the behavior of discrete systems, especially in the networking field (Billington, Gallasch et B. Han, 2004a).

2.6 Conclusion

In this chapter we have reviewed the current literature and identified the research gaps of our thesis. First of all, we performed a comprehensive overview on 6TiSCH protocol stack from the physical to the application layer while focusing particularly on TSCH protocol.

Furthermore, we have covered a selection of centralized and decentralized TSCH scheduling algorithms, which we consider to be the most impactful in existing literature. We extended our

focus particularly to RL-based algorithms, broadening our scope to include not just TSCH-based, but also other TDMA-based MAC scheduling approaches. A comparative study on these approaches is provided to analyze them in terms of objectives, how RL is used in the proposed schedulers, the performance evaluation setup of each, the advantages and drawbacks of each, etc. Moreover, the need of proposing an NDT for the IIoT is made evident especially to adopt and integrate the Deep RL-based TSCH schedulers in a real IIoT network.

DTs are introduced after that since they make the foundations of our next chapters. NDTs are defined and related work on NDT architectures is provided, before discussing state-of-the-art network modeling techniques for the NDT.

In the next chapters, we will present the contributions of our thesis, starting with an assessment of RL-based and traditional decentralized TSCH schedulers under heterogeneous traffic conditions.

Chapter 3

RL-based and Decentralized TSCH Scheduling : The Case of Heterogeneous Traffic

RL-based scheduling algorithms demonstrate great potential due to their continuous observations within the network environment. This feature enables their adaptability in uncertain environments, such as those found in wireless networks. The exploration of reinforcement learning methods to enhance the performance of TDMA-based networks has resulted in several innovative proposals. We reviewed them in the previous chapter while emphasizing that QL-TSCH (H. Park, Haeyong Kim, S.-T. Kim et al., 2020) is the most elaborated RL-based TSCH scheduler in the literature that can be deployed in a decentralized manner. However, its energy consumption has not been assessed and it should be intuitively high because of the allocation of all the slots for reception except for the transmission slot.

In the first part of this chapter, we introduce QL-TSCH-plus as an enhancement to QL-TSCH scheduler to reduce its energy consumption. Our approach introduces a distributed scheme for the Action Peeking mechanism while maintaining QL-TSCH original reward function. Instead of constantly monitoring its neighbor nodes' communications, a node now broadcasts the learned transmission slot to its neighboring nodes, allowing the receivers to update their Action Peeking Table for the transmitted slot. Moreover, this broadcast message is also exploited for Rx slots allocation while ensuring their alignment with the RPL routing tree.

In the second part of this chapter, we consider evaluating a selection of representatives among decentralized schedulers, mainly, autonomous, distributed and RL-based. The evaluation aims at creating their energy consumption and network performance profiles to get some insight into their relevance to applications that have to handle simultaneously highly heterogeneous traffic. We define a scenario that includes two heterogeneous traffic flows qualified as *heavy* or *light* depending on their respective data rates and carry an extensive performance evaluation based on a unified environment under the same network conditions and metrics. Based on the experiment results, a set of conclusions is drawn for each scheduler in an attempt to help choose a scheduler depending on the application requirements. This is motivated by two important observations. First, a mix of different applications can generate heterogeneous traffic conditions in an industrial context. This includes the concurrent functioning of applications for periodic monitoring and those for process control, leading to a varied traffic scenario. Some nodes in such scenarios may be handling a high volume of traffic, while others only deal with low-rate traffic, a situation particularly prominent in Industry 4.0 settings. For example, some sensors may be sending out low data

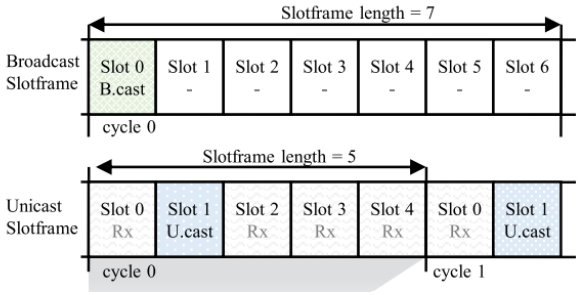


FIGURE 3.1 – Slotframes design of QL-TSCH (H. PARK, Haeyong KIM, S.-T. KIM et al., 2020)

rate traffic to monitor various environmental parameters on a shop-floor, such as temperature, humidity, or gas levels. In contrast, nodes embedded in cameras or scanners for automated quality inspection systems may be dealing with a substantial amount of data, necessitating high-rate data transmission. Another instance is sensors transmitting high-resolution visual data to formulate a 3D digital twin of an industrial component (HE, GUO et ZHENG, 2018), functioning simultaneously with sensors providing diagnostic data about the component's performance and condition. Second, the evaluation of decentralized schedulers has not been investigated in the literature under highly heterogeneous traffic conditions. Most of the related work considered low data rate traffic and not highly heterogeneous traffic. In addition, the integration of RL-based schedulers in the evaluation under highly heterogeneous traffic conditions is unprecedented. RIGHETTI et al., 2020 provide a comprehensive analysis of 6TiSCH distributed mode and they carry out a performance evaluation of distributed schedulers. Although extensive performance evaluation is done, autonomous schedulers were not considered and traffic conditions were not heterogeneous. Two autonomous schedulers were assessed in ELSTS et al., 2020 without targeting heterogeneous traffic. This is also the case of the work in ORFANIDIS et al., 2021 that additionally considered one distributed scheduler to evaluate with heterogeneous mobility patterns.

3.1 From Autonomous to Distributed QL-TSCH : introducing QL-TSCH-plus

3.1.1 QL-TSCH and its Limitations

As presented in Section 2.3 of the previous chapter, QL-TSCH (H. PARK, Haeyong KIM, S.-T. KIM et al., 2020) is a Q-learning based TSCH scheduling algorithm allowing contention but striving to minimize collisions. It operates on two slotframes : broadcast and unicast, as shown in Figure 3.1. The broadcast slotframe is used for broadcast traffic (TSCH EB, RPL DIO, etc). It has 7 timeslots in length with only the first slot being allocated for shared transmission (Tx) and reception (Rx), while the node turns off its radio during the remaining slots. In contrast, the unicast slotframe, which is used for data communication, can have different sizes (9, 15, 25). Here, only one slot is considered a Tx cell, and all other slots act as Rx cells.

The network is represented as a multi-agent system where each node serves as a QL agent learning from its environment and the neighboring nodes to determine the appropriate transmission slot. To handle the non-stationarity problem that might disrupt the algorithm's convergence in the multi-agent system, QL-TSCH incorporates an Action Peeking mechanism. In this scheme, a node listens to the medium during Rx slots and logs this data into the Action Peeking Table, which has the same size as the unicast slotframe. Basically, the APT records the usage rate of

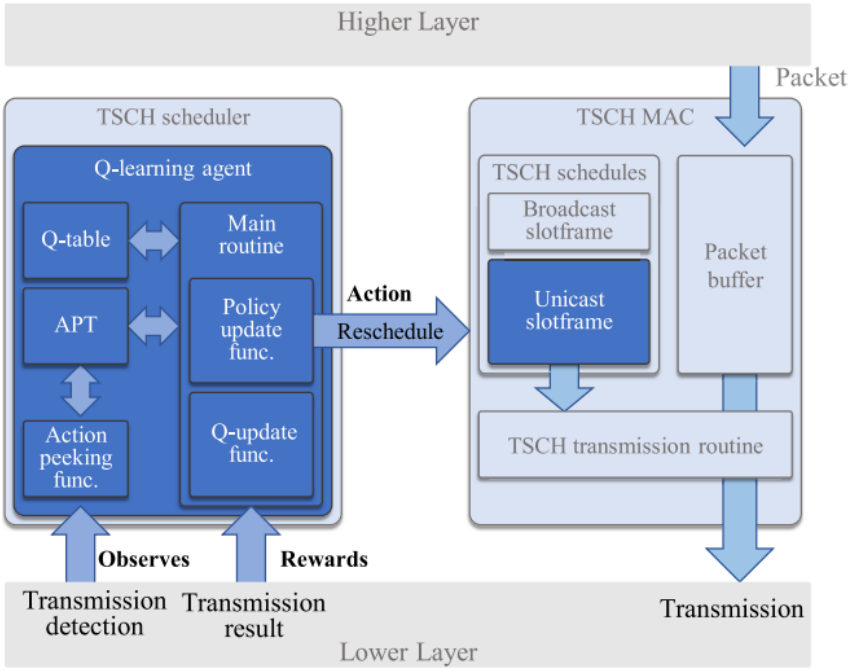


FIGURE 3.2 – Structure diagram of QL-TSCH (H. PARK, Haeyong KIM, S.-T. KIM et al., 2020)

each slot. As a result, the protocol can select the least recently utilized slot (i.e., the one with the minimum value) from the APT as a Tx slot.

The protocol awards a node based on the success of its transmission on the Tx slot, providing a reward of 0 for a successful transmission and -1 for a failed one. This $0/-1$ reward design was found to be the most efficient after testing different setups by the QL-TSCH authors. Following the receipt of the reward, the node updates the respective Q-value in the Q-table, which matches the size of the unicast slotframe. Figure 3.2 summarizes the global operation of QL-TSCH algorithm.

After each unicast slotframe cycle, both the Q-table and the APT are updated, and a new Tx slot is scheduled. QL-TSCH adopts an epsilon-greedy exploration/exploitation strategy. During exploration, a node selects the Tx slot with the minimum APT value, indicating that it is the least recently utilized slot. This mechanism minimizes collisions in QL-TSCH while mitigating the non-stationarity problem. Conversely, during exploitation, the node selects the Tx slot based on the Q-table values, specifically the timeslot with the highest Q-value, suggesting that the node has earned the most rewards from this timeslot. QL-TSCH incorporates a policy update function that encourages nodes to prioritize exploration and then gradually shift to exploitation as time passes.

3.1.2 QL-TSCH-plus

A major drawback of QL-TSCH is that it requires a node to constantly listen to communications between its neighboring nodes. This is an inherent part of the action peeking mechanism used to address the non-stationarity issue in the network. As a result, the nodes never enter a sleep state, leading to a high energy consumption within the network. To mitigate this high energy consumption issue, we introduce an enhancement to the AP mechanism. Recognizing that the primary contributor to increased energy consumption is the active listening feature, our pro-

posed modification involves disabling this feature. This is achieved by eliminating the Rx slots that are allocated across all the slotframe except for the Tx slot. Instead, these slots are allocated according to the RPL DODAG (A Routing Protocol for Low-Power and Lossy Networks (RPL) Destination-Oriented Directed Acyclic Graph (DODAG)). Algorithm 1 describes our proposed AP mechanism enhancement.

Algorithm 1 Distributed AP in QL-TSCH-plus

```
1: Input : Node node, Action Peeking Table APT, Transmission slot Tx_slot, RPL DODAG
   RPL_DODAG
2: procedure BroadcastTxSlot(node, Tx_slot)
3:     Broadcast Tx_slot to neighboring nodes
4: end procedure
5: procedure OnReceive(packet)
6:     APT[packet.Tx_slot] += 1
7:     if IsChildNode(packet.sender, RPL_DODAG) then
8:         Rx_slot = packet.Tx_slot
9:         Allocate Rx_slot in unicast slotframe (for communication with packet.sender)
10:    end if
11: end procedure
12: function IsChildNode(sender, RPL_DODAG)
13:    if sender is a child node in our RPL_DODAG subtree then
14:        return true
15:    else
16:        return false
17:    end if
18: end function
```

In order to maintain the correct functioning of the AP mechanism, a node will not actively sense communication between its neighbors. Instead, it will broadcast the learned unicast Tx slot (e.g., denoted t_i). A neighboring node upon receiving such a packet, increments the APT value of t_i. Furthermore, the receiving node verifies if the sender node is a child node in relation to its RPL DODAG subtree. If that is the case, it allocates an Rx timeslot that corresponds to the position of the received Tx slot (t_i). Figure 3.3 illustrates a scenario where Node X learns its transmission (Tx) slot and allocates it at timeslot 1. After learning the Tx slot, Node X broadcasts this information, which is received by its two neighboring nodes, Y and Z. Node Z updates its APT at position 1 but does not reserve a reception (Rx) slot at that position, as X is not a child node in Z routing tree. Conversely, Node Y not only updates its APT but also reserves an Rx slot at position 1, because X is one of its child nodes in the routing tree. This guarantees that the Rx slots are in sync with the learned Tx slots in the neighboring nodes to ensure that each node can send data to its RPL parent or receive data from its children nodes.

With the need for nodes to exchange a message containing the learned Tx slot for coordination, the algorithm evolves into a distributed one and loses its autonomous nature. This additional data flow, which is of a broadcast type, will be dispatched within the broadcast slotframe, thus, we allocate a second broadcast timeslot for it. This latter is allocated to reduce contention on the already allocated broadcast timeslot where TSCH EBs are sent. On another note, RPL signaling messages can take the form of either unicast or broadcast. If they are unicast, they will be sent in the unicast slotframe, causing them to run simultaneously with the unicast data packets. To manage this, we integrate an additional slotframe equipped with a common shared Tx/Rx slot

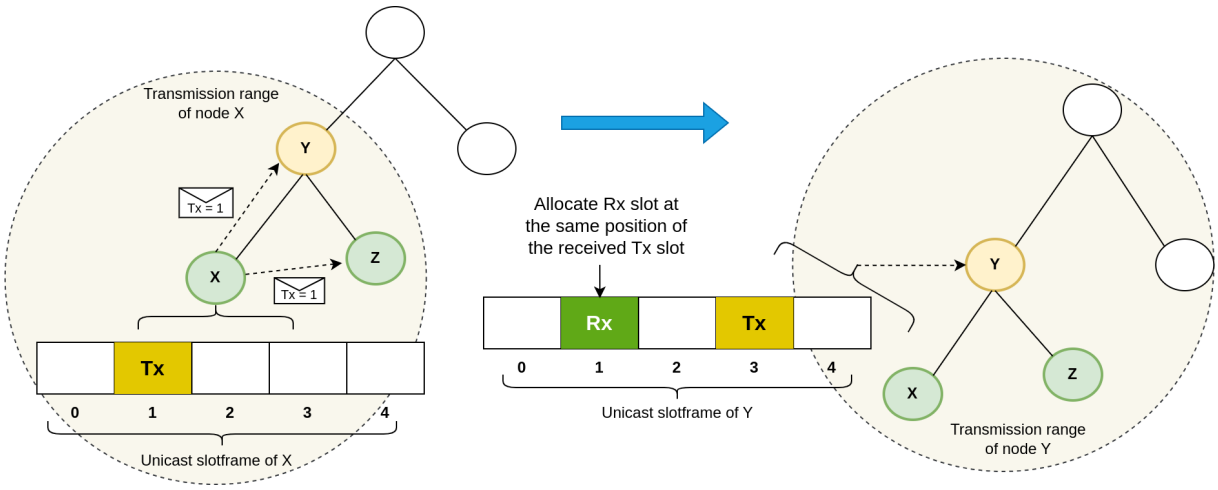


FIGURE 3.3 – Rx slots allocation

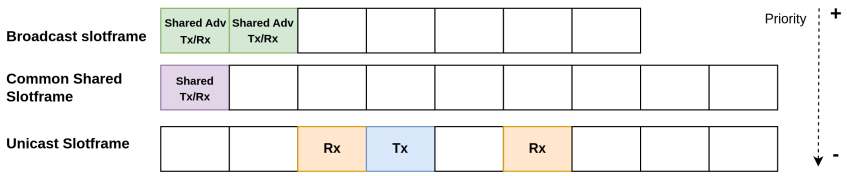


FIGURE 3.4 – QL-TSCH-plus slotframes design

specifically to accommodate RPL unicast packets. This results in three slotframes in total as shown in Figure 3.4, one for broadcast traffic (TSCH EBs, RPL DIOs, Packets containing the learned Tx slot, etc), one for RPL unicast signaling traffic and one for unicast data traffic, each running on a different channel offset.

3.2 Decentralized Scheduling Algorithms and Heterogeneous Traffic

In this Section, we aim to assess the performance of decentralized TSCH schedulers under highly heterogeneous traffic. As depicted in Figure 3.5, we mainly classify the schedulers into three categories : autonomous, distributed and RL-based. We selected representative protocols from each category to evaluate in a highly heterogeneous traffic scenario.

In the category of autonomous schedulers, we consider two of the discussed schedulers in the previous chapter, namely, Orchestra (DUQUENNOY et al., 2015) and ALICE (S. KIM, H.-S. KIM et C. KIM, 2019). This is motivated by the fact that while these schedulers do not add network overhead, they proved to achieve very good performance in various scenarios (ELSTS et al., 2020). In the RL-based category, QL-TSCH and QL-TSCH-plus that we propos in Section 3.1 are selected. This is for the joint goal of evaluating QL-TSCH-plus against QL-TSCH and also assess both of them under highly heterogeneous traffic conditions. Additionally, from the distributed category, we only evaluate MSF (Minimal Scheduling Function) given its high significance and ongoing standardization efforts (currently at RFC9033). The considered schedulers along with the adopted naming conventions with respect to the different settings are listed in Table 3.1.

We conduct a set of experiments under a unified environment that includes a notably hetero-

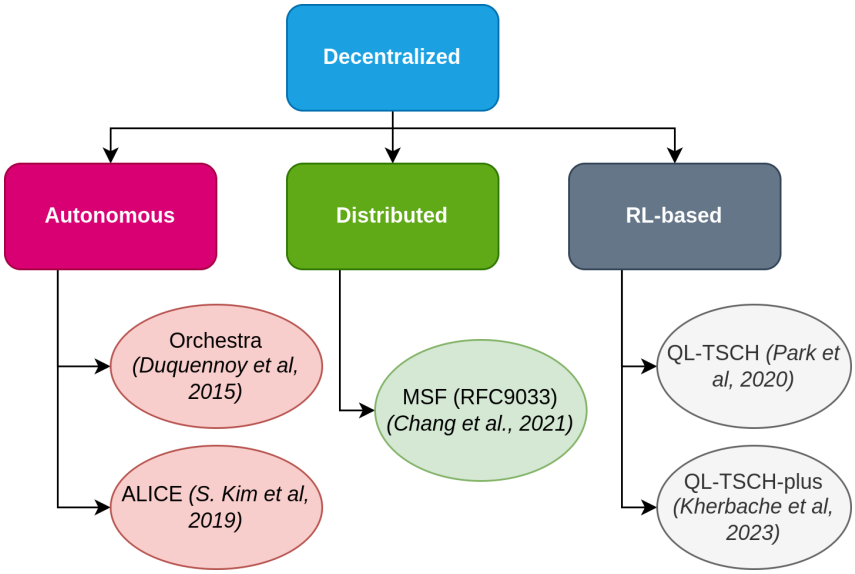


FIGURE 3.5 – Decentralized TSCH schedulers classification and chosen schedulers for evaluation.

TABLE 3.1 – Our naming conventions for the Evaluated Protocols

Reference name	**Scheduler**
Orchestra	Orchestra with link-based rule, root rule and storing mode routing enabled. The default configuration is considered in our evaluation (EBSF Period = 397, Common Shared Period = 31, Unicast Period = 17)
ALICE_mc_sf7	ALICE with node-based channel allocation (enhancement proposed in ELSTS et al., 2020) with a unicast slotframe size of 7
ALICE_sf7	ALICE original version with link-based channel allocation (S. KIM, H.-S. KIM et C. KIM, 2019) with a unicast slotframe size of 7
ALICE_mc_sf19	ALICE with node-based channel allocation with a unicast slotframe size of 19
ALICE_sf19	ALICE original version with link-based channel allocation with a unicast slotframe size of 19
MSF	6TiSCH Minimal Scheduling Function with a unicast slotframe size of 53 (version 08 impelementation)
QL-TSCH-5	QL-TSCH with : unicast slotframe size of 5, broadcast slotframe size of 7
QL-TSCH-plus-5	QL-TSCH-plus with : unicast slotframe size = 5, broadcast slotframe size = 7, RPL slotframe size = 9

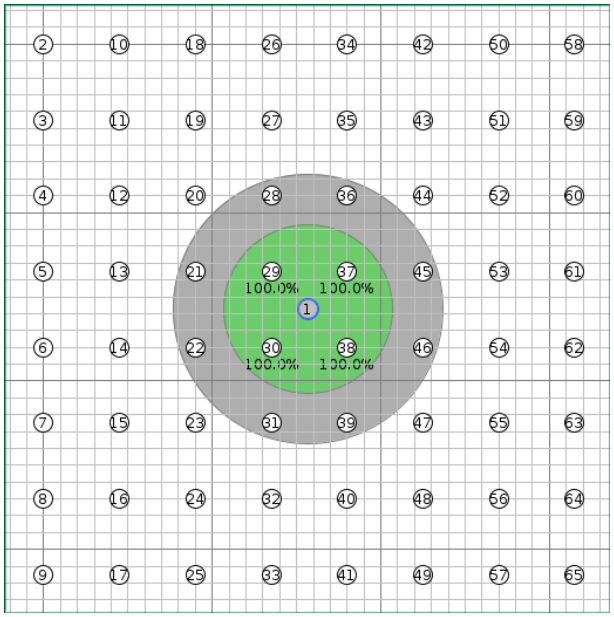


FIGURE 3.6 – Network topology

geneous traffic scenario. Consistency is ensured across all tests via a common network model and evaluation metrics. Our experiments have been conducted using COOJA simulator (OSTERLIND et al., 2006) provided by Contiki-NG (OIKONOMOU et al., 2022), an operating system for the IoT that implements a low-power IPv6 networking stack. Our experimental network topology, depicted in Figure 3.6, consists of a central Sink (node 1) encircled by 64 sensor nodes that form an 8x8 grid arrangement. The inter-node distance within the grid is maintained at 45 meters. All sensor nodes are equipped with an IEEE 802.15.4 radio, and adhere to the well-established Unit Disk Graph Medium (UDGM) as the propagation model. Cooja motes, being generic sensor nodes, are selected as the nodes of choice for these experiments. In terms of energy consumption quantification, we utilize an energy model based on Sky Motes (*Tmote Sky : Datasheet* 2006). Additionally, Contiki-NG Energest module is exploited, which is set to periodically calculate energy use every 10 seconds. In our scenario, sensor nodes are responsible for forwarding data from their respective areas of interest to the Sink, also referred to as the RPL root node. This data relay follows a 'convergecast' or many-to-one communication pattern, a widely used pattern in many IIoT applications. Four sensor nodes (nodes 2, 9, 58, and 65), strategically located at the grid's corners, carry the additional duty of producing and transmitting data at a high frequency. This results in what we refer to as 'heavy traffic' within our experiments. Each of these high-traffic flows involves the delivery of two packets per second (2 pps), each bearing a payload of 50 bytes, leading to an overall transmission rate of 800 bps (bits per second). The remaining 60 nodes transmit a single packet per minute, carrying a 10 byte payload. This results in an individual transmission rate of 1.33 bps, hence we label these as 'light traffic' flows. The Sink, or RPL root node, stands as a purely receptive entity in this setup, and hence does not generate any data packets. This setting with multiple flows, some of which exhibiting high data rates, will likely create bottlenecks close to the sink due to the converge-cast scheme. The results will show how the chosen TSCH schedulers manage the situation. Different settings of the chosen schedulers have been considered, resulting in a total of 8 schedulers configurations depicted in Table 3.1. We run each protocol configuration 10 times with a different random seed for each run

Table 3.2 – Simulation Parameters

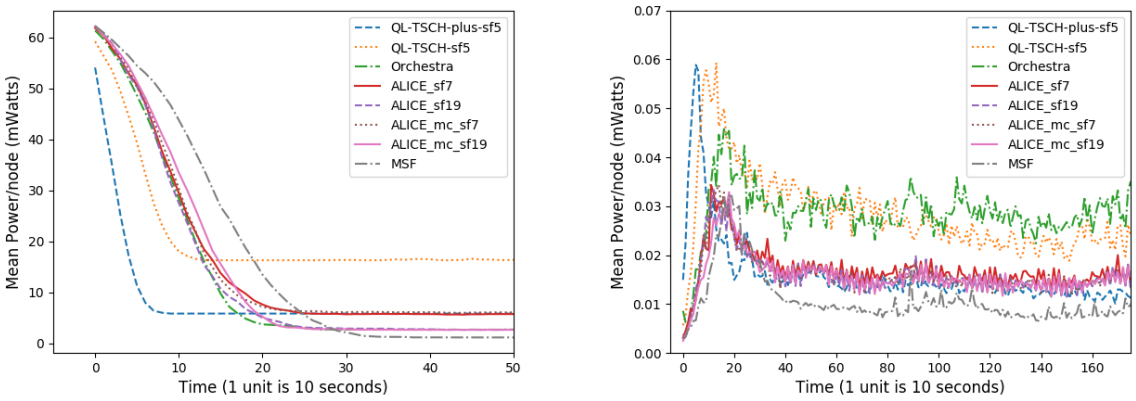
Parameter	**Value**
Warm-up duration	30 minutes
Data generation duration	30 minutes
Mote type	COOJA mote
Protocol stack	6TiSCH
Transport Protocol	UDP
Routing Protocol	RPL Classic
Number of channels	4
Network topology	Grid 8×8
Number of nodes	65
Distance between nodes	45 meters
Propagation model	UDGM
Interference range	80 meters
Tx range	50 meters
Heavy traffic	4 nodes, each transmitting at 800 *bps*
Light traffic	60 nodes, each transmitting at 1.33 *bps*
Timeslot duration	10ms
Maximum re-transmissions	3
Queue buffer size	8
Power Radio Rx	65.4 mw
Power Radio Tx	58.5 mw
Power CPU	7.2 mw
Power Low Power Mode (LPM)	3.6 mw

to increase the confidence interval of our simulations, mounting up to $8 \times 10 = 80$ simulations in total. The main simulation parameters are listed in Table 3.2.

3.2.1 Network Convergence Time

We report hereafter on the obtained results when the nodes do not exchange any application-level data packets (warm-up period) to analyze the network convergence with the chosen schedulers. This is done based on the study of the evolution of power expenditure over time which indirectly gives insight into how long it takes for the network to converge. The convergence time is the time required for the network to be in a steady state where all the nodes have joined the network and all configurations (MAC and routing level) have been successfully established.

Figure 3.7a depicts the mean power consumption per node over time. This gives insights into how the network consumes power with each scheduler to converge as a whole. We observe that QL-TSCH-plus clearly builds the network faster than the other schedulers, resulting in lower power consumption over a shorter period of time (around 70 seconds). It consumes as much power as ALICE_sf7 and ALICE_mc_sf7 when reaching the steady state. QL-TSCH converges to a steady state for approximately the same period as QL-TSCH-plus. However, after convergence, it showcases the highest power consumption among the evaluated schedulers (around 20 *mWatts*). This is mostly due to its action peeking mechanism which obliges a node to continuously listen to



(a) Overall power consumption

(b) Transmission power consumption

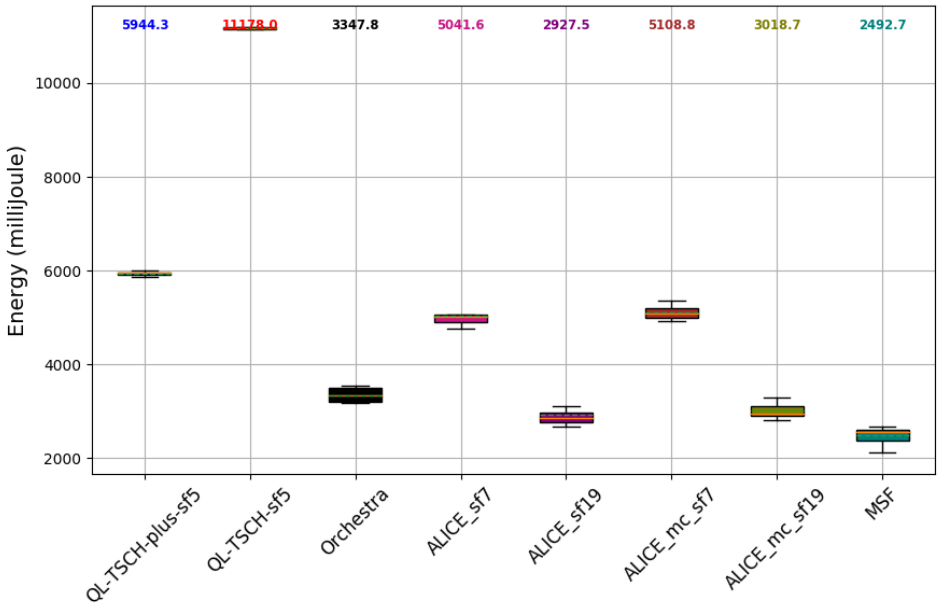
FIGURE 3.7 – Mean power consumption per node over time

communications between its neighbors. Additionally, the fast convergence of these two schedulers is also due to the short unicast slotframe size (5) considered in this evaluation. Orchestra and ALICE with the various considered configurations converge in the interval of $[150, 220]$. We also notice that Orchestra, ALICE_sf19 and ALICE_mc_sf19 consume less power than QL-TSCH and QL-TSCH-plus after convergence. MSF is the longest scheduler to converge because nodes take time to negotiate cells allocation. However, it provides the lowest power consumption after convergence which confirms the benefit of its traffic adaptive nature enabled by the cells negotiation process.

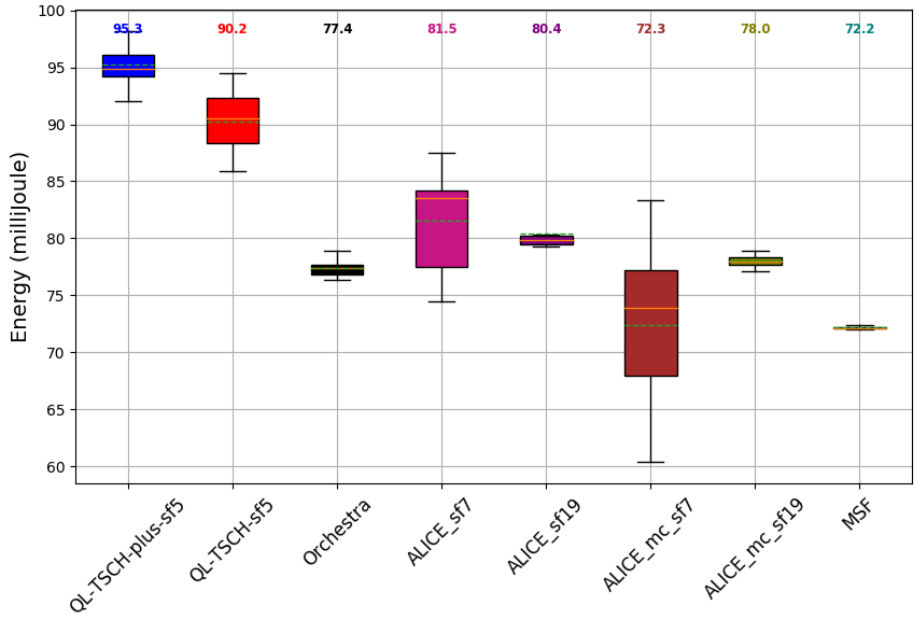
Based on our experiments, The overall power averaged on a 10-second period is mainly driven by the radio reception mode which accounts for about 95% of the total consumption. As a result, the latter follows the same pattern as the overall power. Figure 3.7b illustrates the mean transmission power usage per node which, as can be seen, is much lower than the overall and reception mode consumption. This can provide more insight into the actual exchanges that take place during the convergence phase. Initially, in the $[0, 250]$-second interval, we observe a high transmission power consumption for all the schedulers. This is due to the networking convergence or bootstrapping process. In order for the network to converge, various control packets are exchanged between nodes. For instance, at the TSCH level, upon receiving an EB message, considered as a request to join the TSCH network, a node joins this latter and broadcasts the message to its unknown neighbors to invite them to join the network in turn. Moreover, other control messages are exchanged at the routing level to build the RPL DODAG. Once the network has converged, we observe that Orchestra along with QL-TSCH-sf5 consume slightly more power than the other schedulers. MSF presents the lowest transmission power consumption after convergence because it has finished negotiating cells allocation and reached a stable state.

3.2.2 Energy Consumption

In this Section, we are interested in the total amount of energy consumed by the network during the whole simulation time. The obtained results are presented using box plots to illustrate their distribution. The lower and upper whiskers represent the minimum and maximum values, respectively. The box represents values from the first to the third quartile including the median value. The average values are highlighted above each box.



(a) Total energy



(b) Transmission energy

FIGURE 3.8 – Network energy consumption.

Figure 3.8 depicts the energy consumption of the schedulers. The boxplots are plotted using average values from the whole simulation time ; 30 minutes warm-up plus 30 minutes with heterogeneous traffic generation. Since the pattern of energy consumption for reception has the same tendency of the total energy, the figure solely illustrates the total and transmission energy expenditure for clarity and brevity. Remarkably, QL-TSCH-sf5 presents the highest total energy consumption as expected, because of the AP mechanism that obliges the nodes to keep listening for communications between their neighbors. QL-TSCH-plus-sf5 reduces energy consumption by up to 47% compared to QL-TSCH-sf5 because of our AP mechanism enhancement and Rx slots allocation based on the RPL DODAG. Now, a QL-TSCH-plus node broadcast its learned Tx slot to his neighbors which, depending on their position in the routing tree, decide to allocate or not an Rx slot for the node broadcasting its Tx slot. More specifically, a receiver node allocates an Rx slot at the received Tx slot if it finds that the transmitter node is one of its child nodes in the RPL tree. If it is not a child node, it only updates its APT without any allocation. The transformation of QL-TSCH-plus into a distributed scheduler with an additional control flow (to broadcast the learned Tx slots) as compared to its predecessor, QL-TSCH, results in QL-TSCH-plus having the highest transmission energy expenditure among all the schedulers, as highlighted in Figure 3.8b. Nevertheless, this increase in transmission energy is overshadowed by the energy savings achieved through the more efficient allocation of reception cells in QL-TSCH-plus.

It is true that QL-TSCH-plus energy consumption is 47% lower than QL-TSCH but it does not attain the levels of Orchestra or ALICE. Increasing the unicast slotframe size for QL-TSCH-plus will certainly reduce energy consumption but also increase the learning search space for the algorithm. This latter increase may affect the scheduler network performance and augment the need for additional time for the algorithm to converge.

Concerning the other schedulers, the energy consumption is highly correlated with the unicast slotframe size. For instance, Orchestra consumes slightly more energy than ALICE_sf19 and ALICE_mc_sf19 which run with a unicast period of 19 slots because it operates with a unicast period of 17 slots. Moreover, ALICE energy total consumption is not affected by whether or not node-based channel allocation is enabled. In contrast, the transmission energy consumption of ALICE with node-based channel allocation is slightly lower than without it. This confirms the utility of that enhancement which reduces collisions by using a more efficient channel allocation, thus, lowering the number of re-transmissions for ALICE. QL-TSCH-plus-sf5 consumes more energy than ALICE_sf7 and ALICE_mc_sf7. However, MSF presents the lowest energy consumption among all the schedulers because it operates with a unicast slotframe size of 53. Additionally, its traffic adaptability contributes to reducing energy consumption.

Apart from QL-TSCH-plus-sf5, which has the highest energy consumption for transmission as discussed earlier and shown in Figure 3.8b, QL-TSCH-sf5 ranks second in terms of energy use for transmitting messages. This is primarily because it utilizes the smallest unicast slotframe size. The transmission energy consumption of ALICE_sf19 and ALICE_mc_sf19 is approximately the same, with a slight reduction for the latter. A high variability is recorded for ALICE_sf7 and ALICE_mc_sf7, but the latter is noticeably more efficient. This confirms that the use of node-based channel allocation significantly cuts down on collisions and the need to re-send packets, making the scheduler more energy efficient. Among all the schedulers, MSF stands out for being the most energy-efficient in terms of transmission, thanks to its ability to adapt to traffic conditions.

3.2.3 Network Performance

This Section sheds light on the comparative performance evaluation of the selected schedulers. We consider the following network performance metrics :

Packet Delivery Ratio (PDR). The ratio of the number of packets received by the sink to the number of packets sent by the source nodes.

End-to-end Delay. The time that elapses between the transmission of a packet by a source node and its reception by the sink.

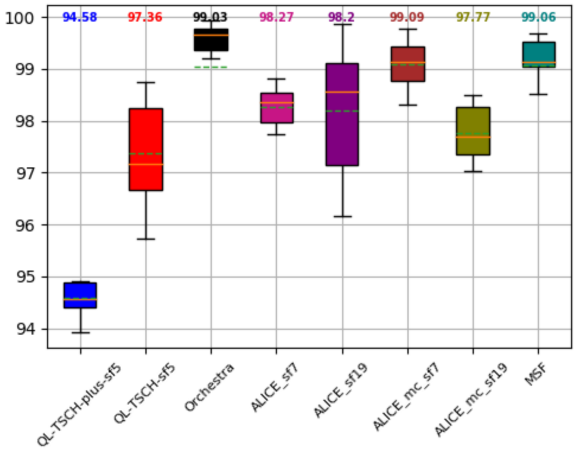
Network Throughput. The amount of data successfully received, in a time unit, by the sink transmitted by the different source nodes, expressed in *bps*.

Figure 3.9 depicts the end-to-end delay and PDR achieved by each scheduler overall and separately for heavy and light traffic flows described in Table 3.2. We should mention that depending on the application requirements, one can assign weights to each metric and perform a multi-criteria analysis in order to choose the best scheduler that satisfies the defined application requirements. We conduct such a study in Section 3.3. However, we consider that metrics have equal weights in the analysis we conduct in this section. We observe that Orchestra, ALICE_mc_sf7 and MSF present the highest PDR (99%) when considering the whole traffic (light and heavy traffic flows), but they differ in terms of end-to-end delay. MSF records the highest delay with a mean value of 1.55s. It is followed by Orchestra with approximately half that value (mean of 0.76s) and then ALICE_mc_sf7 presents an even lower mean delay value of 0.45s.

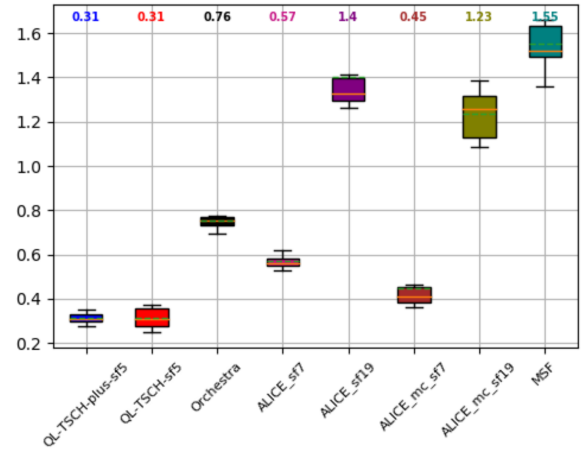
It is also noticeable that QL-TSCH-plus-sf5 has the lowest delay but with reduced PDR (average of 94%). On the other hand, QL-TSCH-sf5 presents the same delay performance while keeping a better PDR (97%) but its high energy consumption does not qualify it for use in a real industrial application. This makes QL-TSCH-plus a potential better fit scheduler for time-constrained industrial applications as it provides approximately the same network performance as QL-TSCH with a considerably lower energy usage (47% energy saving).

When considering the light traffic, ALICE_mc_sf7 presents the best performance when considering both metrics (PDR and end-to-end delay). However, it handles poorly the heavy traffic compared to the other schedulers (mean PDR of 90%). The same applies to ALICE_sf7 with a slight increase in delay. Operating ALICE with a unicast slotframe size of 19 enhances the PDR for the heavy traffic but with the disadvantage of increased delay values. Orchestra performs best in terms of PDR stability for both light and heavy traffic (average of 99%). However, it experiences higher delays when dealing with heavy traffic. This increased delay is mainly because of the need for packet re-transmissions in heavy traffic conditions. MSF is also stable in terms of PDR for handling both traffic flows, but a big increase in delay values is recorded for the heavy traffic (mean of 1.63s) compared to the light traffic (mean of 0.96s). The best stability when considering both flows and both metrics is assigned to QL-TSCH-sf5 and QL-TSCH-plus-sf5.

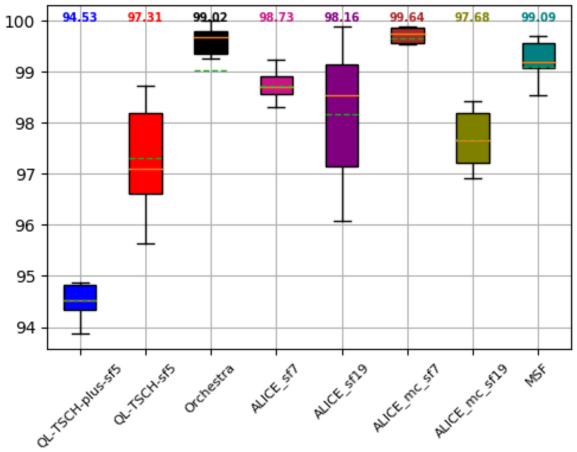
The average throughput of each scheduler is depicted in Figure 3.10. It is obvious that Orchestra shows the best performance, which evidences that Orchestra is an appropriate scheduler for applications requiring very high reliability and acceptable delay. Figure 3.11 depicts the achieved throughput by the different schedulers over time for the light and heavy traffic. The blue lines show the expected throughput for each traffic flow. The expected throughput is calculated theoretically to show the achievable data reception rate. For heavy traffic : 4 nodes send 800 *bps*,



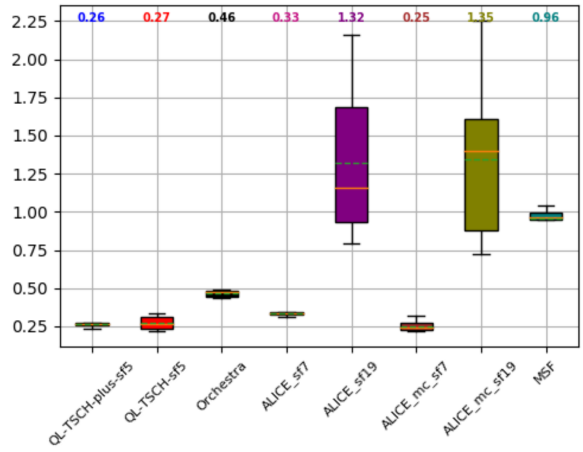
(a) PDR for the whole traffic



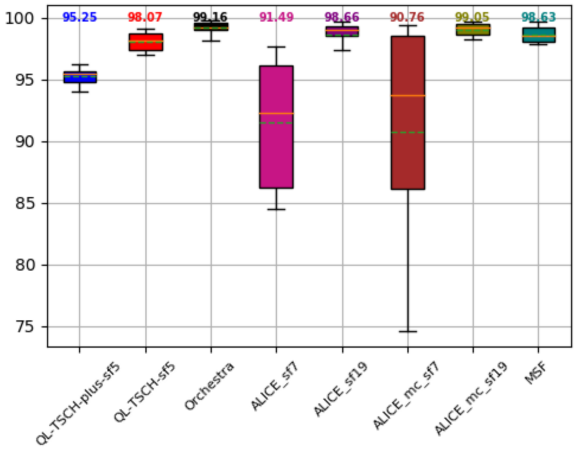
(b) Delay for the whole traffic



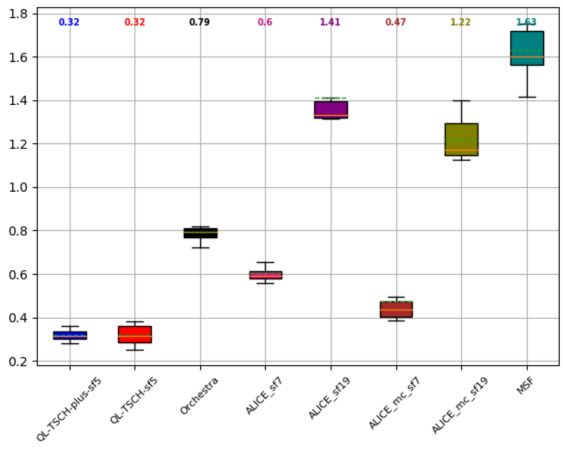
(c) PDR for the light traffic



(d) Delay for the light traffic



(e) PDR for the heavy traffic



(f) Delay for the heavy traffic

FIGURE 3.9 – PDR (%) and average delay (s) results.

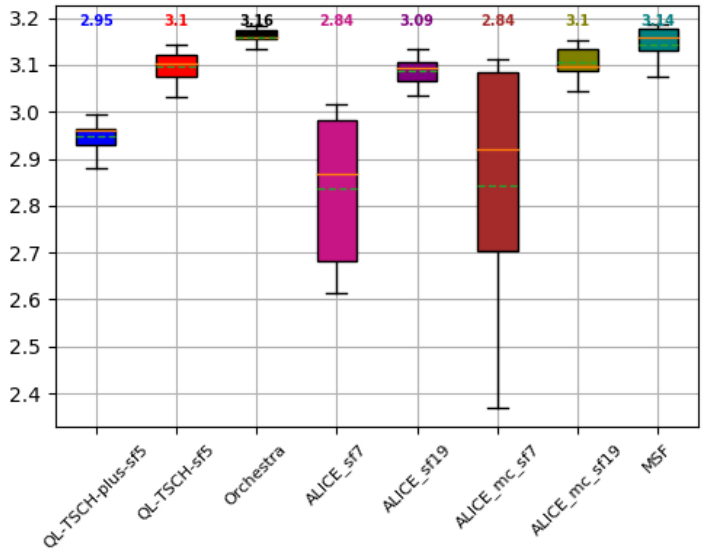


FIGURE 3.10 – Average network throughput in Kbps.

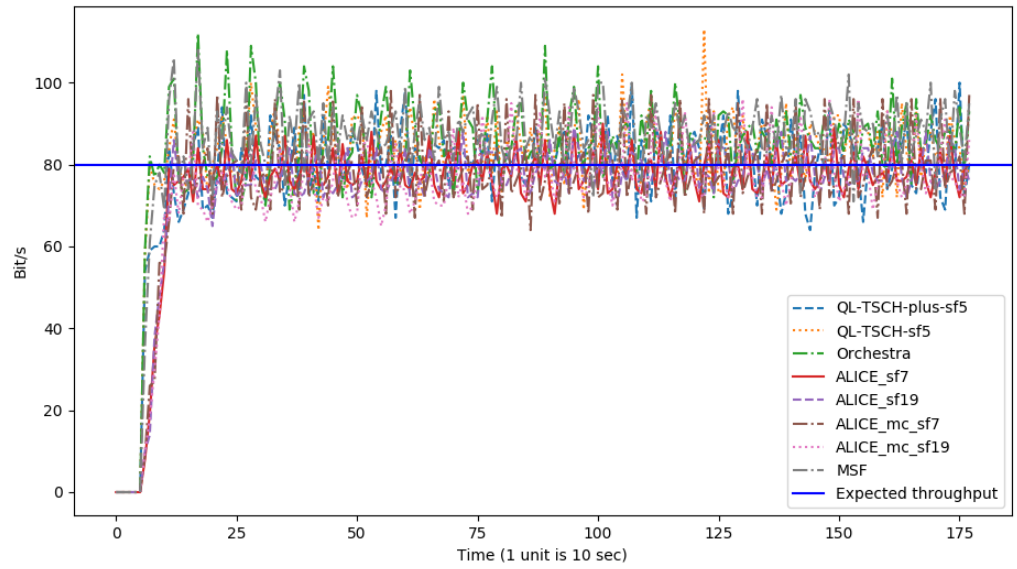
which results in $4 * 800 = 3200$ *bps* for the whole network. As for the light traffic : 60 nodes send $10 * 8/60$ *bps* each, resulting in 80 *bps* for the whole network.

Figure 3.11a showcases throughput levels for the light traffic for all the evaluated schedulers. There is no significant difference between the schedulers, showing that they all performed similarly in terms of throughput for light traffic evolution over time. Note that some levels are above the expected throughput because packets may be received with a variable jitter. Packets may be queued in the buffer and delivered in bursts, not to mention the effect of re-transmissions that can instantaneously increase the transmission rate.

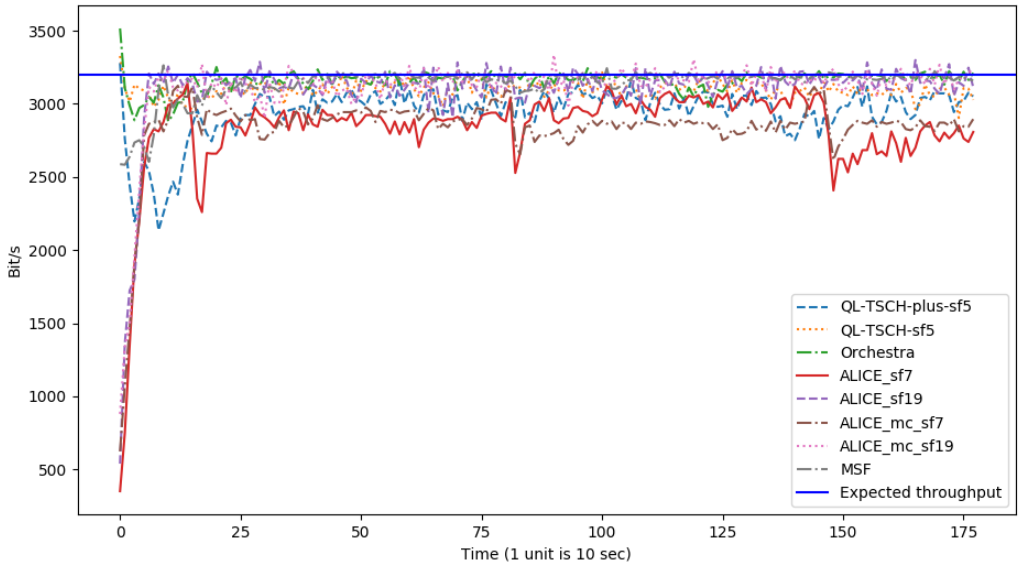
Figure 3.11b depicts the throughput for the heavy traffic for each scheduler over time. All the schedulers approach the expected throughput line except for ALICE_sf7 and ALICE_mc_sf7 (red and brown lines, respectively) which exhibit lower levels of throughput. This gives us insights to when packet loss exactly happened, for instance, ALICE_sf7 after 1500s presents high packet drops beyond the lowest point of the throughput. QL-TSCH-plus-sf5 heavy traffic throughput decreases at the beginning then increases and reach a more stable state, i.e., reaching convergence (values around 3000bps). This is because of the time it takes to learn the nodes best transmission timeslot and allocate reception slots accordingly.

3.3 Results Summary

The application requirements need to be considered in order to choose the best-fit scheduler, as previously stated. One can adopt a multi-criteria analysis of the schedulers to help in deciding which scheduler is suited for a specific application. Hereafter, we give an illustrative example where we consider a simple method in which a linear utility function is defined for each criterion, each of which corresponds to one evaluation metric. The utility function permits normalizing the metrics values in the range 0..1 where the value 1 is associated with the best value of the criterion. For instance, in the utility function defined for energy consumption, the minimum value should be the value associated with this criterion. On the other hand, a maximum value is to be considered when it comes to reliability (estimated via the PDR metric). Table 3.3 summarizes the



(a) Light traffic



(b) Heavy traffic

FIGURE 3.11 – Network throughput over time.

Criterion	QL-TSCH-plus-sf5	QL-TSCH-sf5	Orchestra	ALICE_sf7	ALICE_sf19	ALICE_mc_sf7	ALICE_mc_sf19	MSF
Energy (milliJoules)	5944.3	11178	3347.8	5041.6	2927.5	5108.8	3018.7	2492.7
Delay (seconds)	0.31	0.31	0.76	0.57	1.4	0.45	1.23	1.55
PDR (%)	94.58	97.36	99.03	98.27	98.2	99.09	97.77	99.06
Throughput (Kbps)	2.95	3.1	3.16	2.84	3.09	2.84	3.1	3.14
Convergence Time (s)	70	100	200	250	250	250	230	310

TABLE 3.3 – Multi-criteria average values of the different schedulers

Criterion	**Surveillance Application**		**Industrial application**	
	Score	*Weight (%)*	*Score*	*Weight (%)*
Energy	5	0.33	3	0.19
Delay	1	0.07	5	0.31
PDR	2	0.13	2	0.12
Throughput	4	0.27	1	0.06
Convergence Time	3	0.2	5	0.31

TABLE 3.4 – 5-Level Scale Criterion Preferences

average values of the evaluated schedulers for the five criteria (metrics) : Energy consumption, Delay, PDR, Throughput and Convergence Time. Let $E(x)$, $D(x)$, $P(x)$, $T(x)$, and $C(x)$ be the utility function for, respectively, energy consumption, end-to-end delay, packet delay ratio, throughput, and convergence time. Based on the minimum and maximum values achieved, a utility function U is defined as follows : $U(x) = a * (x - Max_{metric})/(Max_{metric} - Min_{metric})$, where a equals -1 if the minimum metric value is the preferred one, 1 if the maximum value is more desirable. Max_{metric} and Min_{metric} represent respectively the maximum and minimum values for each metric in Table 3.3. Thus, we obtain the following expressions for the metrics utility functions :

$$
\begin{aligned}
E(x) &= (-x + 11178)/8685.3 \\
D(x) &= (-x + 1.55)/1.24 \\
P(x) &= (x - 94.58)/4.51 \\
T(x) &= (x - 2.84)/0.32 \\
C(x) &= (-x + 310)/240
\end{aligned}
$$

After defining the utility functions, a preference (score) is set to each criterion on a 5-level scale. The preferences are to be specified based on the application requirements. In this example, we consider an environment surveillance application with multimedia traffic and an industrial application. The former requires mainly high throughput and energy efficiency, while the latter has strict requirements for real-time in terms of delay and convergence time. The convergence time is crucial in an industrial setting, especially when a network failure occurs and requires a quick reconfiguration of the network. Despite its importance, this metric has never been considered in evaluating TSCH schedulers. Table 3.4 defines the weighting of each criterion according to the target application.

Finally, an aggregation function is used in order to choose which scheduler is best suited for a specific application. It assigns a score to each solution by accumulating the utility functions of the predefined criteria, multiplied by their respective weights. Let $w_1, w_2, .., w_n$ be the weights of criterion $1, 2, .., n$ and $F_1, F_2, .., F_n$ their respective utility functions. We can characterize a solution (or a scheduler in our case) with a vector, $X = (x_1, x_2, ..x_n)$. The aggregation function

Scheduler	Surveillance application (A_1)	Industrial application (A_2)
QL-TSCH-plus-sf5	0.559	**0.759**
QL-TSCH-sf5	0.541	0.714
Orchestra	**0.833**	0.697
ALICE_sf7	0.447	0.560
ALICE_sf19	0.690	0.443
ALICE_mc_sf7	0.425	0.611
ALICE_mc_sf19	0.708	0.5
MSF	0.716	0.370

TABLE 3.5 – Aggregation values

is defined as the weighted average of the utility functions : $A(X) = \sum_i w_i \ F_i(x_i)$. This way, the aggregation function for the environment surveillance and the industrial application respectively, are given by :

$$A_1(X) = 0.33 \ E(x_1) + 0.07 \ D(x_2) + 0.13 \ P(x_3) + 0.27 \ T(x_4) + 0.2 \ C(x_5)$$
$$A_2(X) = 0.19 \ E(x_1) + 0.31 \ D(x_2) + 0.12 \ P(x_3) + 0.06 \ T(x_4) + 0.31 \ C(x_5)$$

Table 3.5 lists the values of the aggregation functions for each scheduler, both in environmental surveillance (aggregation function A_1) and industrial contexts (aggregation function A_2). This table shows that Orchestra is the top choice for environmental surveillance applications, where the main needs are energy efficiency and high throughput. Meanwhile, industrial applications need to meet strict real-time requirements in terms of delay and convergence time necessary for rapid network reconfiguration. Here, QL-TSCH-plus-sf5 proves to be the most suitable scheduler, with the highest aggregation function value of 0.759.

3.4 Conclusion and Perspectives

In the first part of this chapter, we have proposed QL-TSCH-plus, a distributed version of QL-TSCH that reduces considerably its energy consumption (by up to 47%) without compromising network reliability and timeliness.

Evaluating decentralized schedulers including autonomous, distributed and RL-based ones under highly heterogeneous traffic conditions has not been investigated in the literature. Hence, we defined a scenario including two heterogeneous traffic flows (heavy and light) to assess a selection of the most representative decentralized schedulers based on a unified environment under the same network conditions and metrics. The experiments results allowed us to gain insights into the behavior of the three types of schedulers under heterogeneous traffic conditions, present in Industry 4.0 settings. Finally, a multi-criteria analysis is conducted to showcase the process of deciding which scheduler is best suited for a specific application. The obtained results highlight the importance of the enhancement proposed in the first part of this chapter, QL-TSCH-plus. It proved to be the most suitable scheduler in industrial settings where fulfilling real-time requirements is crucial. On the other hand, Orchestra deemed to be the scheduler of choice in environmental surveillance applications.

In contrast, when all metrics are given equal importance, conventional schedulers like MSF and Orchestra outperform those based on RL. This can be interpreted as traditional schedulers being more versatile and adaptable to a diverse range of applications compared to RL-based

schedulers. Thus, more effort must be made in RL-based TSCH scheduling to be able to attain the maturity level of traditional schedulers. For instance, the combination of RL-based timeslot selection and channel blacklisting will further mitigate the effects of interference and multi-path fading in TSCH networks, thus, increasing performance.

While RL-based scheduling is flexible and can adjust to changing conditions, it may face hurdles in highly dynamic traffic conditions. Its need for continuous learning means that when traffic intensities fluctuate, the algorithm may struggle to maintain good network performance. This challenge is amplified by the extended time required for the algorithm to adapt to new traffic patterns, especially when it operates within a large search space. In this chapter, QL-TSCH and QL-TSCH-plus were set up with a slotframe size of 5, which is a relatively small search space. However, increasing this size to improve energy efficiency, for instance, could potentially reduce the algorithm performance and responsiveness. A larger slotframe size means the algorithm may take longer to relearn the optimal transmission timeslot, impacting its overall efficiency in dynamic traffic scenarios. Considering Deep RL-based schedulers in a centralized fashion is promising based on the state-of-the-art, as they can address the challenges of highly dynamic traffic intensities with their generalization capabilities. However, integrating these algorithms in a real network stack is still a research gap and requires new innovative architectures.

This chapter has been derived from two of our papers. The first one about QL-TSCH-plus is accepted to be presented and published in The 9th IEEE World Forum on Internet of Things (IEEE WFIoT2023) (Kherbache, Maimour et Eric Rondeau, 2023). The second regarding the evaluation of decentralized schedulers under highly heterogeneous traffic conditions is published in Elsevier Internet of Things journal (Kherbache, Sobirov et al., 2023).

In the next chapter, we dive into the integration of Digital Twins with SDN for the IIoT. We propose a holistic NDT architecture to enable closed-loop network management across the entire network life-cycle, including design and service phases. The proposed architecture is validated through its application to the early design stage of an industrial project with real-time requirements. Moreover, we present the NDT core architecture describing its main internal components and their interaction. Particularly, it allows the integration of Deep RL TSCH scheduling algorithms in the IIoT.

Chapter 4

Digital Twin and Network Softwarization Integration in the IIoT Landscape

IIoT networks are known for their complexity and the challenges they present, given their role in controlling industrial applications. These applications often come with a myriad of requirements, with high reliability, strict real-time constraints, and energy requirements being paramount among them. Despite the growing interest in research related to NDTs, there still exists a notable gap in articulating a comprehensive architecture for a network digital twin that is tailored for the IIoT.

In this chapter, we propose a holistic NDT architecture for the IIoT to enable closed-loop network management across the entire network life-cycle. This fosters a progression from the current network design methodology to a more dynamic one. In fact, the NDT allows to leverage the output from both twins to suggest improvements on the designed networking protocols and algorithms. An ongoing evolution of the physical network is possible through interventions during the network service phase with the goal of maximized performance. In practice, we choose to leverage the SDN paradigm as an expression of network softwarization. SDN decouples the data plane from the control plane and allows centralized network orchestration (Kreutz et al., 2015). Controllers form the control plane and hold the control of the network by sending instructions and commands to the devices in the data plane. They also collect the required information from the physical network to build a centralized global view. This two-way connection can be exploited by the NDT for real-time network monitoring, predictive maintenance mechanisms and network diagnostics. On the other hand, one should carefully design the NDT core with several factors in mind to enable these intelligent services. This involves considering how we gather and store data, how we use this data to create a digital model of the network, and how we set up and request services to meet the needs of the applications. That is why, we also present the NDT core architecture to answer these latter questions and detail the main components of an NDT for the IIoT.

To validate the proposed holistic architecture, an industrial project aiming to connect a Flexible Production System (FPS) to the Internet using sensor networks is considered. The concept of NDT is used in the early stage of the project. One design issue relates to choosing the communication mechanism that suits the real-time requirements of the FPS application. An NDT is built to allow the assessment of different networking policies that aim to achieve reliability and timeliness prior to the deployment of the most suitable one.

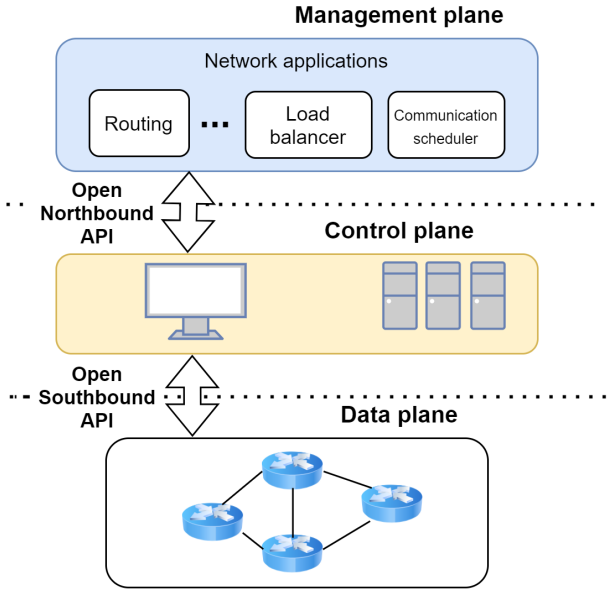


FIGURE 4.1 – Simplified view of SDN architecture.

4.1 Software Defined Networks

According to the Software-Defined Networking paradigm, computer networks can be split in three planes : management, control and data planes as depicted in Figure 4.1. The data plane is composed of the forwarding devices (switches, routers..etc). The control plane is responsible for sending commands to the forwarding devices in order for them to apply the required networking policy. The management plane is represented by network management applications such as those related to traffic engineering, mobility management and wireless communications, security and reliability. These applications are implemented using network programming languages and they interact with the control plane thanks to an open northbound API (Application Programming Interface).The control plane controls the forwarding devices via an open southbound API such as OpenFlow (MCKEOWN et al., 2008).

This three-plane division allows for more flexibility, scalability and performance compared to traditional networking. In fact, SDN abstracts network intelligence from forwarding devices to a logically centralized controller. This simplifies network management, configuration and evolution (Hyojoon KIM et FEAMSTER, 2013). It can be defined as a network architecture with four characteristics according to (KREUTZ et al., 2015) : i) the control and data plane are decoupled ; ii) flow-based forwarding, meaning that routing decisions will be made based on packet properties instead of its destination, which offers high network flexibility ; iii) control logic is moved to an external entity, called SDN controller, which facilitates the programming of network devices by providing an abstract global view of the network ; iv) the network is programmable by the means of software applications implemented on top of the control logic.

4.2 A holistic Digital Twinning Architecture for the IIoT

The process of designing and validating network solutions can go through a theoretical analysis as a preliminary step to prove the underlying algorithms convergence and their correctness. On the other hand, simulation (or emulation) tools are widely used by network researchers to

develop and evaluate their algorithms and protocols. This is due to the fact that these tools are a good way to quickly test protocols on a large scale at a low cost. To evolve the developed solution, the process is repeated as in Agile methodology using inputs from the previous steps and eventually from experimental validation and deployment steps, until it is fully functional and ready for deployment. These iterations are done off-line implying human intervention which makes this approach prone to errors. We argue that the problem of this methodology is the lack of connection with the real world network throughout the entire lifecycle from the first to the final phase. In other words, the coordination between the different steps can be quite challenging. Moreover, with this approach, the designed solution can only be completely validated at the end of the deployment phase. This is further exacerbated in the context of Wireless Sensor Networks, a basic building block of the IoT.

We envisage that the industrial IoT is a complex system since it is characterized by a large network of components, many-to-many communication channels and sophisticated information processing that makes prediction of system states difficult (Mitchell, 2009). We would contend that complex systems have a major element of surprise, as in "*I didn't see that coming*". That surprise is generally, although not always, an unwelcome one. So the element of surprise is not to be ignored. In fact, in the real world, many factors can impact the operation of a WSN : outdour conditions, weather conditions, radio interference from other wireless technologies, etc. The LOFAR-agro project presented in Langendoen, Baggio et Visser, 2006 is the perfect example on how things can go extremely wrong after deployment. The project members intended to deploy a large scale WSN of up to 100 nodes for a pilot in precision agriculture. Everything worked fine in simulation and in short-scale deployment (10 nodes), but during the real world deployment, they faced an endless stream of hardware malfunctions, programming bugs, software incompatibilities, combined with the harsh nature conditions and time pressure. That made them face unsolvable problems due to the layering of the different problems and, in our opinion, the lack of continuous connection between the real world network and the design/validation process.

In the networking field, it is known that simulation is not a tool for fully validating a solution since it cannot take into account all the environmental variables surrounding the network although it helps provide a better understanding of current performances. On the other hand, deployment testing is costly in terms of time and money. That is why, we are proposing a novel Digital Twinning based architecture for the IIoT that fosters closed-loop network management across the entire network life-cycle from the early design stages to the service and maintenance phases. To do so, we introduce the concept of the Network Digital Twin.

By creating a digital twin of the industrial network, a *'living model'* that is constantly updated, decisions are, therefore, made based on current conditions rather than those of the original form. As depicted in Figure 4.2, modeling and analysis can be tightly coupled with execution, enabling a cycle of continuous improvement and innovation. Enhancing network reliability and dealing with network risks in advance by predicting future network status using, for instance, AI algorithms in the digital twin. Improved performance can also be achieved by adjusting network configuration based on different options, adaptation to evolving traffic and resource demands and by experimenting safely different solutions to determine the optimal configuration of networks without jeopardizing the operation of the physical network. This eliminates the risks related to testing new network policies in a production environment and decreases the corresponding costs since the experiments are done in the network digital twin.

In order to implement the proposed design approach, we propose a holistic architecture comprising three main parts as shown in Figure 4.3. A physical world composed of physical entities which could be any industrial object/system such as a 3D printer, an oil platform, a conveyor, a machine, etc. Each physical plant is equipped with wireless sensor nodes that are

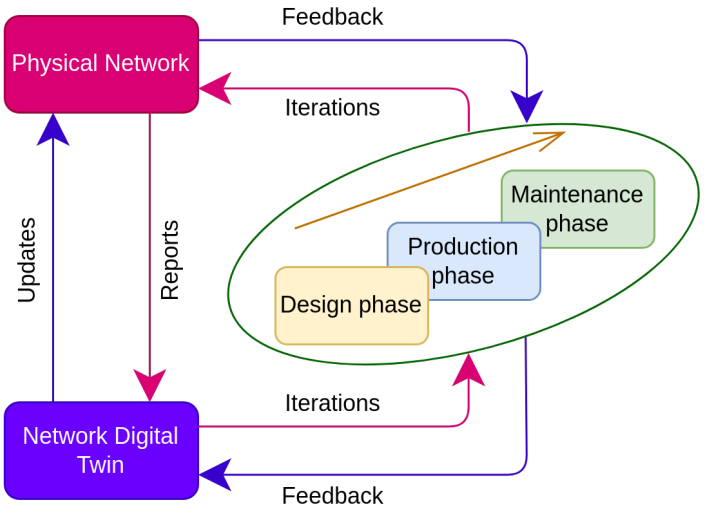


FIGURE 4.2 – NDT role throughout the whole network development process

responsible for collecting data on their operating conditions. The second part consists of the cyber world that integrates a digital twin for each industrial system present in the physical world all interacting with the NDT, the digital twin of the industrial network. The physical and cyber worlds are connected via an SDN controller that acts as a bridge between the two worlds, forwarding information flows from the cyber world to the physical world and vice-versa. With this approach, all the data describing the physical network is captured by the SDN Controller that constructs the network topology model and provides the necessary intelligence to the NDT. Moreover, commands sent by the NDT are forwarded to the physical network through the SDN controller. The SDN paradigm is adopted since it facilitates the management of networks, enables network centralization, allows network programmability and also network slicing when combined with Network Function Virtualization (NFV). NFV makes it possible to run multiple applications on the same network infrastructure while network slicing can divide the network into virtual slices independent from each other and each tailored to satisfy the requirements of a specific application. The capabilities brought by these technologies play an essential role in unlocking the full potential of the NDT.

When it comes to practical implementation of our architecture, we need to consider the constrained nature of the IIoT in terms of computation and storage means as well as energy. By including an NDT in the architecture, more data has to be exchanged in a bidirectional way with more packets processing to ensure the synchronization between the physical and digital network. This leads to increasing the traffic load in the network and the nodes energy consumption. So, the challenge when implementing an NDT would be to find a trade-off between energy consumption and the effective execution of digital twinning operations. For example, the NDT would invoke more energy consumption in the early stages of the network operations (due to the high amount of information that should be exchanged to synchronize the two sides) but once it becomes stable, it can apply mechanisms that should increase the network's remaining useful life.

In what follows, we discuss some of the benefits of this architecture in the design and the service phases. The latter includes production and maintenance.

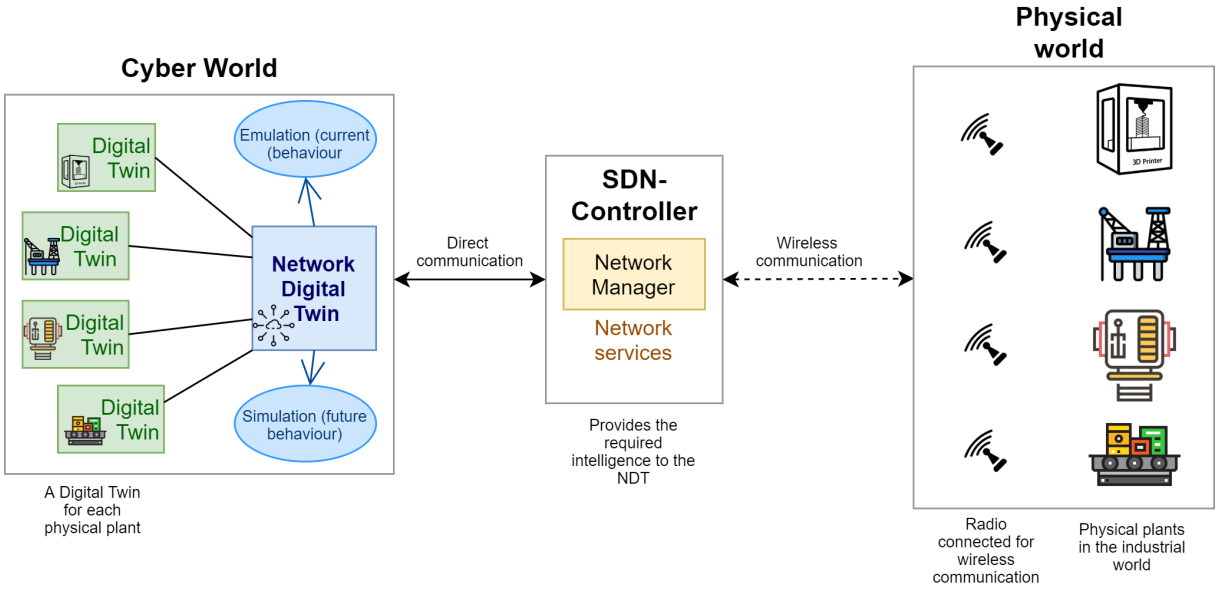


FIGURE 4.3 – Digital Twin Networking based architecture for industry 4.0

4.2.1 Design phase

With the proposed architecture, it is possible for the NDT to leverage collected data from the real world to provide insights on the changes to be made to the designed network solution in order to ensure the intended operation and get the required performance, thus validating the solution more quickly. Moreover, data visualization and analysis tools can be implemented in the NDT which may help network designers to accurately interpret the behavior of a network protocol and analyze the interactive behaviors among the network components. More interestingly, the NDT would permit testing new network solutions under various conditions, having more accurate results than current simulation tools because the NDT can take into consideration the surrounding environmental conditions and provides more immersive experiments attributed to its permanent connection with the real world. This helps network designers to detect and eradicate the eventual surprising undesirable behaviors that could occur in the deployment environment. Last but not least, the NDT can interact continuously with industrial systems digital twins to get insights on the networking requirements of each one and adjust the network's resource allocation policy accordingly.

4.2.2 Service Phase

Our architecture allows a continuous evolution of the physical network. In fact, the NDT would provide continuous real-time remote network monitoring based on the information it receives from the physical network. In addition, predictive maintenance mechanisms could be implemented in the NDT using AI algorithms and specific network policies could be applied to increase the network lifetime. Network diagnostics can also be ensured by the NDT. Based on the reports generated, it adapts the implemented mechanisms continuously to improve their efficiency. Also, the NDT can take appropriate actions to ensure that the requirements of network applications are met by accommodating network configuration. For instance, when multiple network applications are running on the same network stack, the NDT can provide a network configuration that makes a balance between the current network capacity and the applications

requirements.

4.3 The NDT core detailed architecture

The NDT enables multiple intelligent features due to the data collected continuously from the physical network. This data can be translated to knowledge by intelligent algorithms and simulation capabilities integrated in the NDT. In fact, the NDT allows predictive maintenance, efficient energy optimization, optimized resource allocation, and real-time network monitoring, among others. Furthermore, interoperability between different networking devices can be ensured since their corresponding components in the digital world are platform independent and they can be managed following common mechanisms considering the technicalities of the different devices. To enable such intelligent services, the NDT should be carefully crafted to take into consideration how data is collected and stored, how it is processed to model the network in the digital world and how services are implemented and requested to satisfy the applications requirements. A detailed architecture of the NDT is proposed in Figure 4.4.

The physical network interacts with the SDN-Controller to share relevant data such as nodes general information (e.g., address, radio interference range, the list of direct neighbors, etc.) or data representing the node's state (e.g., energy level, link quality, number of sent packets, number of received packets, etc.). The SDN-Controller interprets the received node's general information data to construct a global topology and stores the node's state information as well as the constructed topology in the data lake. Also, the SDN-Controller acts as an interface between the NDT and the physical counterpart in a way that it can interpret service requests from the real network and translate them so that the NDT can understand them and provide the requested service. In addition, the SDN-Controller is responsible for applying the network decisions made by the NDT in the real network.

The data lake is part of the NDT and it is considered as a database where different data structures are stored and retrieved. Logical models of the network can be built based on the data provided by the data lake. They can either be basic models representing the network components or they can be more advanced models including more features for functional purposes such as anomaly detection, communication scheduling, security management, among others.

Digital Twin Entity Orchestrator (DTEO) is the management entity of the whole NDT system. Requesting models to assess the network performance by executing different what-if scenarios and learning continuously the best network parameters. Setting simulation parameters into a simulation/emulation asset and getting back performance metrics that should help in improving the network operation. This orchestrator leverages intelligent networking services to satisfy the applications requirements.

Real-time network monitoring is implemented in the NDT by interacting with the DTEO and also requesting the intelligent networking services when needed. It can be seen as the human-machine interface of the NDT system since it includes a visualization tool where the network engineer can intervene when needed to better assist the network.

The NDT exposes the networking capabilities to the applications which in return send their QoS/QoE (Quality of Service/Quality of Experience) requirements. An interface between the NDT and the applications is provided to translate the networking capabilities to the applications and also the QoS/QoE requirements to the NDT. Based on the sent applications requirements, the NDT finds the best networking configuration that should satisfy these needs.

Intelligent networking services are implemented in a modular fashion and can be requested anytime by the DTEO or the network monitoring tool :

Predictive maintenance. the objective of this service is to predict network failures in advance, AI algorithms could be leveraged for this purpose based on data related to the networking operation. In addition to the historical data collected from the physical network, possible what-if scenarios could be simulated to get more insights and further enrich the dataset used for predictive maintenance.

Network diagnosis. through this service, the network behavior is observed continuously and possible breakdowns such as node failures, link failures, network misconfigurations, security attacks, etc. can be detected. Comparing the NDT behavior with the real network's behavior continuously and whenever an unexpected behavior in the physical network occurs, an anomaly arises. This service outputs diagnosis reports that can be visualized in the network monitoring tool giving the network engineer insights on the networking operation. Based on these reports, accommodation of the current network policies is performed or in more severe cases the network is reconfigured to overcome the detected anomalies.

Security management. this service enables testing security attacks safely in the NDT in order to detect possible security breaches and reinforce the network's security.

Resource Allocation. this service allows efficient resource allocation for scheduling the communication between the wireless network nodes. RL is a potential algorithm for this purpose since it can provide a general policy that can give the best communication scheduling valid for various scenarios. Taking advantage of the NDT, the learning phase will be executed in a replicated version of the real network and continuous enhancements of communications scheduling can be ensured. During training, the actions chosen in each episode can be simulated and evaluated in the NDT. This latter returns the reward to the RL agent based on the simulation performance and the training agent iterates in this way until converging to an optimal policy. This policy can be stored in the SDN controller which propagates scheduling commands to the network nodes. Moreover, it can be updated periodically to cope with new networking scenarios.

Interoperability. this service is responsible for ensuring the interoperability between the different application digital twins and breaking the software silos between their various closed platforms or interfaces. It includes formatting capabilities allowing the translation of multiple closed platforms/interfaces specificities to a common format understood by the NDT.

Mapping and Mobility. this service aims at tracking the network nodes and can provide their location coordinates or even estimate them in case of the unavailability of such information. It plays an essential role in the sustainability service, e.g., giving the location of a node that should be removed or replaced for environmental purposes.

Energy Optimization. this service has the objective of increasing the network's lifetime by providing mechanisms that should reduce the energy consumption of the network. It can be requested for example when a node's energy level has reached a certain threshold in order to reduce its usage by adapting the deployed network policies according to that.

Sustainability. dedicated sustainability features provide a global view of the environmental footprint of the IIoT infrastructure. They take into account issues related to energy, battery management to change and recycle batteries, management of wireless nodes that will have to be

replaced and removed to avoid contaminating nature in the case of environmental monitoring applications, monitoring the impact of radio waves, etc. Node obsolescence models integrated in the digital twin coupled with real system information allow to anticipate maintenance and recycling phases. This module leverages the *Mapping and Mobility.* service providing the location of the wireless nodes when an intervention should be carried out on them (e.g., replacement, removal). This sustainability aspect is essential, given that in 2025, the number of connected objects is estimated at more than 75 billion.

4.4 Industrial Case Study

The proposed architecture in Figure 4.3 can be applied in many DT-based architectures to allow an efficient connection between the real and the digital worlds. In the personalized production and distributed manufacturing context for instance, a digital twin for a connected micro smart factory is designed and implemented in K. T. Park et al., 2019. The DT uses an IIoT network to ensure the synchronization with the physical manufacturing components. This synchronization allows the DT to monitor the present in real-time, to track the past, and make predictions to support decision-making for the future. The NDT concept can be included in this architecture to manage the IIoT network and boost its performance.

In order to validate the proposed Network Digital Twin architecture, we consider its application to the early design stage of an industrial project. The purpose of this project is to satisfy the real-time requirements of a control application that monitors the operation of a Flexible Production System (Tolio, 2008). To do so, a WSN needs to be deployed to allow collecting information on the manufacturing operations carried out by the FPS. One raised question concerns which mechanism allows to meet our real-time requirements. As a preliminary step, an NDT is built to assess the performance of the platform equipped with the WSN under three MAC protocols along with an oversampling mechanism. Figure 4.5 depicts the practical scheme of our proposed NDT architecture adapted to the considered industrial case study.

In the physical world, there is the industrial platform that consists in an FPS installed in approximately a $20m^2$ area within our university. This platform aims to support teaching activities in automation engineering, industrial supervision, industrial communication, control and system integration. The FPS is composed of six assembly stations connected by a conveyor where each station is equipped with a Programmable Logic Controller (PLC). A sensor node is installed at each station in order to report information on its operation process to a central node suspended in the ceiling centrally above the FPS.

In order to carry out the design of our project, we create an NDT to get insights on the most appropriate choices with regard to network protocols to satisfy the real-time constraints of our FPS. That is, in the cyber world, Cooja simulator (Osterlind et al., 2006) is used in order to replicate the behavior of the WSN deployed in the physical world. To do so, the different distances between the sensor nodes and the central node (the Sink) are measured and the corresponding topology in Cooja is reproduced as shown on the left side of Figure 4.5. Green node numbered 1 is the Sink and it is located at an altitude of 2.35 m with respect to sensor nodes.

In order to allow communication between the real and the cyber worlds, SDN-WISE (Galluccio et al., 2015 ; Anadiotis et al., 2019) is used. SDN-WISE focuses on network flexibility and security with the goal of making WSNs modular in terms of communication and processing, reducing the amount of information exchanged between the sensor nodes and the SDN controller, and making the nodes programmable as finite state machines. The SDN-WISE controller keeps track of the network topology using a graph where vertices are the nodes and the edges are the links

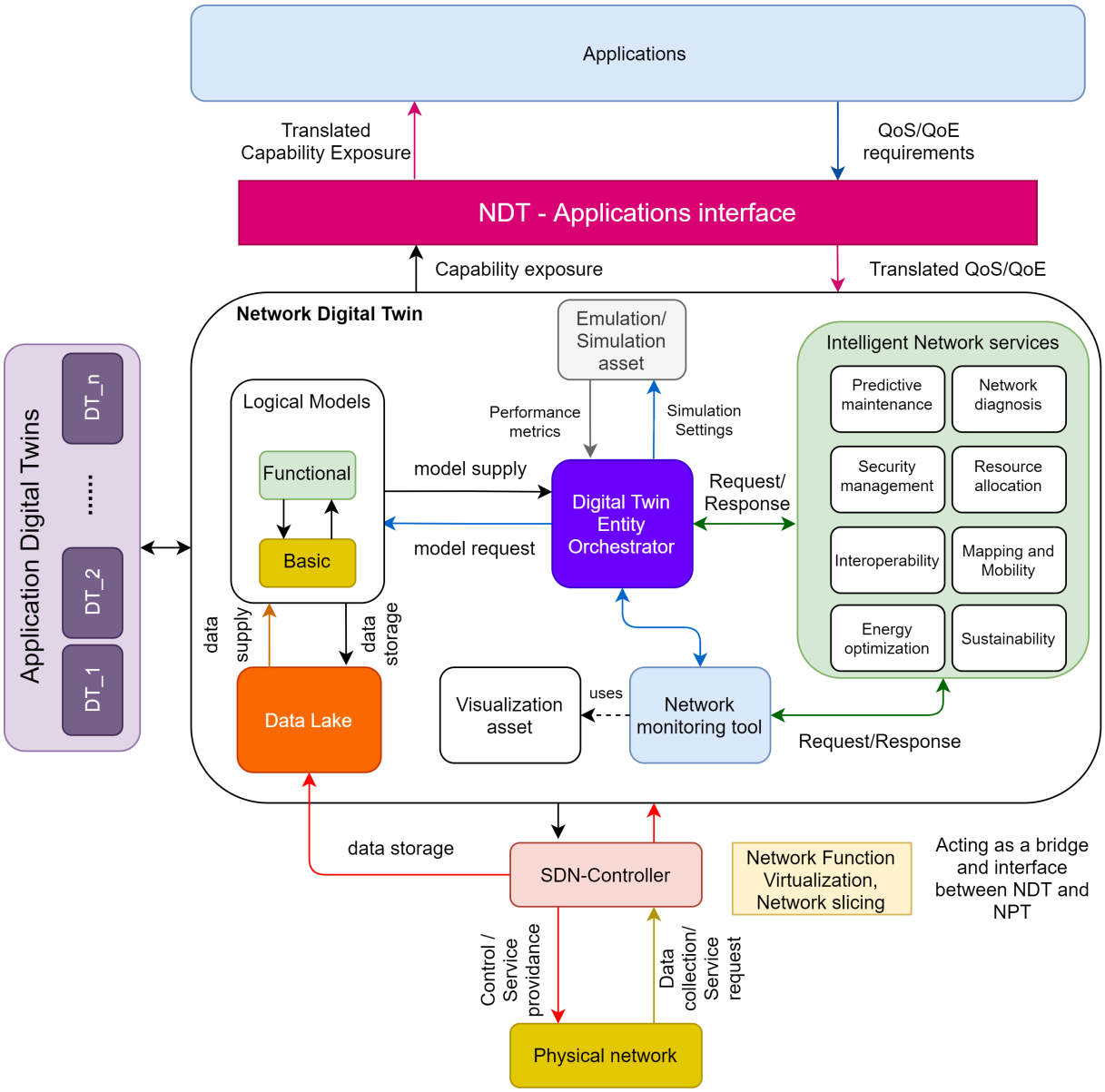


FIGURE 4.4 – Network Digital Twin core architecture for the IIoT.

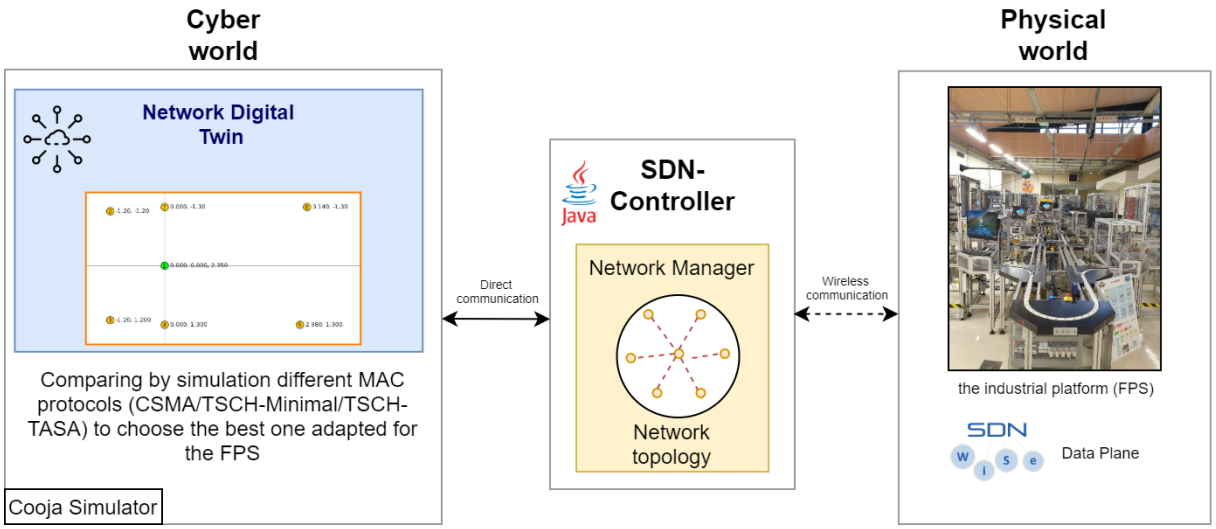


FIGURE 4.5 – Practical architecture adapted to our industrial case study.

between these nodes. In SDN-WISE, the sink is the intermediary between the sensor nodes and the controller. It starts by broadcasting a packet called "beacon" that contains the identity of the sink that generated it, a battery level, and the current distance from the sink which is initially set to 0. A neighbor node, upon receiving such a packet, inserts the source node in its neighbor nodes list. If the current distance from the Sink is better than the one it holds then the source node becomes its next hop to the Sink. The current distance is incremented in the beacon packet before it is rebroadcast. After constructing its list of neighbors using the different "beacon" packets received, a node generates a "report" packet containing its current list of neighbors and sends it to the controller. This latter, uses report packets to construct a global view of the network. This protocol is run periodically to ensure that the controller always have an updated view of the network. The frequency of sending "beacon" and "report" packets are application specific and impacts network performance.

The process of building a global view of the network is a key feature in the proposed architecture. Even if in the considered case study, the network topology was manually defined, one can make automatic topology discovery. This may be useful to consider already deployed networks in harsh environments. More interestingly, any change in the physical network, mainly during the service phase, would be detected by the controller which updates the digital replica consequently. These dynamics can be accommodated by the NDT and the running solution can be updated accordingly. In the proposed practical implementation, the network topology can be described within the XML (Extensible Markup Language) file that describes the scenario to run by Cooja simulator.

4.4.1 Real-Time Requirements

In an industrial environment, the control of automation applications is usually based on cyclic processes that run according to a predefined sampling. The sampling period must allow the control system to be updated while counting for the communication overhead. In order for the system to be updated every period, it must therefore be ensured that all communications arrive within the period. If a network message gets lost then the controller will be deprived of fresh input data and/or a remote control action will not be executed. Reliability and timeliness

TABLE 4.1 – Parameters used for TSCH with the Traffic Aware Scheduling Algorithm scenario

Parameter	Value
Period of sending "*beacon*" packets	5 seconds
Period of sending "*report*" packets	10 seconds
Slotframe size	15 timeslots
Timeslot's duration	10ms

are of a paramount importance in an industrial application. Since, we aim to endow our FPS with a WSN, it is worth noting that real-time communication in WSNs is more challenging due to their severe constraints in terms of processing and communication means in addition to the unreliable nature of the wireless medium and its shared access.

To overcome the above mentioned problem, a common solution is to apply an oversampling mechanism where a message is sent more than once in the sampling period to ensure the timely delivery of at least one copy of this message. The drawback of this solution is that sending multiple copies of each message within a given period would lead to network overloading due to congestion which decreases reliability and increases the experienced delays. As a result, a careful setting of the amount of redundancy to apply is crucial to ensure timeliness without affecting the application reliability.

Another solution consists in adopting a contention free access protocol. This includes TSCH protocol which received the attention of both academia and industry due to its high performance in real-time constrained applications. As previously mentioned in Chapter 2, TSCH targets application areas such as industrial automation and process control and offers support for multi-hop and multi-channel communications (KURUNATHAN et al., 2018). It has been designed to satisfy the requirements of IIoT applications as it provides time critical assurances and very high reliability.

4.4.2 Scenarios and Metrics

To answer our question on which solution would satisfy the real-time requirements of our industrial platform, a progressive methodology is followed. Based on the obtained results at each step, we decide whether a new mechanism is to be investigated. CSMA and TSCH minimal schedule [3] are considered first as they are already provided in Contiki-NG (OIKONOMOU et al., 2022), the embedded operating system containing Cooja as a simulator. Next, Traffic Aware Scheduling Algorithm (PALATTELLA, ACCETTURA et al., 2012) is implemented and evaluated at the SDN controller. Table 4.1 presents TASA parameters setting.

The platform's refreshing time is 2 seconds so the main data rate is one packet every 2 seconds. To assess the oversampling mechanism, we considered both duplicating and triplicating a data packet within a 2-second observation period. The data transmission rate without oversampling is one packet every 2 seconds. When duplication is considered, the rate becomes 1 pps (packet per second) where one packet is sent at the beginning of the period and the second message with an offset of one second. When triplication is performed, the first message is sent at the beginning of the period, the second with an offset of 2/3 seconds and the third with a 4/3 seconds offset, resulting in a data rate of 3/2 pps. The purpose of these offsets is to avoid creating congestion in the output buffers. Consequently, in case of duplication for instance, the first message meets

3. The TSCH minimal schedule is composed of one slotframe with three timeslots and only the first one is used and shared between all nodes.

TABLE 4.2 – Percentages of packet delays exceeding 2 seconds for each scenario along with their respective PDR values.

Protocol \ Data Rate	1p/2s		2p/2s		3p/2s	
percentage of	> 2 seconds	PDR	> 2 seconds	PDR	> 2 seconds	PDR
CSMA	6%	100%	11%	100%	33%	97.5%
TSCH minimal	1%	86%	2%	85%	9%	74%

the requirements if it is received in less than two seconds, while the second message will only have one second to arrive within the time limit. It is sufficient for one of the messages to meet these constraints to conclude that the system has been refreshed. The reported data from the FPS to the controller are of boolean type, that is the payload of each transmitted packet is one byte long. Note that additionally ten bytes are used by the SDN-WISE header. Every scenario has been executed during 40 minutes and repeated four times with a different random seed in each run. The process of sending data packets starts after an initialization phase of six minutes.

In addition to two network performance indicators, namely packet delay and PDR, an application performance indicator called the Freshness Indicator (FI) is considered. The packet delay is the difference between the time of its reception by the Sink and its transmission time by a sensor node. The PDR is the ratio of the number of received packets by the Sink to the number of sent packets by the different sensor nodes of the platform. The FI is defined as the ratio of the number of periods in which at least one packet is received by the Sink within the period duration to the total number of periods of the whole simulation. In this case study, the period is 2 seconds and the total number of periods for the whole simulation is $(40 - 6) * 60/2 = 1020$ periods. We start counting after the network initialization phase, which equals six minutes in our case. In what follows, the simulations results are presented using box plots in order to visualize the distribution of obtained data points. Mean values are also plotted as empty squares inside the box plots.

4.4.3 Simulation Results

CSMA

Experiments are first conducted using CSMA as MAC protocol without oversampling (i.e. using a data rate of $1p/2s$) then an oversampling with $2p/2s$ and $3p/2s$ data rates is considered. Figure 4.6 plots the obtained delays (log-scale) when using CSMA for each sensor node numbered 2 to 7 in the x-axis. The horizontal line at ordinate 2 seconds recalls the refreshing time period. It is noted that all nodes obtain similar latencies for the different settings. This is due to the fact that they are located almost at similar distances from the Sink reached in one hop resulting in a star topology. In the absence of oversampling ($1p/2s$), an average delay of 719 ms with an average median delay of 575 ms is obtained. However, it is observed that some packets (about 6 % as shown in Table 4.2) arrive after a delay that exceeds 2 seconds. This translates into an average freshness indicator of 97% while the PDR achieves its maximum value (100%). The distribution of PDR and FI obtained in the different experimented MAC protocols with or without oversampling are presented in Figures 4.7 and 4.8 respectively.

Since no loss is experienced, the duplication of the transmission rate (to $2p/2s$) can be afforded by sending each packet twice in the 2-second period window. Not only a PDR of 100% is kept but also a 100% of freshness is achieved for all nodes. Delay results show an increase of the

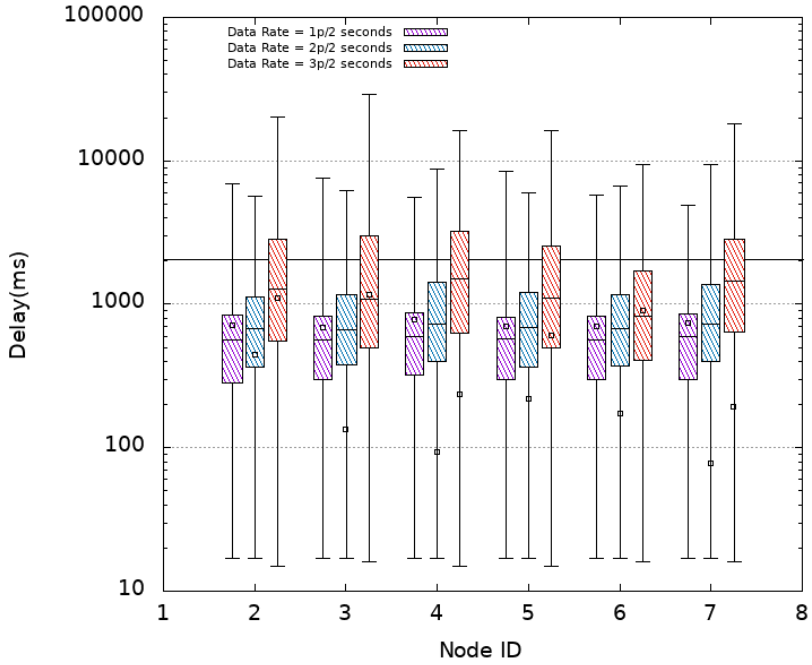


FIGURE 4.6 – Packet Delay - CSMA

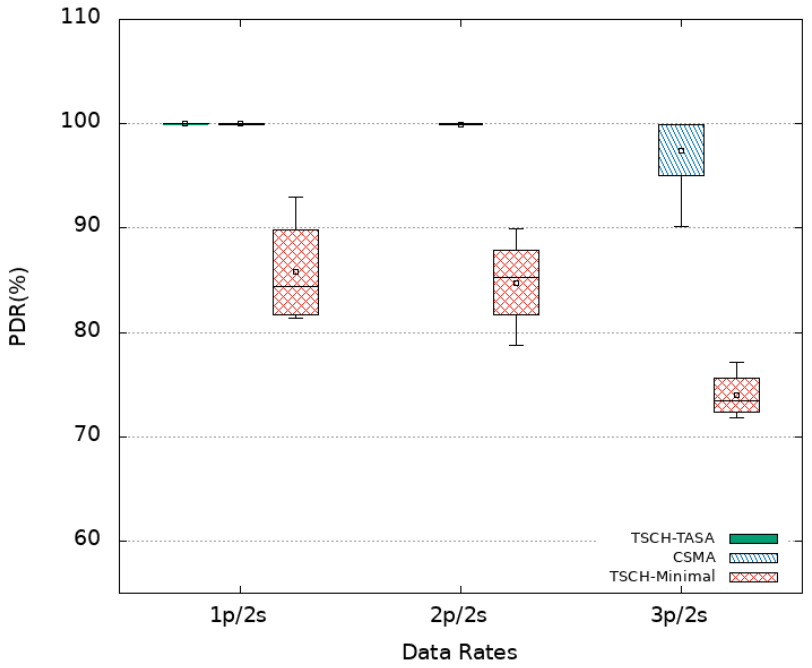


FIGURE 4.7 – PDR results.

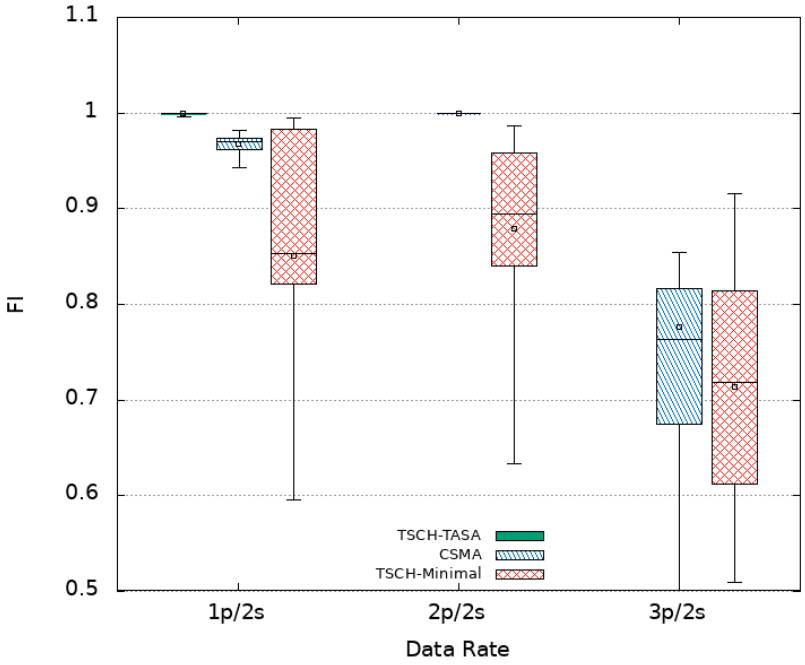


FIGURE 4.8 – FI results.

mean values, with an average of 931 ms, but the median values decreases to 696 ms. This latter explains the obtained results of the FI even if some packets still arrive out of time as depicted in Figure 4.6. Certainly, at least one copy of each message is received and losses only concern duplicates. Duplication increases the probability that a message is received within the 2-second window. Given the obtained results, triplicating seem to be of no interest. This is confirmed by our experiments where an increase is noticed in the delay results with 33% of packet delays exceed 2 seconds and a decrease in terms of PDR as an average value of 97.5% is obtained. Moreover, a significant decrease in the FI (77% in average) is to be noted. The increase of delay values is due to more experienced collisions that result from overloading the network by higher data rates. In fact, CSMA uses a contention window that imposes a waiting time (back_off time) for nodes to avoid collisions. This window doubles in size every time a collision occurs which causes greater delay values.

Minimal TSCH

As opposed to CSMA, TSCH has been introduced to meet the real-time needs of industrial applications as the nodes do not compete to access the medium. We decided to consider a TSCH-based MAC protocol in our study. Its minimal scheduling available in the Contiki-NG is considered first. Figure 4.9 presents the obtained results where lower delays can be observed when compared to CSMA. For instance, without oversampling ($1p/2s$), it is obtained in average, a mean and median delay of 133 ms and 56 ms respectively. Among received packets, only 1% are out of time as shown in Table 4.2. Despite that, the achieved PDR and FI as shown in Figures 4.7 and 4.8 are lower with an average value of 86% and 85% respectively. Even worse, the minimum value for FI may drop as low as 60% because of 1% of the packets that arrives with a delay that exceeds 2 seconds.

At this stage, is it worth applying oversampling ? A priori no, but we further increased the

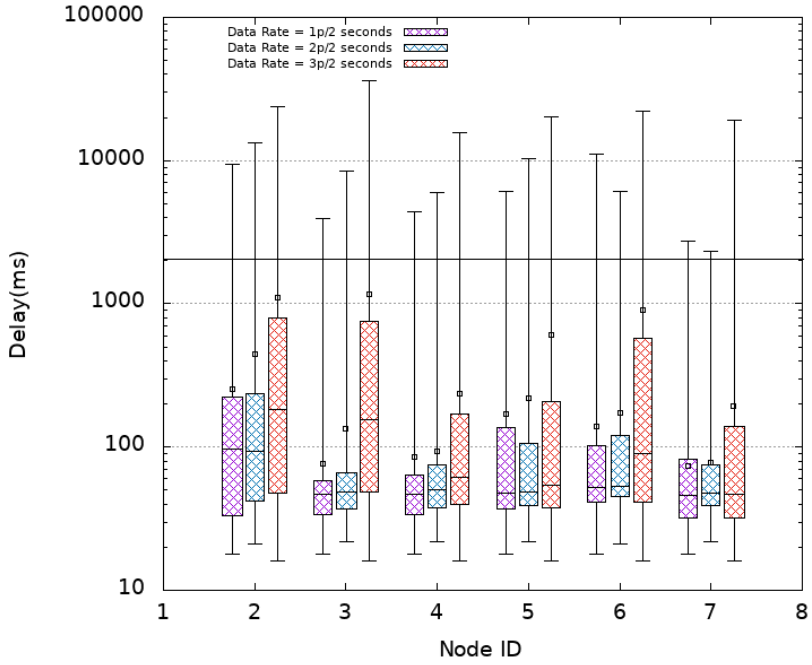


FIGURE 4.9 – Packet Delay - Minimal TSCH

transmission rate to confirm our assumptions. When duplication ($2p/2s$) is considered, the delays are slightly increased when compared to the case of $1p/2s$ with an average of 189 ms and a median of 57 ms. Duplication allows a slight improvement of the FI from 85% to 87% of the periods getting fresh data as shown in Figure 4.8 but decreases the PDR to 85% as shown in Figure 4.7. It is noted that the FI box plot spreads over a wider range with a minimum value of 66% and a maximum value of 98%. The former value results from the 2% of packets with delay exceeding 2 seconds and the latter (as well as the median) value is due to the fact that the probability of receiving a data packet within the two-second window is increased. Going further and triplicating messages ($3p/2s$) is not worth doing either since higher delays are experienced with lower PDR and FI values as we obtain 74% and 71% in average respectively.

The increase in delay when applying oversampling is due to the fact that each sensor node has more packets to transmit within the same duration which adds to the delay, the time each node spends waiting its turn to transmit. The experienced losses can be explained by the fact that scheduling with Minimal TSCH is not optimal.

TSCH TASA

The obtained results when using TSCH with its minimal schedule is inefficient and cannot suit our needs. This is why, we considered implementing and testing a more advanced scheduling algorithm called TASA. As done with the previously considered MAC protocols, we begin by experimenting TASA without oversampling i.e. using a data rate of $1p/2s$. The obtained delays as shown in Figure 4.10 fully satisfy our real time requirement as all packets arrive with a delay that does not exceed 2 seconds. Even better, almost 75% of the packets experience a latency below the 200 ms. An overall average and a median of 178 ms and 148 ms are recorded. Both the PDR and the FI achieve their maximum values (100%) as shown in Figures 4.7 and 4.8 respectively. As a result, this configuration (i.e. TASA without oversampling) answers perfectly the real-time

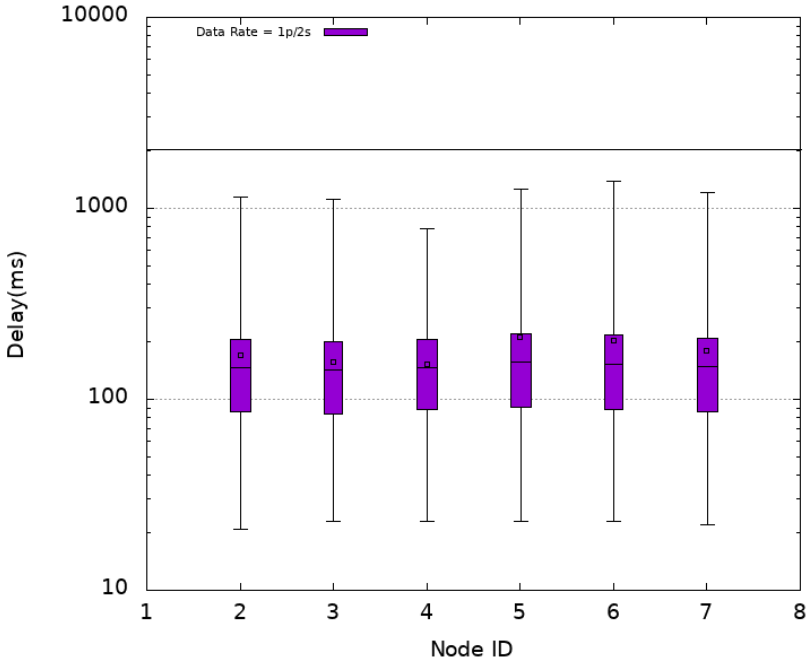


FIGURE 4.10 – TSCH/TASA delay results.

requirements of the industrial system of interest, the goal of this study. Both reliability and timeliness are achieved with the minimum transmission rate. This allows to not consume extra energy that could have been consumed by oversampling for packets replications that are no longer needed.

Following the obtained results, the corresponding firmware is uploaded to the FPS sensors using the SDN controller. This latter will have in charge to monitor their operation based on the reports sent periodically. When an anomaly is detected, the designer is alerted to correct and/or adapt the current implementation. The newly obtained firmware is then uploaded and a new "agile" iteration is undertaken.

4.5 Conclusion and Perspectives

In this chapter, we propose a holistic digital twin architecture for the IIoT where the network component is considered through the adoption of the NDT concept. The aim is to enable quick validation of networking solutions in an industrial environment since the NDT is continuously linked to the physical world from the early design stage to the production phase. Moreover, the NDT provides a continuous evolution of the industrial network by exploiting the data collected along with AI algorithms to enhance the networking performance, prevent network failure and increase the network remaining useful life. The proposed NDT can be included in any DT-based architecture where a communication network is required in the physical world. We validate the proposed holistic architecture through its application to the early design stage of an industrial project with real-time requirements. Precisely, we use the concept of NDT to assess different scenarios in order to choose the best suited communication mechanism satisfying the real time requirements of the FPS application. Our findings reveal that oversampling is no longer required when using a TDMA-like MAC protocol such as TSCH with TASA as a scheduler.

A detailed NDT core architecture has also been provided to describe the various NDT internal components and their interactions. In addition, it is designed to enable various intelligent services including resource allocation. Particularly, this allows to integrate Deep RL TSCH scheduling algorithms in the IIoT.

The next step consists of undertaking subsequent iterations to consider the effectiveness of the proposed solution in a real network. More elaborated network topologies will be considered to assess the scalability of the proposed solution to more complex industrial systems. Moreover, the adaptive character of the NDT in response to network failures in addition to its dynamics will be investigated possibly using AI-based solutions for network diagnosis and prediction.

In this work we have leveraged the SDN paradigm due to its maturity in providing centralized control, network programmability, network slicing, etc. Adopting SDN implies changing the whole network protocol stack to decouple the data plane from the control plane. However, due to the early stages of SDN proposals for WSNs, a standardized stack appears to be a distant prospect. In contrast, distributed wireless networking is a mature field with established standard protocols across each network layer. For instance, 6TiSCH presented in chapter 2 of our thesis provides a whole protocol stack composed of a set of standard protocols at each layer tailored for the IoT. Consequently, opting for SDN in the context of WSNs implies sacrificing all the extensive standardization work done in this domain. As a result, our strategy tends towards maintaining a decentralized protocol stack, while leveraging open-source tools to ensure the effective operation of the NDT.

The content of this chapter is the outcome of our two publications in MDPI Sensors journal (Kherbache, Maimour et Eric Rondeau, 2021) and IEEE WoWMoM 2022 conference (Kherbache, Maimour et Eric Rondeau, 2022b).

In the proposed NDT architectures, a simulation asset is leveraged to allow what-if performance evaluation with the aim of improving network operation. However, simulators are delicately designed and tightly coupled, leading to high execution time and computational complexity. Even though existing literature address this issue by employing Graph Neural Networks or ontologies/knowledge graphs, these modeling techniques lack the formal rigor and the capacity for real-time fault detection that are essential in mission-critical industrial applications. In the next chapter, we propose a framework for modeling the IIoT using Petri-nets within the NDT environment. Petri-nets will serve as coarse-grained formal models allowing fast what-if scenarios execution when used offline, and real-time fault-detection when used online, among other benefits.

Chapter 5

Using Petri-nets for the IIoT Modeling in the NDT

Digital Twins can help satisfy the requirements of the IIoT by enabling simulations, monitoring, and optimization of IoT devices and systems. A DT is a virtual replica of a physical system that reflects its properties, condition and behavior by means of models, information and data (Stark, Kind et Neumeyer, 2017). Also, it allows controlling the physical system with the aim of achieving increased performance and reduced costs. A unique feature of a DT is the bidirectional data exchange between itself and its physical counterpart. Telemetry is transmitted to the DT to offer insights into the state of the physical system, whereas the DT sends back control commands to regulate the behavior of the physical system.

The networking industry has shown great interest in DTs due to their immense potential in improving the field (Almasan et al., 2022). In this regard, specifically, the focus has been on NDTs. However, due to being in the early stages of development, NDTs face numerous challenges that need to be addressed including scalability, interoperability, real-time requirements, data modeling difficulties, security risks, etc (Zhou et al., 2023). Thus, the ongoing research efforts are a crucial initial step towards overcoming these obstacles. Regarding the real-time requirements challenge, it is inherited from the fact that a perfectly synchronized DT is unattainable. There will always be a replication bias and the challenge is to ensure temporal consistency between the digital twin and the physical twin. On another note, modeling the network in the NDT presents an important challenge due to the variety of possible approaches. Several works have started addressing the aforementioned challenges. To address network system modelling process related challenges, a simplified model for Data Center Networks is presented in H. Hong et al., 2021 that builds on information retrieved from configuration files and devices status. The network is represented using three models, namely, device, network and service models. GNNs are used to construct an NDT in H. Wang et al., 2022 ; Ferriol-Galmés et al., 2022, while knowledge graphs are employed in Zhu et al., 2021 ; Ma et al., 2022.

Petri-nets are widely used to model and simulate the behavior of discrete systems due to their ability to express the characteristics and the behavior of a system through strict mathematical and intuitive graphical expressions. They have been used for a long time in computer networks in the context of Local Area Networks (LAN) (Berge et al., 1994), for protocol verification (Billington, Gallasch et B. Han, 2004b) and for evaluating Networked Control Systems (NCSs) (Brahimi, E. Rondeau et Aubrun, 2008). With regards to wireless applications, Petri-nets are utilized for active product security (Zouinkhi et al., 2009) as well as network security (Xu et Xie, 2011). The integration of Petri-nets into DTs enables data-driven Petri-net

models that have the capability to simulate simultaneously both the state-driven behavior and the influence of real-time working conditions, operational parameters and production load on each individual state of the physical system.

In this chapter, Petri-nets are used as a modeling framework for the IIoT within the context of the NDT. We propose an architecture that leverages open-source tools to exemplify the process of integrating Petri-nets for the purpose of modeling the IIoT in the NDT environment. To validate the proposed architecture, Timed Petri-Nets (TPN) are leveraged to model a small scale IIoT network aiming to predict its reliability based on its operational parameters. Besides, real-time fault detection is also investigated by considering the Petri-net model as the baseline one capturing the normal behavior of the network. Finally, a study on the observability of the system is conducted as an effort to address the challenge of real-time requirements in the NDT.

5.1 Petri-nets

A Petri-net is a formal modeling technique used to represent and analyze systems that involve discrete events and concurrent activities (Giua et Silva, 2018). A Petri-net is a directed bipartite graph that consists of two types of nodes namely, places and transitions connected through directed edges called arcs. Places represent the system's states, and transitions represent the events that cause the system to change its state. The arcs indicate the flow of tokens in addition to connecting places with transitions. The tokens represent resources, information, or other entities in the system. Petri-nets are often used in the analysis and design of systems, such as computer networks, manufacturing processes, and biological systems in order to study the behavior, identify potential problems, and optimize the performance.

TPNs (Zuberek, 1991) are an extension of the classical Petri-net model, incorporating timing information to capture the temporal dynamics of a system. In a TPN, transitions or places are associated with time delays, enabling the representation of time-dependent behavior and the analysis of real-time systems. A time delay can be assigned to either transitions or places. For transitions, the time delay represents the duration that it takes to fire once it is enabled. When a transition is enabled (i.e., all input places have sufficient tokens), a timer starts, and the transition can only fire after the specified time delay has elapsed. For places, the time delay represents the duration that a token must remain in that place before it can be used to enable a transition. When a token arrives at a place with a time delay, it becomes “aged" and the transition can only be enabled once the token has “aged" for the specified time delay. TPNs are useful for modeling and analyzing systems where time constraints and deadlines are critical, such as real-time computing, communication protocols, manufacturing processes, or transportation systems. They allow for the examination of system performance under various timing conditions, enabling the identification of bottlenecks, optimization opportunities, and potential timing-related issues.

5.1.1 Petri-nets for networking

As previously stated, Petri-nets have found extensive application in computer networks, such as in LANs (Berge et al., 1994), where they are exploited for an exhaustive qualitative and quantitative analysis of the message exchanges and time constraints of a low-level industrial local network for process control. Billington, Gallasch et B. Han, 2004b present a formal methodology for specifying, analyzing, and verifying protocols using Petri-nets. Brahimi, E. Rondeau et Aubrun, 2008 develop a simulation tool based on Petri-net modeling for assessing NCSs. Additionally, Petri-nets have been employed in Zouinkhi et al., 2009 to address security concerns in industrial settings by transforming hazardous products into communicating

entities, known as active products, which monitor their environment and gather information through wireless systems. Petri-nets are used to depict the behavior of these active products and the communication flow through a wireless network, thereby promoting cooperative interaction among various products. The application of Petri-nets in analyzing network security protocols is investigated in Xu et Xie, 2011.

5.1.2 Petri-nets based digital twins

The integration of Petri-nets in the context of DTs enables data-driven Petri-nets which can efficiently simulate the behavior of the physical system while simultaneously taking into account its real-time status. Several works investigate the application prospects of such a powerful modeling technology in DTs, for instance, the development procedure of an electric vehicle is modeled using Petri-nets in a DT (G. J. Tsinarakis et al., 2020) to reduce the manufacturing duration by optimizing the production plan. In Dai et al., 2020, a DT based on Petri-nets is used for real-time simulation and feedback control of industrial pipelines. Vitor et al., 2021 analyze the dataflow of a DT in the context of Industry 4.0 using Petri-nets. Furthermore, G. Tsinarakis, Sarantinoudis et Arampatzis, 2022 creates a Petri-net model of a steel reinforcement industry using a synthesis procedure that can be used in offline or online contexts of DTs. Additionally, Hongcheng Li et al., 2022 present a data-driven hybrid Petri-net methodology for modeling energy behavior, specifically designed for DT-based energy management in manufacturing settings.

5.1.3 Research Motivation/Gap

From the aforementioned works, it is clear that network modeling should be extensively investigated for the NDT by exploring different modeling approaches. Although there has been substantial research employing data-driven techniques such as Graph Neural Networks and ontologies/knowledge graphs to model network behavior, these approaches often lack the formal rigor and the capacity for real-time fault detection that are essential in mission-critical applications. For instance, neural networks based solely on data analysis can set control rules, but they act as black boxes. Their lack of explainability is not yet accepted in highly critical industrial settings such as nuclear power plants and avionics. Additionally, they need to gather large amounts of data, consuming valuable network bandwidth - a resource we aim to conserve in the IoT. We would rather reserve this bandwidth for monitoring the industrial application rather than the network. Therefore, we choose formal modeling to not only reduce traffic but also to ensure the performance needed for industrial implementation.

Petri-nets bring a well-defined formal modeling technique to the table, excelling in both qualitative and quantitative analysis. This formal nature of Petri-nets permits mathematical rigor in the model, contributing to higher reliability and the possibility for formal verification. This is especially critical in applications requiring high levels of trust. Moreover, the intuitive graphical representation of Petri-nets aids in the visualization and understanding of complex systems behavior. This means that there's no hidden logic or unknown transformations ; the model's behavior can be traced and explained directly using the graph, ensuring explainability unlike "black box" models. Furthermore, Petri-nets are renowned for their efficiency in modeling and simulating the behavior of discrete systems (Giua et Silva, 2018), especially within a networking context (Billington, Gallasch et B. Han, 2004b ; Brahimi, E. Rondeau et Aubrun, 2008). This stems from their intrinsic properties that aptly represent concurrent behaviors, capture simultaneous operations, and precisely depict points of resource contention and

synchronization needs. Qualitatively, they can identify potential errors or bottlenecks in a network, thereby aiding in system verification. Quantitatively, Petri-nets provide a framework for evaluating network performance metrics and identifying optimization opportunities. This dual capability offers a comprehensive approach to understanding and enhancing network systems. Moreover, the temporal dimension added by Timed Petri-nets fosters the incorporation of crucial time-sensitive parameters, enabling accurate predictions of key performance metrics such as Packet Delivery Ratio (PDR). This facilitates real-time monitoring and analysis, which are essential for predictive maintenance and fault detection in IIoT networks.

To the best of our knowledge, this is a novel piece of work that seeks to leverage Petri-nets for network modeling within the context of the NDT. Particularly, an architecture based on open-source tools is proposed to exemplify the process of integrating Petri-nets for the purpose of modeling the IIoT in the NDT environment. Next, a rigorous case study is conducted in order to enable PDR prediction and real-time fault detection in IIoT networks within the context of NDTs.

5.2 A Petri-nets Based NDT

Previous research has already demonstrated the maturity and efficiency of Petri-nets for modeling network protocols behavior. Meanwhile, the data-driven approach with real-time updates from the operating network enabled by the NDT paradigm paves the way for a new era of Petri-nets utilization. In the context of NDTs, Petri-nets can be used either offline to run what-if scenarios or online by interacting in real time with the NDT for analysis of network behavior and real-time adjustments to optimize its performance.

5.2.1 Offline usage

Petri-nets can be exploited offline without direct interaction with the real network to execute what-if scenarios aiming at augmenting the NDT capabilities for better decision-making. This may include :

Performance Analysis. Petri-nets are able to perform quantitative analysis of the modeled system (Diallo, Rodrigues et Sene, 2015), thus, enabling estimation of network performance factors like throughput, latency, and reliability. Simulation of different network scenarios by varying traffic patterns and configurations helps predict potential future states of the network and identify potential bottlenecks or areas for improvement.

Network Planning and Expansion. Petri-nets can be utilized to simulate and evaluate various network expansion scenarios (Kouvatsos, 2013), such as adding new nodes, links, or implementing new network policies. By analyzing the impact of these changes on network performance, the NDT can make informed decisions on the most effective strategies for network growth.

Energy Efficiency. Petri-nets can model the energy consumption of network components (Dâmaso, Rosa et Maciel, 2014), helping to identify potential areas for energy savings. By simulating different power management strategies and network configurations, the NDT can optimize the energy efficiency of the network while maintaining performance.

Security Analysis. Petri-nets can be used to model potential security threats and vulnerabilities in a network (Henry, Layer et Zaret, 2010), allowing the NDT to evaluate the effectiveness of security measures. By simulating different attack scenarios and analyzing the response of the network, Petri-nets can help identify weak points and inform the development of more robust security strategies.

5.2.2 Online usage.

The online usage of Petri-nets for NDTs involves real-time or near real-time modeling and analysis of the physical network. In this context, the Petri-net model is continuously updated to reflect the current state of the actual network and is used to make immediate decisions or adjustments. The following are some examples of the online usage of Petri-nets for NDTs :

Congestion control. Petri-nets can be employed to model and monitor the network real-time traffic conditions, enabling dynamic congestion control mechanisms (Hafidi et Gharbi, 2019). Based on the current network state, the Petri-net model can suggest adjustments to routing policies or resource allocations to alleviate congestion and maintain optimal network performance.

Traffic Shaping and Quality of Service. Petri-nets are known for their ability to model concurrent processes (Diallo, Rodrigues et Sene, 2015), hence, they can be employed to model concurrent data flows and optimize traffic shaping policies, which can prioritize certain types of network traffic and allocate resources accordingly. This ensures that critical applications and services receive the appropriate level of QoS.

Protocol verification. Online Petri-nets models can be used to verify the correct implementation and operation of network protocols. By continuously comparing the modeled protocol behavior with the actual network operation, discrepancies can be identified, and potential errors or vulnerabilities can be addressed in real-time. Petri-nets has already been used to model network protocols such as CSMA/CA (Neto et al., 2017), TCP (Transmission Control Protocol) (Bao, F. Liu et L. Hong, 2003 ; Bitam et Alla, 2005) and MQTT (Message Queuing Telemetry Transport)(Rodríguez, Kristensen et Rutle, 2019 ; Rodriguez, Kristensen et Rutle, 2018). With the data-driven approach, one can have a Petri-nets model for each network layer including MAC protocols (CSMA, TSCH, etc.), transport protocols (TCP, UDP) and application protocols (MQTT, CoAP, etc.). These models are continuously running in the NDT and serve as verification models of the protocol stack operating in the physical network.

Real-time fault diagnosis and recovery. Petri-nets have already been used for fault detection and diagnosis in distributed systems with an application to telecommunication networks in Boubour et al., 1997. In the context of NDTs, it can function as a foundational model effectively capturing the anticipated normal behavior of the network. This enables the identification of faults when deviations arise in the actual network performance. Through continuous network monitoring and real-time fault detection, Petri-nets can initiate recovery measures, thereby reducing downtime and ensuring network stability.

Security monitoring and mitigation. This involves real-time detection and response to potential security threats. By modeling network behaviors, access control policies, and incident

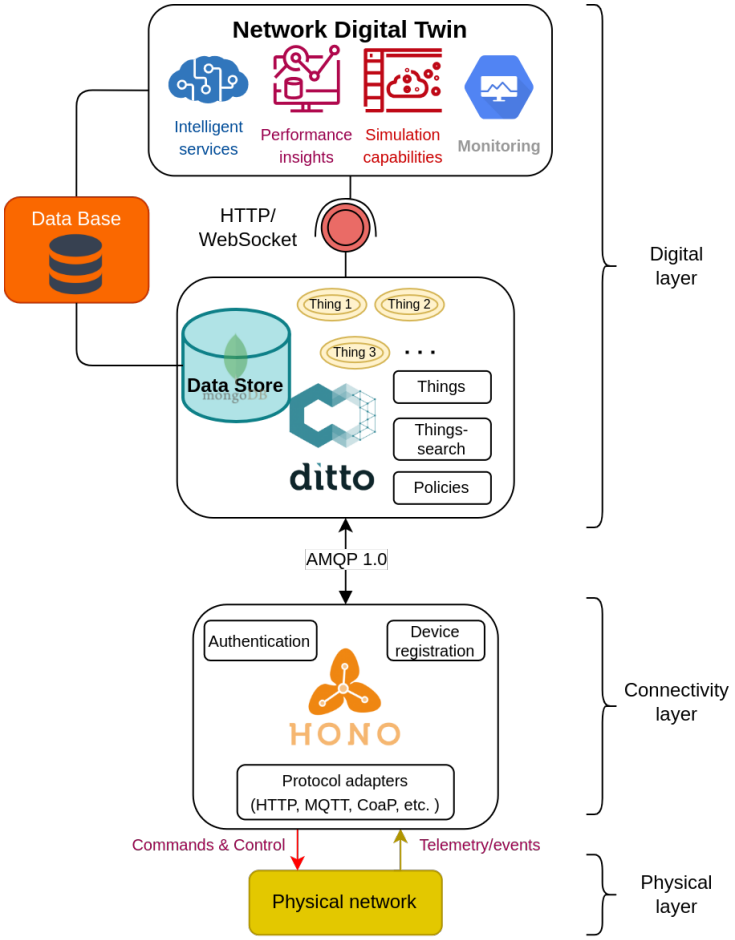


FIGURE 5.1 – Network Digital Twin using Eclipse Ditto + Hono.

impacts, Petri-nets enable the NDT to identify and address vulnerabilities, ensuring robust network security (KIVIHARJU, VENÄLÄINEN et KINNUNEN, 2009). Techniques such as intrusion detection, anomaly detection, firewall management, incident response, and continuous security assessment can be enabled and contribute towards maintaining the network integrity and stability while adapting to evolving threats.

5.2.3 Architecture in the IIoT

In this section we describe the NDT architecture for the IIoT using the open-source tools, Eclipse Hono (ECLIPSE, 2017a) and Eclipse Ditto (ECLIPSE, 2017b). As depicted in Figure 5.1, the physical IoT network interacts with the connectivity layer where Eclipse Hono is used. It is designed for simplified IoT device connectivity by eliminating the protocol silos of the different devices. In fact, for each supported protocol (e.g., HTTP, MQTT or CoAP), Hono contains a microservice called Protocol Adapter which maps the connection protocol of the device to Hono's APIs. A device registration API is included to make Hono aware of devices that can or will connect to the service. In addition, authentication API is used to verify the identity of the devices willing to connect to Hono. Telemetry and events APIs are used by protocol adapters to send telemetry/events downstream (from devices to business applications). Command and Control API is used by business applications to send commands to connected devices.

Eclipse Hono communicates with back-end business applications using Advanced Message Queuing Protocol (AMQP) 1.0. In our case it is connected to Eclipse Ditto which is an open-source frame-work for creating digital IoT twins. It focuses on back-end scenarios by providing web APIs in order to simplify working with already connected devices and things from customer apps or other back-end software. Ditto ensures that access to twins can only be done by authorized parties via the policies service. Eclipse Ditto structures the data sent by devices via Hono into Digital IoT Twins. A digital twin of an IoT device in Ditto is represented as a Thing consisting of attributes and features. The Things service is responsible for persisting things and features in Ditto. The follow-up of the modifications made to the Things, Features and Policies is ensured with the Things-search service. Also, this latter updates an optimized search index and executes queries on it.

Two channels are available for use in Ditto, a twin channel that connects to the digital representation of a Thing and a live channel that routes a command/message towards an actual connected device. Ditto uses MongoDB as a database and only stores the latest state of Things. Furthermore, Ditto provides an HTTP API and a WebSocket to applications to facilitate retrieving Things-related information. The NDT interacts with Ditto via this HTTP API or a WebSocket. The NDT stores things states in the database and updates them wherever a modification to the Things is made in Ditto. This is done because Ditto only stores the latest Things states in its data store (mongoDB). This interaction allows storing historical data of the Digital IoT Twins during the entire network lifecycle.

Figure 5.2 illustrates the IIoT-NDT implementation architecture incorporating Petri-nets modeling at the digital level. At the local level, the IIoT network protocol stack is implemented with Contiki-NG (Oikonomou et al., 2022). The application protocol is MQTT for sending data to the NDT. The IIoT motes publish telemetry to a local MQTT broker (Mosquitto (Light, 2017)) which serves as an intermediary between the physical network and its digital counterpart. In this setup, nodes transmit data to the broker using a topic named “telemetry". Simultaneously, a Python-based MQTT client subscribes to this topic, receives the published data, and forwards it to Eclipse Hono’s MQTT adapter at the cloud level. The selection of this particular setup is based on the fact that the IIoT network operates using IPv6 communication, while Eclipse Hono and Ditto exclusively support IPv4 communication. This arrangement enables seamless communication and data transfer between the actual network and its DT.

At the cloud level, Eclipse Hono acts as a connectivity layer for the IoT network where its MQTT adapter receives the published telemetry that is forwarded to Eclipse Ditto for updating the things features (updating the IoT devices state). These features are designed based on the case study and the parameters needed for the Petri-net model. The Petri-net model, implemented using SNAKES [4] library, communicates with Eclipse Ditto through its HTTP API to obtain the essential “things” features. These features are then employed to parameterize the Petri-net model. More particularly, the Petri-net model can extract numerous features’ values of a number of “things” for calculating a required parameter for its functioning, enabling data-driven Petri-net models for the NDT of the IIoT.

4. SNAKES is a versatile and user-friendly Python library designed for the creation, manipulation, and analysis of Petri-nets. Developed by Franck Pommereau, SNAKES (Pommereau, 2015) aims to provide a comprehensive toolset for working with various types of Petri-nets, including Place/Transition, High-Level, and Timed Petri-Nets. The library emphasizes simplicity and extensibility, with a clear API and a plugin system that allows users to customize and extend its functionalities.

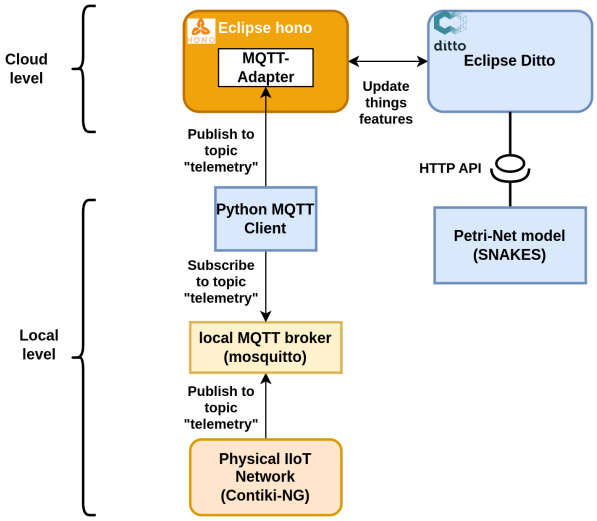


FIGURE 5.2 – IIoT-NDT implementation architecture incorporating Petri-nets.

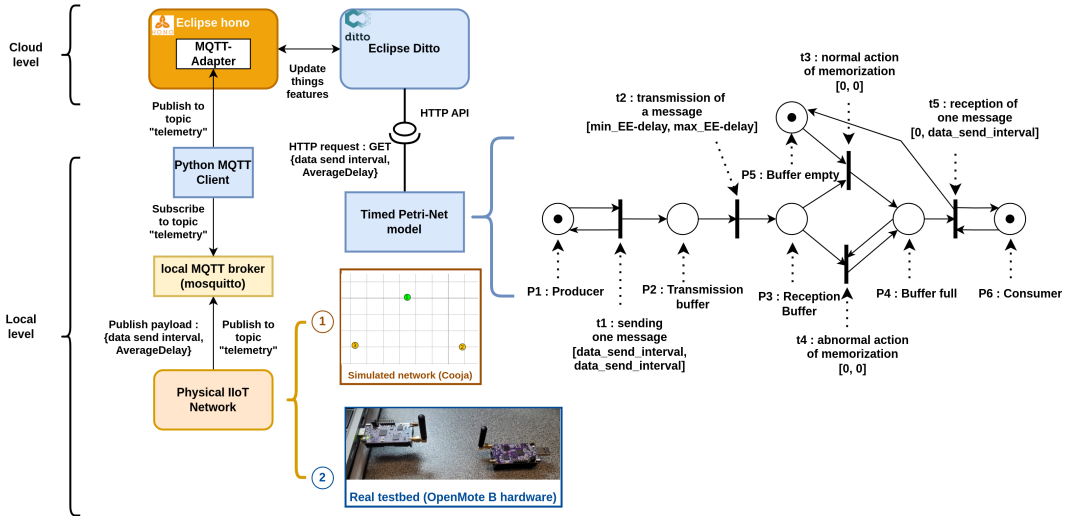


FIGURE 5.3 – Case study architecture.

5.3 Case Study

Here we provide a case study of the proposed architecture for a small scale network modeled with a TPN. Figure 5.3 illustrates the NDT technical architecture with the TPN model for the considered case study. The developed framework is available in GitHub via the link. The IIoT network is simulated initially using Cooja simulator (OSTERLIND et al., 2006) provided by Contiki-NG. Opting to use a simulated network during the initial stage of designing the NDT model allows for verification of its functionalities prior to the deployment of a real IIoT network, as illustrated in Figure 5.4. In fact, this approach employs a simulated network initially as part of a verification loop to test the NDT model, ensuring that all implemented features are operating correctly and to determine the optimal network configuration. Once this has been achieved, a real IIoT network can be deployed and integrated as an operational network. Table 5.1 summarizes the parameters of the conducted experiments scenario. RPL is used for both the simulation and the real network described below.

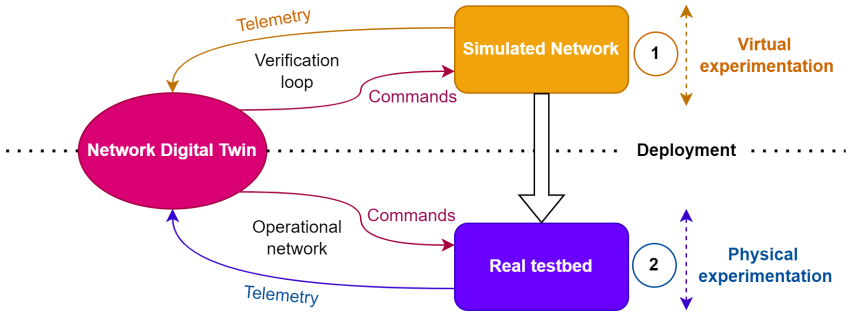


FIGURE 5.4 – NDT from design to operation.

TABLE 5.1 – Physical Network Settings

Parameter	**Value**	
	Simulation	**Real Testbed**
Mote type	Cooja Mote	OpenMote B
Data Send Intervals (ms)	100, 200, 500	300, 500
MAC protocol	CSMA	TSCH
Queue buffer size	8 packets	
MQTT publish period	20 x Data Send Interval	
Payload Size	80 bytes	
Routing Protocol	RPL	
Experiment duration	30 minutes	

Simulated Network The simulated network consists of three Cooja motes (provided by Cooja simulator), two transmitters (the orange nodes 2, 3 in Figure 5.3) and one receiver (the green node 1 in Figure 5.3 which is a sink and an RPL border-router). The transmitter nodes act as UDP clients by sending data to the sink and also as MQTT clients publishing telemetry to mosquitto MQTT broker via the sink node, which are then forwarded to Eclipse Hono as explained previously. CSMA is employed on the MAC layer with an 8 packets queue buffer for transmitting packets. Three Data Send Intervals (DSIs) are considered in this scenario to evaluate the PDR prediction accuracy of the Petri-net model ; $100ms$, $200ms$ and $500ms$.

Real Testbed The real testbed consists of two OpenMote B motes (INDUSTRIAL-SHIELDS, p. d.), which is an ultra low-power wireless development board for IoT applications. It features the Texas Instrument's CC2538 ARM Cortex-M3 SoC (System on Chip) and supports the IPv6 stack through open source implementations such as Contiki-NG and OpenWSN. As depicted in Figure 5.3, two motes are deployed, one transmitter and one receiver (the sink) which is plugged to the computer where Eclipse Hono & Ditto are deployed, in order to forward telemetry to the NDT. Only one transmitter is deployed in this scenario due to the significant interference encountered when deploying two transmitters, which hindered their ability to synchronize the NDT. Two DSIs are considered to evaluate the prediction accuracy of the Petri-net model in this case, $300ms$ and $500ms$. These are different from the simulation scenario because the real motes are not able to support lower DSIs ($200ms$, $100ms$). Moreover, TSCH is employed on the MAC layer with an 8 packets queue buffer. The reason behind using TSCH is to guarantee synchronization between the two motes for which it manages a globally-synchronized network

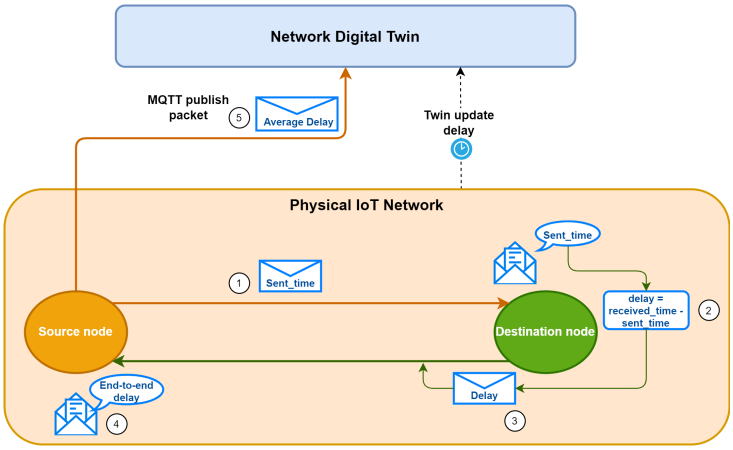


FIGURE 5.5 – Online end-to-end delay calculation and twin updating.

where the synchronization is coordinated by the RPL root towards the leaf nodes along the RPL DODAG. When a node receives a packet, it synchronizes itself relative to the time of its parent node. This synchronization is required for the online delay calculation, as opposed to the simulated network scenario where Cooja simulator takes care of the synchronization.

The synchronization rate of the NDT is determined by the MQTT publish period, which is set to 20 times the DSI in both scenarios. This choice is made to maintain energy efficiency in IIoT motes and to avoid disrupting network operations. In fact, if the MQTT publish period is excessively high, it may negatively impact network performance.

Figure 5.5 demonstrates the online end-to-end delay calculation method. A transmitter/source node timestamps UDP packets with the current time which is referred to as the sent time. Upon receiving a packet, the sink node retrieves the sent time value and computes the delay as the difference between the reception time and the retrieved value. Then, the sink node responds to the transmitter node with a packet containing the calculated delay in its payload. Finally, the source node publishes the average delay value to the NDT after the aggregation process. We aggregate the delay values within the publish period by computing their average value. The twin update delay, which is the time required for the MQTT publish packet containing the average delay to reach the NDT, is negligible in our scenario since the NDT synchronization rate equals 20 times the DSI. In fact, the synchronization interval equals $20 * 100ms = 2$ seconds for the worst case scenario when the DSI is set to $100ms$ (simulation), or $20 * 300ms = 6$ seconds when experimenting with the real testbed. However, in real-world scenarios, it is crucial to investigate the impact of this updating delay on the NDT synchronization. Additionally, the potential consequences of MQTT packet losses which could hinder successful updating of the NDT should also be explored. An analysis of the impact of MQTT packet losses on the system observability is conducted in Section 5.3.1.

The right side of Figure 5.3 depicts the TPN model that we implemented using SNAKES library. We adopted a producer-consumer model to reflect the IIoT with multiple sources/producers (place P1) and one destination/consumer (place P6). Transition t3 represents successful packet reception while t4 models packet loss. The parameters of the model include t1 delay which is the nodes' DSI, t2 delay which is the network end-to-end delay, t3 and t4 delays with null values and t5 delay which is between 0 and the nodes' DSI. Based on these parameters, we designed a "thing" structure (the IIoT mote digital counterpart) with two features, i.e., the DSI and the average delay of sending data to the sink. Hence, the IIoT motes publish these two values

periodically as telemetry in order to update their features in Eclipse Ditto.

The designed data-driven producer-consumer model is exploited to run what-if scenarios for PDR prediction in order to maintain high network reliability. In addition, it serves as a baseline model capturing the normal behavior of the network, thus, a fault is detected when a deviation within a certain threshold occurs between the actual PDR, experienced by the physical network, and the PDR predicted by the Petri-net model. Consequently, it is a "coarse-grained" model of the IIoT designed for fast PDR prediction and real-time fault detection.

To demonstrate the utility of the model in predicting the PDR, we design a scenario where the network end-to-end delay is learned over a 30-minutes period by the producer-consumer model. Then, the model is launched to predict the PDR with the same duration. In this case, the TPN model operates offline allowing it to run faster than the IIoT network, as it deals with time units and does not need to adhere to the real network's timing constraints. This results in a more efficient simulation and fast accommodation of the real network. Consequently, we run the IIoT network for 30 minutes for each DSI, incorporating a 5-minute network convergence period before initiating the transmission of UDP data packets. The MQTT connection process for the transmitter nodes is activated after 280 seconds have elapsed.

As for the faults detection functionality, the producer-consumer model operates online with the IIoT by retrieving continuously the average delay value from the NDT to parameterize transition t2. We design a scenario where the IIoT network operates initially with $500ms$ DSI before adding unexpected traffic with the goal of disturbing its normal operation and provoking a fault. The additional traffic is generated with a $333ms$ DSI, resulting in a global DSI of $200ms$.

5.3.1 Results and Discussion

Hereafter, the PDR prediction results are discussed for the different considered DSIs, firstly for the simulation scenario before moving to the real network. In this latter scenario, the real-time faults detection functionality is highlighted. Additionally, a study on the observability of the system is provided to highlight its importance in the context of NDTs.

PDR Prediction and Fault Detection

The model predicted PDR is calculated using the formula :

$$PredictedPDR = 1 - (\#\_firing\_times\_t4/\#\_firing\_times\_t1)$$

where $\#\_firing\_times\_t4$ is the number of firing times of transition t4 which models the number of lost packets while $\#\_firing\_times\_t1$ is the number of firing times of transition t1 which represents the number of sent packets.

Simulation As previously explained, the developed producer-consumer model parameters require the DSI and the network end-to-end delay. Table 5.2 presents the experienced end-to-end delay range of the simulated physical network for each DSI. Only the minimum and maximum values are considered because the intention is to fire the transition t2 randomly within the interval. A uniform distribution was chosen for its simplicity and tractability, serving as a starting point for this study. This choice allows us to focus on validating the overall approach, with the understanding that future work could explore alternative distributions to more accurately reflect real-world network behavior and further refine the model's accuracy and applicability. Figure 5.6 highlights the accuracy of our model PDR prediction where the IIoT PDR is compared to the PDR calculated by the Petri-net producer-consumer model for each DSI. The presented results

TABLE 5.2 – Experienced end-to-end delay range for each DSI - Simulation

DSI (ms)	Delay range (ms)
100	[19, 42]
200	[7, 71]
500	[8, 68]

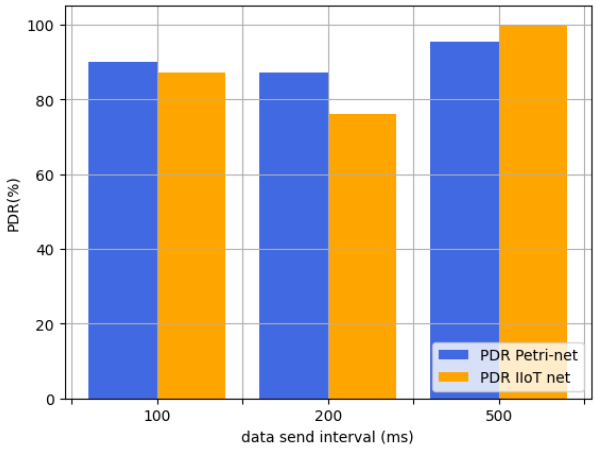


FIGURE 5.6 – Predicted PDR vs Real Network PDR (Simulated Network)

showcase the model utility as it is demonstrated by its ability to act as a lightweight model mirroring the IIoT behavior and make accurate predictions in terms of PDR. In fact, the model provides a platform to run what-if scenarios where the required parameters are retrieved from the NDT to anticipate network failure in terms of packet loss. If so, the NDT can send control commands to the IIoT network to maintain high reliability when the nodes DSI provokes many packet losses. More explicitely, the NDT can be configured with the application requirements in terms of PDR, i.e., what is the lower bound to qualify a PDR as "good" or "bad" for the application. By predicting the PDR based on the network operational parameters, the NDT can send commands to reduce the DSI in order to improve reliability when the predicted PDR is below the lower bound.

Real testbed The end-to-end delay ranges for the two considered DSIs in the real testbed scenario are presented in Table 5.3. These values are employed to parametrize transition t2 in the producer-consumer model, similarily to the simulation scenario. We experience higher latency for the $300ms$ DSI compared to the low DSIs in the simulation scenario ($100ms$, $200ms$). This is mainly due to the real world Wireless Sensor Networks problems that were not perfectly captured in the simulation model, such as multi-path fading, interference, etc. In fact, these problems invoke packet loss which increases the number of retransmissions of a packet, thus increasing the delay for a packet to reach its destination. Figure 5.7 presents the predicted PDR by the petri-net model against the real IIoT network PDR for both DSIs. We observe an accurate prediction of the petri-net model with very low deviation from the real IIoT network for both DSIs, which validates the results obtained by simulation.

Furthermore, the model can serve as a fault detection module by comparing the IIoT PDR with the model's predicted PDR in real-time (online). Figure 5.8 depicts the PDR curves of the IIoT network against the Petri-net model. We observe an accuracy between the Petri-net

TABLE 5.3 – Experienced end-to-end delay range for each DSI - Real Testbed

DSI (ms)	Delay range (ms)
300	[44, 1000]
500	[40, 76]

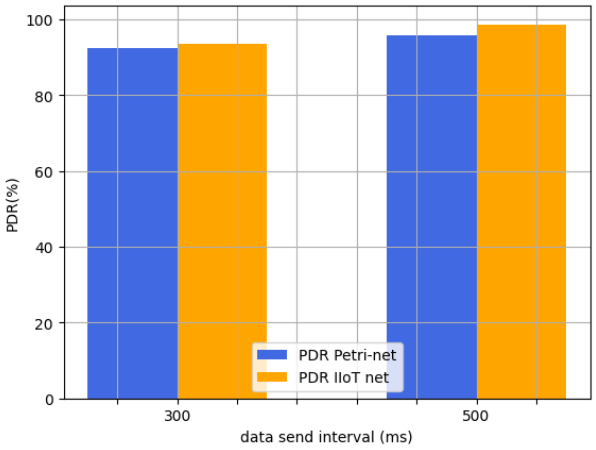


FIGURE 5.7 – Predicted PDR vs Real Network PDR (Real Testbed)

PDR and the IIoT PDR with an average deviation of 2% before the vertical dashed line, which represents the period where the IIoT network is running in normal conditions with a 500ms DSI. After that, the IIoT PDR drops until reaching 90% because we disturbed its normal operation by introducing additional traffic of 333ms DSI to provoke a fault, while the Petri-net PDR remains at 97%, resulting in a deviation of 7%. Thus, if considering for instance the fault detection threshold equals to 5%, one can alert the physical system about the potential fault. Knowing that the fault is generated in the IIoT network after 1141 seconds of operation and the intersection between the fault detection threshold tube and the PDR IIoT curve occurs at around 1450 seconds, the fault detection delay in this case equals 309 seconds.

System Observability

The observability of the physical network by the NDT depends on the volume of collected telemetry and its representativeness, the frequency of sending the twin updates defined as the synchronization rate, the reliability of the twin updates, among others. In this Section, a study on the impact of the twin updates reliability on the NDT's performance is conducted. We employ the MQTT publish messages PDR, which is the ratio of the number of received publish messages by the NDT to the number of messages sent by the physical IIoT network, as the metric to measure the twin updates reliability.

For this analysis, two scenarios have been designed by varying the number of the MAC layer retransmissions using CSMA in the IIoT. CSMA is employed as we are working with a network simulated through Cooja simulator, which handles the necessary synchronization for online delay calculation, eliminating the need for TSCH in this scenario. In the first configuration, no retransmissions are allowed (set to zero), while in the second configuration, up to two retransmissions are permitted. The objective is to evaluate how this parameter influences the MQTT publish PDR and demonstrate the direct correlation between this metric and the prediction accuracy of

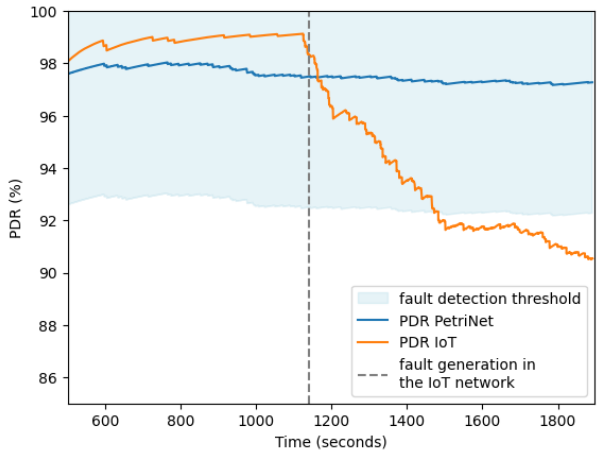
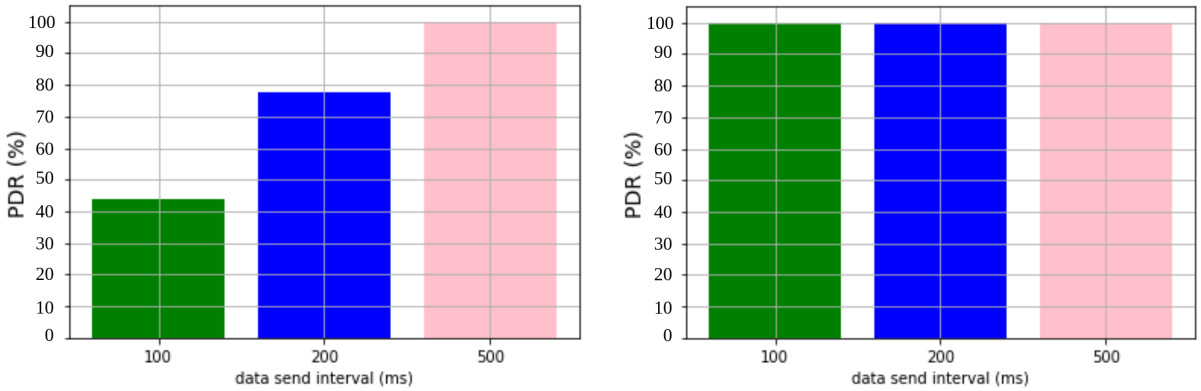


FIGURE 5.8 – Real-time faults detection



(a) When CSMA retries = 0.

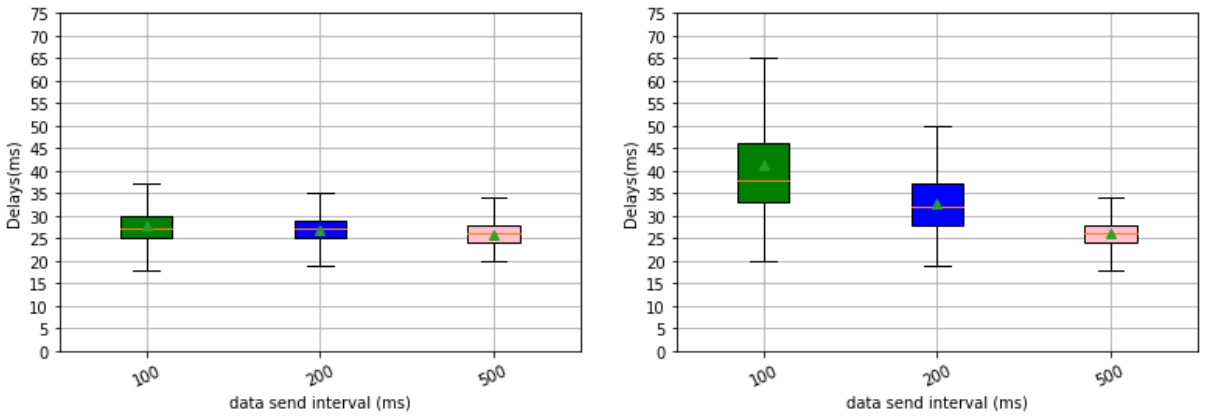
(b) When CSMA retries = 2.

FIGURE 5.9 – MQTT publish PDR.

the producer-consumer model.

Figure 5.9 depicts the MQTT publish PDR in both scenarios for the three considered DSIs ($100ms$, $200ms$ and $500ms$). With CSMA retries set to zero (Figure 5.9a), the MQTT publish PDR falls below 50% for the $100ms$ DSI, reaches up to 80% for the $200ms$ DSI, and attains 100% when the DSI is set to $500ms$. In fact, the IIoT network experiences high congestion in the first DSI, and since no retransmissions are permitted, the nodes are unable to guarantee successful MQTT publish message transmissions, leading to numerous packet losses. The IIoT network experiences lower congestion for the second DSI, resulting in fewer packet losses as compared to the first case. Lastly, the IIoT network is not congested for the $500ms$ DSI and therefore can ensure high reliability (100%). On the other hand, the MQTT publish PDR equals 100% for the three DSIs when CSMA retries equals two, as shown in Figure 5.9b. Since the Petri-net model requires the IIoT end-to-end delay to operate, we plot the published average-delay values in Figure 5.10 for both scenarios. The minimum and maximum values for each DSI are considered to parametrize transition t2 in the producer-consumer model.

Finally, the Petri-net predicted PDR is plotted against the physical IIoT network PDR in Figure 5.11 for both configurations. We observe a significant inaccuracy in PDR prediction for



(a) When CSMA retries = 0 (b) When CSMA retries = 2

FIGURE 5.10 – Published delay values.

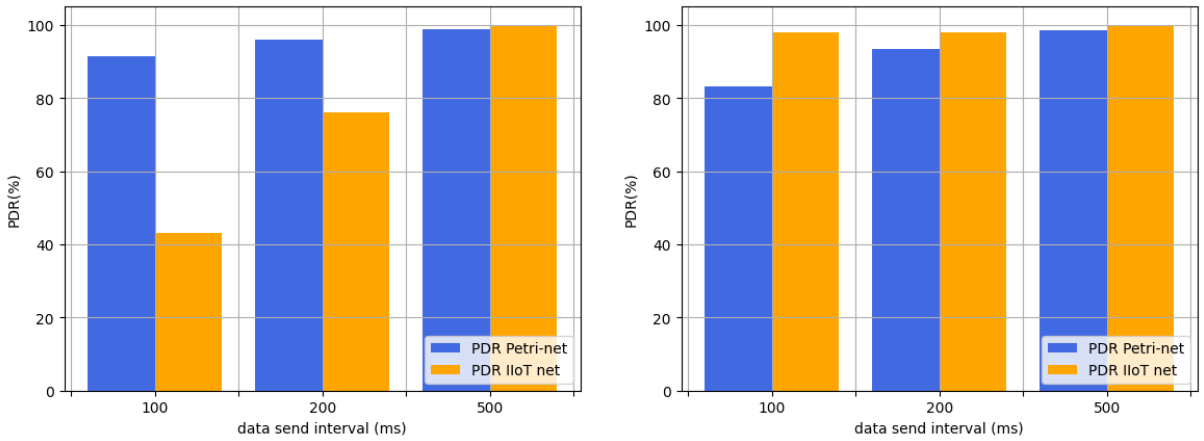
the first scenario (CSMA retries = 0) when the DSI is set to 100ms. In this case, the actual PDR of the IIoT network is 43%, while the predicted PDR value is 91%. This leads to a 53% deviation, primarily due to the high packet loss (57%) experienced with MQTT publish messages for this DSI, as shown in Figure 5.9a. Moreover, a 21% deviation is observed for the 200ms DSI, as there is less packet loss (23%) compared to the first DSI. Lastly, the Petri-net predicted PDR closely matches the IIoT network PDR for the 500ms DSI, as no packet loss is recorded in both scenarios.

For the second scenario, when the number of permitted CSMA retries is set to two, we observe a better prediction accuracy since no packet loss is experienced with the three considered DSIs. Although, the producer-consumer model does not perfectly mirror the IIoT network which is justified by the 15% deviation for the 100ms DSI and a 4.5% deviation for the 200ms DSI. A deviation threshold can be established to assess the prediction accuracy of the model. Based on the results of our study, a 20% threshold could be chosen. If the deviation is lower than this threshold, the model's accuracy can be considered satisfactory. Conversely, if the deviation exceeds the threshold, the model's accuracy may be deemed unsatisfactory.

The accuracy of the model serves as an indicator of the system observability. In this study, we have successfully demonstrated a strong correlation between the reliability of the twin update messages and the Petri-net model accuracy.

Limitations

Despite the promising results obtained with our approach, we only modeled a small-scale network using TPNs to validate our proposition. However, scaling to bigger networks remains an issue and brings several challenges. In fact, modeling large-scale IIoT scenarios using Petri-nets could be error prone and complex. As the IIoT network grows in terms of the number of devices, connections, and processes, the Petri-net representing the network can become exceedingly large. This can lead to an explosion in the number of places, transitions, and arcs, making the Petri-net cumbersome to work with both in terms of visualization and analysis. Given this complexity, it might be tempting to abstract certain details for simplicity. However, striking a balance between a high-level overview and the necessary granularity can be challenging. Too much abstraction might omit crucial details, while too little can make the model unwieldy.



(a) When CSMA retries = 0

(b) When CSMA retries = 2

FIGURE 5.11 – Predicted PDR vs Real Network PDR .

5.4 Conclusion and Perspectives

In this chapter, we have addressed the challenging problem of network data modeling for the NDT by utilizing Petri Nets as a modeling tool. We start by exploring the numerous advantages offered by this well-established technique when applied for such purpose, including offline applications (performance analysis, network planning, energy efficiency, etc.) and online applications (congestion control, traffic shaping and QoS, protocol verification, etc.). We introduce a data-driven Digital Twin architecture for the IIoT that incorporates Petri Nets. This proposed architecture is assessed through a case study of a small-scale network modeled using a light-weight coarse-grained TPN producer-consumer model, which involved both simulated and real IIoT testbed.

The results demonstrate that the model is capable of accurately predicting the PDR of the IIoT, making it a valuable tool for running what-if scenarios and maintaining high network reliability. Additionally, the model serves as a fault detection module, enabling prompt action to be taken in case of deviations between the model PDR and the real network PDR. The assessment of the system observability highlights the crucial role of reliable twin updates in influencing the NDT performance, as demonstrated by the accuracy of the PDR prediction in our model. By ensuring high reliability in twin updates, the physical system observability using the NDT is enhanced, as a larger volume of telemetry data is collected. This increased observability enables a more accurate prediction for our model. However, maintaining a perfectly reliable wireless communication in real-world scenarios is challenging, which can make it difficult to ensure good system observability. Therefore, Artificial Intelligence mechanisms can be employed to estimate the values of lost data based on previously collected information.

As future work, we plan to refine the TPN model to more accurately represent the networking behavior of the IIoT, taking into account factors such as message retransmissions in the event of packet loss and destination acknowledgments for received messages. Also, the network end-to-end delay distribution for transition t2 should be carefully investigated to more precisely mirror real-world network behavior, thereby enhancing the model's accuracy and broadening its applicability.

The content of this chapter is the outcome of two papers, the first is presented and published in the 6th IFAC Symposium on Telematics Applications (KHERBACHE, MAIMOUR et

Eric Rondeau, 2022a) and the second is published in Elsevier Internet of Things journal (Kherbache, Ahmed et al., 2023).

In the next chapter, we conclude this thesis and provide future perspectives.

Chapter 6

Conclusion and Perspectives

This thesis has addressed various challenges of the IIoT aiming to respond to its strict requirements. For this purpose, we propose to employ RL-based TSCH scheduling given its great adaptability inherited from its continuous observations and interactions within the network environment. Moreover, we propose the Network Digital Twin for the IIoT which brings several benefits to these networks ranging from their design to service phases.

In Chapter 3, we start by introducing a Q-learning distributed TSCH scheduling algorithm, QL-TSCH-plus, aiming to reduce the energy consumption of the well-established QL-TSCH algorithm (H. Park, Haeyong Kim, S.-T. Kim et al., 2020). Our methodology presents a distributed scheme for the Action Peeking mechanism while preserving the original reward function of QL-TSCH. Rather than incessantly monitoring the communications of neighboring nodes, a node now broadcasts the learned transmission slot to its neighbors, enabling the receivers to refresh their Action Peeking Tables for the communicated slot. Furthermore, this broadcast message also facilitates the allocation of Rx slots, ensuring their synchronization with the RPL routing tree. Moreover, we designed a scenario with highly heterogeneous traffic conditions to assess the performance of autonomous, distributed, and RL-based schedulers. This evaluation has allowed us to gain insights into the behavior of these three types of schedulers under heterogeneous traffic conditions, prominently present in Industry 4.0 settings. First, QL-TSCH-plus reduced considerably QL-TSCH energy consumption by up to 47% while maintaining network performance. Based on the conducted multi-criteria analysis where each metric is given a different weight based on its importance with regard to the application requirements, QL-TSCH-plus proved to be the most suitable scheduler in industrial settings where fulfilling real-time requirements is crucial. In contrast, autonomous and distributed schedulers like Orchestra and MSF performed better than RL-based ones when all metrics are given equal importance. This demonstrates their higher maturity and versatility to a wide range of applications compared to RL-based ones. These findings highlights the imperative for adopting more sophisticated scheduling solutions, particularly based on Deep RL. Although some Deep RL based TSCH schedulers are proposed in the literature, they are not yet integrated in a real network stack. In fact, the current networking architectures are not ready for adopting such complex technique in a real network stack. Consequently, we reorient our research towards Network Digital Twins for the IIoT to lay the groundwork for this transition.

In Chapter 4 we address the integration of SDN with Digital Twin for the IIoT. Particularly, we proposed a holistic NDT architecture for the IIoT to enable closed-loop network management across the entire network life-cycle (from design to service). This architecture enables quick validation of networking solution in an industrial environment because of the continuous link between

the NDT and the physical network. To validate the proposed architecture, we have leveraged it to satisfy the real-time requirements of an industrial project. This has allowed us to choose the best communication strategy for the considered Flexible Production System, which is to use TSCH with TASA without oversampling. Additionally, we have designed the NDT core architecture for the IIoT to describe its various internal components and their interactions. Various intelligent services are included in the proposed architecture such as predictive maintenance, network diagnosis, energy optimization, among others. A resource allocation service is also incorporated in the NDT to allow the integration of Deep RL TSCH scheduling in a real IIoT stack.

In Chapter 5, we have proposed a framework for modeling the IIoT using Petri-nets within the NDT environment. Mainly motivated by the computational cost of simulators in the NDT and the lack of formal rigor of existing network modeling techniques (GNNs, knowledge graphs). Petri-nets incorporation in the NDT enables data-driven Petri-nets having the capability to simulate simultaneously the state-driven behavior and the influence of real-time parameters on the physical network. These models serve as coarse-grained formal models enabling several benefits such as fast simulation time for what-if scenarios execution, and real-time fault detection that is crucial in mission-critical industrial applications, among others. We have proposed an open source NDT architecture for the IIoT based on Eclipse Hono and Ditto that incorporates Petri-nets to model the IIoT. We have assessed and validated the proposed architecture through a case study of a small-scale network modeled using Timed Petri-nets. This model is used offline to run what-if scenarios for PDR prediction based on the network operational parameters (Data Send Interval, end-to-end delay) aiming to maintain high network reliability. Moreover, when used online, it serves as a fault detection module leveraging its PDR prediction feature, enabling prompt action to be taken in case of deviations between the model PDR and the real network PDR.

Overall, the contributions of this thesis can be summarized in Figure 6.1. Chapter 3 mainly contributes to the physical IIoT network by proposing QL-TSCH-plus and evaluating decentralized TSCH schedulers under heterogeneous traffic conditions. Chapter 4 proposes a holistic digital twin networking architecture for the IIoT along with the NDT core architecture leveraging SDN. Chapter 5 incorporates Petri-nets to model the IIoT within the NDT. Deep RL scheduling implementation in the NDT is part of our perspectives that we discuss in the next section.

The contributions of this thesis can be extended in several directions, hereafter we present some of them.

Real Testbed Experiments. In this thesis, we have evaluated decentralized TSCH scheduling algorithms including autonomous, distributed and RL-based under heterogeneous traffic conditions in a unified simulation environment. As future work, we intend to conduct these experiments in FIT IoT-LAB (Adjih et al., 2015) to observe and experience real-world wireless networking conditions (interference, multi-path fading, etc). Additionally, we aim to conduct experiments in industrial testbeds, especially in environments characterized by various factors such as significant metallic reflective surfaces, dust, machinery noise, and elevated temperatures. Specifically, we plan to evaluate the performance of decentralized schedulers in these industrial test-beds, with a particular emphasis on our proposed algorithm, QL-TSCH-plus. Our aim is to confirm whether the conclusions we drew from simulation about its suitability for industrial settings hold true in real-world industrial scenarios.

Traffic-Adaptive RL-based TSCH Scheduling. Our proposed distributed Q-learning TSCH scheduler, QL-TSCH-plus, learns the best transmission slot that provides the best reward and minimizes collisions by choosing the least recently used slot. A notable limitation is its slot

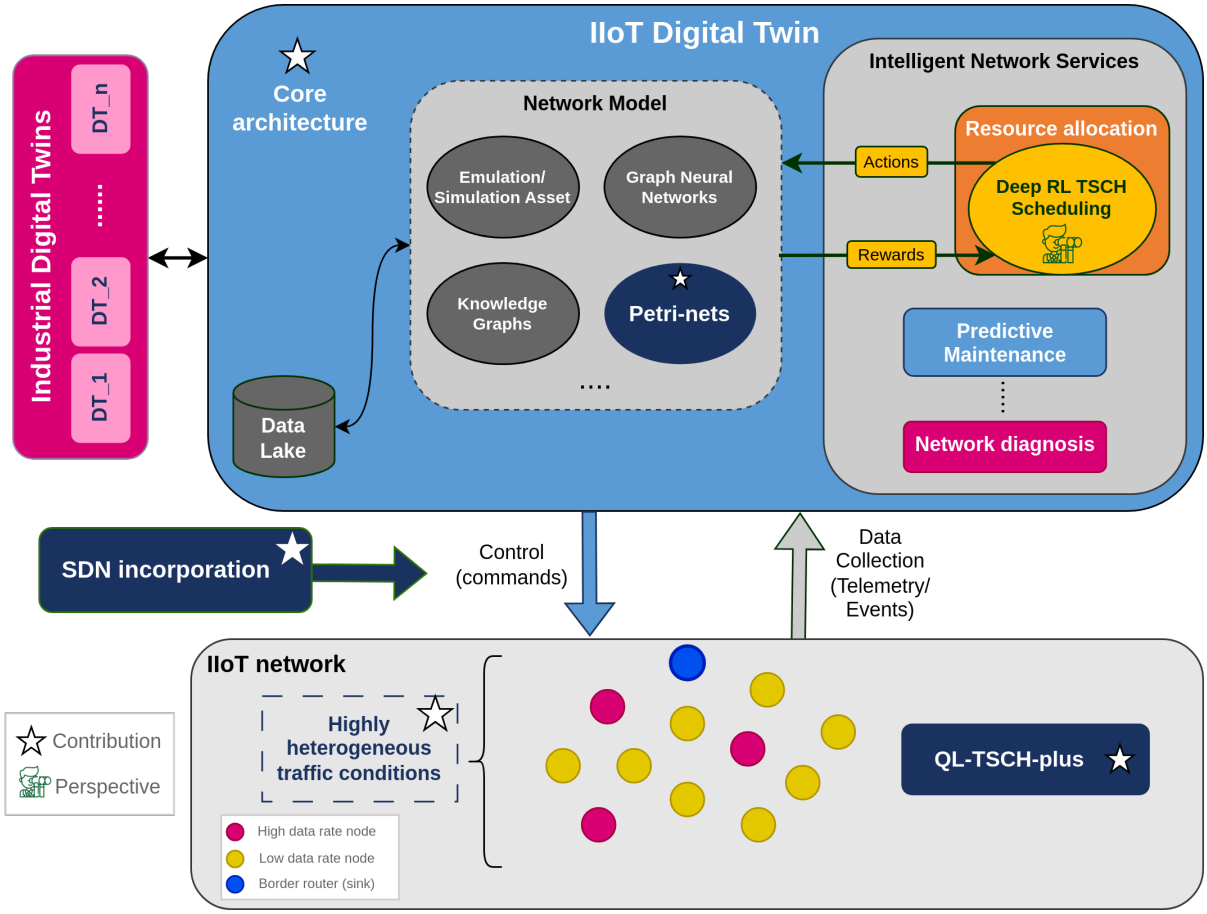


FIGURE 6.1 – Contributions summary

allocation strategy. Currently, regardless of the traffic intensity from a node, QL-TSCH-plus allocates only a single transmission slot. In the dynamic environment of IIoT, this might not be optimal as nodes might experience heterogeneous traffic loads. To address this, future iterations of QL-TSCH-plus aim to be traffic-adaptive. The vision is to create a smart scheduler that could discern the best transmission slot but to also dynamically determine the number of slots a node requires. By integrating a learning mechanism that factors in the node's traffic generation rate, the algorithm will dynamically allocate transmission slots ensuring efficient bandwidth utilization and further reducing the chances of network congestion.

Deep RL Scheduling Using the NDT. As we have already mentioned in the motivations of our thesis in Section 1.2, Deep RL can bring several benefits to TSCH scheduling. Mainly, it would enhance the scalability and responsiveness of centralized schedulers by providing a globally adaptive scheduler. In this thesis, we have prepared the ground for integrating such techniques in a real IIoT stack by proposing the NDT for the IIoT. The next steps consist in effectively implementing a Deep RL TSCH scheduler in the NDT which continuously learns the optimal schedule based on the current network state. In fact, the NDT provides a safe, flexible, and cost-effective environment for developing, training, and validating Deep RL TSCH scheduling algorithms. This is facilitated by its capacities for real-time network mirroring and advanced simulation. The scheduler actions would be assessed in the NDT via the interaction with the network model (Simulator, GNN, Petri-net) and receives rewards from it to learn the best scheduling policy. This way, the scheduler would progressively enhance its generalization capacity to a wide range of networking scenarios by exploiting not only the historical networking data collected by the NDT but also the latest network state. Nevertheless, the deployment of the Deep RL scheduling model in the NDT comes with the risk of network downtime in the event of NDT failure. Thus, decentralized schedulers can serve as "backup" algorithms that take over the scheduling responsibility in case of NDT failure. Moreover, a collaboration between decentralized RL-based and centralized Deep RL-based schedulers can further address this issue and provide a holistic TSCH scheduling solution fortifying the network's resilience against unforeseen complications.

NDT Collaboration with Industrial DTs. In modern industrial settings, various DTs coexist, each fulfilling a distinct purpose. For instance, there may be a DT dedicated to a piece of manufacturing equipment, another for managing the supply chain, and so forth. These DTs highly depend on reliable and efficient network communication for real-time twin updating, monitoring, and decision-making. The NDT, acting as a virtual mirror of the industrial network infrastructure, becomes instrumental in achieving this. By continuously collaborating with industrial DTs, the NDT can gain insights into their specific networking needs, such as bandwidth demands and real-time requirements. This way, the NDT can simulate and optimize network policies to ensure that each industrial DT requirements are satisfied. This harmonized collaboration means that as the requirements of the DTs evolve, the NDT can proactively adapt, ensuring a resilient and optimized network environment. In the future, we intend to ensure this tight collaboration between the different DTs including the NDT with the aim of achieving a better orchestration of the whole industrial system.

Petri-nets Scalability. While Petri-nets offer formal rigor and explainability, ensuring their scalability for large-scale networking scenarios in the NDT environment is a topic yet to be fully addressed. A promising direction lies in adopting more advanced types of Petri-nets, notably

Hierarchical Petri-nets and Colored Petri-nets. Hierarchical Petri-nets provide the ability to model systems at various levels of detail, introducing a modularity that is especially beneficial when considering vast, interconnected IIoT networks. On the other hand, Colored Petri-nets, with their capacity to represent multiple, similar processes with different parameters using a single model, add compactness and clarity to the modeling process. By integrating these advanced types of Petri-nets, we anticipate simplifying the representation of complex IIoT networks. This progression not only strengthens the model's generalization capability but also equips it to adapt seamlessly to a myriad of large-scale IIoT scenarios. However, a subsequent challenge emerges when mirroring large-scale IIoT networks using the NDT, which is to design effectively the data that should be transferred to the NDT necessary for populating these types of Petri-nets with the IIoT operational parameters.

Incorporating Digital Twins for Private 5G Networks - SNCF Use Case. Upon completing my thesis, I am going to take up a post-doctoral role at CRAN focusing on private 5G networks tailored to meet SNCF specific industrial needs. This effort aligns with the shared vision of France and Germany to advance Europe's position in 5G technology for a more efficient and sustainable industrial future. Collaborating on this endeavor are major partners including the aforementioned SNCF, CRAN, along with NOKIA and the FRAUNHOFER Institute. Industry's digital transformation holds the promise of leveraging Digital Twins to reduce production expenses via enhanced supervision and superior quality control measures. SNCF's railway technical centers exemplify this potential. They undertake routine maintenance tasks — such as readying trains for subsequent trips— and oversee comprehensive train overhauls halfway through their lifespan. Such responsibilities present vast challenges, from maintaining delivery schedules and quality checks to controlling costs and gauging risks. In this context, the use of DTs specifically hinges on two connectivity features : (1) network connectivity of the physical process assets with a high Quality of Experience (QoE) without connectivity loss, especially when mobile, i.e., during the movement of operators or M2M terminals, and (2) geolocation through accurate tracking of the position of indoor and outdoor assets. The combination of these two features is essential for supervising and automating production processes on the scale of an entire center (covering about 10 to 30 hectares of indoor and/or outdoor space), as well as for risk management when tasks involving mobile tools and train parts are under consideration. The emerging 5G network technology will be considered and utilized in the project to explore and demonstrate the potential of using a single telecommunications technology to meet all the industrial needs of SNCF's use cases with a higher satisfaction level. The project's aim is to define optimized scenarios for the application of the local 5G network and showcase their potential for supervision, teleoperation, and quality control management of train maintenance processes and analyze the possible extension to any industrial context.

List of Publications

The contributions of this thesis have been published or submitted in peer-reviewed international journals and conferences.

International Journals

— Mehdi Kherbache, Moufida Maimour et Eric Rondeau (2021).“When Digital Twin Meets Network Softwarization in the Industrial IoT : Real-Time Requirements Case Study”. In : Sensors 21.24. ISSN : 1424-8220. DOI : 10.3390/s21248194.

— Mehdi Kherbache, Otabek Sobirov, Moufida Maimour et Eric Rondeau (2023). “ Decentralized TSCH Scheduling Protocols and Heterogeneous Traffic : Overview and Performance Evaluation”. In : Internet of Things 22, p.100696. ISSN : 2542-6605. DOI : https ://doi.org/10.1016/j.iot.2023.100696.

— Mehdi Kherbache, Moufida Maimour, Eric Rondeau (2023). "Constructing a Network Digital Twin through Formal Modeling : Tackling the Virtual-Real Mapping Challenge in IIoT Networks". In : Internet of Things 24, p. 101000. ISSN : 2542-6605. DOI : https ://doi.org/10.1016/j.iot.2023.101000.

International Conferences

— Mehdi Kherbache, Otabek Sobirov, Moufida Maimour et Eric Rondeau (2022). “Reinforcement Learning TDMA-Based MAC Scheduling in the Industrial Internet of Things : A Survey”. In : IFAC-PapersOnLine 55.8. 6th IFAC Symposium on Telematics Applications TA2022, p.83-88. ISSN : 24058963. DOI : https ://doi.org/10.1016/j.ifacol.2022.08.014.

— Mehdi Kherbache, Moufida Maimour et Eric Rondeau (2022). “Digital Twin Network for the IIoT using Eclipse Ditto and Hono”. In : IFACPapersOnLine 55.8. 6th IFAC Symposium on Telematics Applications TA2022, p.37-42. ISSN : 2405-8963. DOI : https ://doi.org/10.1016/j.ifacol.2022.08.007.

— Mehdi Kherbache, Moufida Maimour et Eric Rondeau (2022).“Network Digital Twin for the Industrial Internet of Things”. In : 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), p.573-578. DOI : 10.1109/WoWMoM54355.2022.00089.

— Mehdi Kherbache, Moufida Maimour, Eric Rondeau. “An Advanced Assessment of Decentralized TSCH Schedulers : Unraveling the Implications of Heterogeneous Traffic Scenarios on Performance Efficacy”. 22nd IFAC World Congress, IFAC 2023, Jul 2023, Yokohama, Japan. ⟨hal-04201294⟩.

— Mehdi Kherbache, Moufida Maimour, Eric Rondeau (2023). “QL-TSCH-plus : A Q-learning Distributed Scheduling Algorithm for TSCH Networks”. In : 9th World Forum on Internet

of Things, WF-IoT 2023.

Bibliography

Accettura, Nicola et al. (2015a). "Decentralized Traffic Aware Scheduling in 6TiSCH Networks : Design and Experimental Evaluation". In : *IEEE Internet of Things Journal* 2.6, p. 455-470. doi : 10.1109/JIOT.2015.2476915.

— (2015b). "Decentralized Traffic Aware Scheduling in 6TiSCH Networks : Design and Experimental Evaluation". In : *IEEE Internet of Things Journal* 2.6, p. 455-470. doi : 10.1109/JIOT.2015.2476915.

Adjih, Cédric et al. (déc. 2015). "FIT IoT-LAB : A Large Scale Open Experimental IoT Testbed". In : Milan, Italy. url : https://hal.inria.fr/hal-01213938.

Aijaz, Adnan et Usman Raza (2017). "DeAMON : A Decentralized Adaptive Multi-Hop Scheduling Protocol for 6TiSCH Wireless Networks". In : *IEEE Sensors Journal* 17.20, p. 6825-6836. doi : 10.1109/JSEN.2017.2746183.

Åkerberg, Johan, Mikael Gidlund et Mats Björkman (2011). "Future research challenges in wireless sensor and actuator networks targeting industrial automation". In : *2011 9th IEEE International Conference on Industrial Informatics*. IEEE, p. 410-415.

Alabadi, M., A. Habbal et X. Wei (2022). "Industrial Internet of Things : Requirements, Architecture, Challenges, and Future Research Directions". In : *IEEE Access* 10, p. 66374-66400. doi : 10.1109/ACCESS.2022.3185049.

Almasan, P. et al. (2022). "Network Digital Twin : Context, Enabling Technologies, and Opportunities". In : *IEEE Communications Magazine* 60.11, p. 22-27. doi : 10.1109/MCOM.001.2200012.

Anadiotis, A.G. et al. (2019). "SD-WISE : A Software-Defined WIreless SEnsor network". In : *Computer Networks*.

Anastasi, Giuseppe, Marco Conti et Mario Di Francesco (2011). "A Comprehensive Analysis of the MAC Unreliability Problem in IEEE 802.15.4 Wireless Sensor Networks". In : *IEEE Transactions on Industrial Informatics* 7.1, p. 52-65. doi : 10.1109/TII.2010.2085440.

Aradi, Szilárd (2020). "Survey of deep reinforcement learning for motion planning of autonomous vehicles". In : *IEEE Transactions on Intelligent Transportation Systems* 23.2, p. 740-759.

Arrichiello, Vincenzo et Paola Gualeni (2020). "Systems engineering and digital twin : a vision for the future of cruise ships design, production and operations". In : *International Journal on Interactive Design and Manufacturing (IJIDeM)* 14.1, p. 115-122.

Arulkumaran, Kai et al. (août 2017). "A Brief Survey of Deep Reinforcement Learning". In : *IEEE Signal Processing Magazine* 34. doi : 10.1109/MSP.2017.2743240.

Authority, Danish Maritime (2018). *Digital Twins for Blue Denmark*. Available online : https://www.dma.dk/Documents/Publikationer/Digital%20Twin%20report%20for%20DMA.PDF (accessed on 2021-11-23).

Bao, Ganfeng, Fang Liu et Li Hong (2003). "Modeling and verification of TCP congestion control based on colored Petri nets". In : *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security*

and Assurance (Cat. No.03CH37483). T. 2, 1045-1050 vol.2. DOI : 10.1109/ICSMC.2003.1244550.

Barricelli, Barbara Rita, Elena Casiraghi et Daniela Fogli (2019). "A Survey on Digital Twin : Definitions, Characteristics, Applications, and Design Implications". In : *IEEE Access* 7, p. 167653-167671. DOI : 10.1109/ACCESS.2019.2953499.

Bellavista, Paolo et al. (2021). "Application-Driven Network-Aware Digital Twin Management in Industrial Edge Environments". In : *IEEE Transactions on Industrial Informatics* 17.11, p. 7791-7801. DOI : 10.1109/TII.2021.3067447.

Berge, N. et al. (1994). "Methodology for LAN modeling and analysis using Petri nets based models". In : *Proceedings of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, p. 335-342. DOI : 10.1109/MASCOT.1994.284402.

Billington, J., G.E Gallasch et B. Han (2004a). *A Coloured Petri net approach to protocol verification*. Springer.

— (2004b). *A Coloured Petri net approach to protocol verification*. Springer.

Bitam, M. et H. Alla (2005). "Performance evaluation of a TCP/IP transmission using hybrid Petri nets". In : *The 3rd ACS/IEEE International Conference on Computer Systems and Applications, 2005.* P. 54-. DOI : 10.1109/AICCSA.2005.1387048.

Bouabdallah, Nizar, Mario E. Rivero-Angeles et Bruno Sericola (2009). "Continuous Monitoring Using Event-Driven Reporting for Cluster-Based Wireless Sensor Networks". In : *IEEE Transactions on Vehicular Technology* 58.7, p. 3460-3479. DOI : 10.1109/TVT.2009.2015330.

Boubour, R. et al. (1997). "A Petri net approach to fault detection and diagnosis in distributed systems. I. Application to telecommunication networks, motivations, and modelling". In : *Proceedings of the 36th IEEE Conference on Decision and Control*. T. 1, 720-725 vol.1. DOI : 10.1109/CDC.1997.650720.

Brahimi, B., E. Rondeau et C. Aubrun (2008). "Integrated approach based on High Level Petri Nets for evaluating Networked Control Systems". In : *2008 16th Mediterranean Conference on Control and Automation*, p. 1118-1123. DOI : 10.1109/MED.2008.4602274.

Bruynseels, Koen, Filippo Santoni de Sio et Jeroen van den Hoven (2018). "Digital twins in health care : ethical implications of an emerging engineering paradigm". In : *Frontiers in genetics* 9, p. 31.

Chang, Tengfei, Mališa Vučinić, Xavier Vilajosana, Diego Dujovne et al. (mai 2020). "6TiSCH Minimal Scheduling Function : Performance Evaluation". In : *Internet Technology Letters* 3. DOI : 10.1002/itl2.170.

Chang, Tengfei, Mališa Vučinić, Xavier Vilajosana, Simon Duquennoy et al. (mai 2021). *6TiSCH Minimal Scheduling Function (MSF)*. RFC 9033. DOI : 10.17487/RFC9033. URL : https://www.rfc-editor.org/info/rfc9033.

Chen, Wei (2020). "Intelligent manufacturing production line data monitoring system for industrial internet of things". In : *Computer Communications* 151, p. 31-41. ISSN : 0140-3664. DOI : https://doi.org/10.1016/j.comcom.2019.12.035. URL : https://www.sciencedirect.com/science/article/pii/S0140366419315518.

Chilukuri, Shanti et Dirk Pesch (2021). "RECCE : Deep Reinforcement Learning for Joint Routing and Scheduling in Time-Constrained Wireless Networks". In : *IEEE Access* 9, p. 132053-132063. DOI : 10.1109/ACCESS.2021.3114967.

Chilukuri, Shanti, Guangyuan Piao et al. (2021). "Deadline-Aware TDMA Scheduling for Multihop Networks Using Reinforcement Learning". In : *2021 IFIP Networking Conference (IFIP Networking)*, p. 1-9. DOI : 10.23919/IFIPNetworking52078.2021.9472801.

Cunha, Catherine da et al. (2021). "Designing the Digital Twins of Reconfigurable Manufacturing Systems : application on a smart factory". In : *IFAC-PapersOnLine* 54.1. 17th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2021, p. 874-879. issn : 2405-8963. doi : https://doi.org/10.1016/j.ifacol.2021.08.103. url : https://www.sciencedirect.com/science/article/pii/S2405896321008521.

Dai, Yuanyang et al. (2020). "Digital twins driving model based on Petri net in industrial pipeline". In : *2020 International Conference on Artificial Intelligence and Electromechanical Automation (AIEA)*. IEEE, p. 283-291.

Dam, Tijs van et Koen Langendoen (2003). "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks". In : *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*. SenSys '03. Los Angeles, California, USA : Association for Computing Machinery, p. 171-180. isbn : 1581137079. doi : 10.1145/958491.958512. url : https://doi.org/10.1145/958491.958512.

Dâmaso, Antônio, Nelson Rosa et Paulo Maciel (2014). "Using coloured petri nets for evaluating the power consumption of wireless sensor networks". In : *International Journal of Distributed Sensor Networks* 10.6, p. 423537.

De Guglielmo, Domenico, Simone Brienza et Giuseppe Anastasi (2016). "IEEE 802.15.4e : A survey". In : *Computer Communications* 88, p. 1-24. issn : 0140-3664. doi : https://doi.org/10.1016/j.comcom.2016.05.004. url : https://www.sciencedirect.com/science/article/pii/S0140366416301980.

Diallo, Ousmane, Joel JPC Rodrigues et Mbaye Sene (2015). "Performances evaluation and Petri nets". In : *Modeling and Simulation of Computer Networks and Systems*. Elsevier, p. 313-355.

Ding, Ye, Shi Gang et Jiang Hong (2015). "The Design of Home Monitoring System by Remote Mobile Medical". In : *2015 7th International Conference on Information Technology in Medicine and Education (ITME)*, p. 278-281. doi : 10.1109/ITME.2015.168.

Distefano, Salvatore, Francesco Longo et Marco Scarpa (2015). "QoS assessment of mobile crowdsensing services". In : *Journal of Grid computing* 13, p. 629-650.

Domenico De Guglielmo, Giuseppe Anastasi et Alessio Seghetti (2014). "From IEEE 802.15.4 to IEEE 802.15.4e : A Step Towards the Internet of Things". In : *Advances in Intelligent Systems and Computing*.

Dunkels, A., B. Gronvall et T. Voigt (2004). "Contiki - a lightweight and flexible operating system for tiny networked sensors". In : *29th Annual IEEE International Conference on Local Computer Networks*, p. 455-462. doi : 10.1109/LCN.2004.38.

Dunkels, Adam (mai 2012). "The ContikiMAC radio duty cycling protocol". In :

Duquennoy, Simon et al. (2015). "Orchestra : Robust Mesh Networks Through Autonomously Scheduled TSCH". In : *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. SenSys '15. Seoul, South Korea : Association for Computing Machinery, p. 337-350. isbn : 9781450336314. doi : 10.1145/2809695.2809714. url : https://doi.org/10.1145/2809695.2809714.

Eclipse (2017a). *Connect, Command & Control IoT devices*. en-us. [Online ; Accessed 2022-03-29]. url : https://www.eclipse.org/hono/.

— (2017b). *Eclipse Ditto™ • open source framework for digital twins in the IoT*. [Online ; Accessed 2022-03-15]. url : https://www.eclipse.org/ditto/.

Elsts, Atis et al. (2020). "An Empirical Survey of Autonomous Scheduling Methods for TSCH". In : *IEEE Access* 8, p. 67147-67165. doi : 10.1109/ACCESS.2020.2980119.

FAFOUTIS, Xenofon et al. (2016). "A residential maintenance-free long-term activity monitoring system for healthcare applications". In : *EURASIP Journal on Wireless Communications and Networking* 2016.1, p. 1-20.

FALCONER, David D, Fumiyuki ADACHI et Bjorn GUDMUNDSON (1995). "Time division multiple access methods for wireless personal communications". In : *IEEE Communications Magazine* 33.1, p. 50-57.

FAN, Chao et al. (2021). "Disaster City Digital Twin : A vision for integrating artificial and human intelligence for disaster management". In : *International Journal of Information Management* 56, p. 102049. ISSN : 0268-4012. DOI : https://doi.org/10.1016/j.ijinfomgt.2019.102049. URL : https://www.sciencedirect.com/science/article/pii/S0268401219302956.

FERRIOL-GALMÉS, Miquel et al. (2022). "RouteNet-Fermi : Network Modeling with Graph Neural Networks". In : *arXiv preprint arXiv :2212.12070*.

FOURTY, Nicolas, Adrien VAN DEN BOSSCHE et Thierry VAL (2012). "An advanced study of energy consumption in an IEEE 802.15.4 based network : Everything but the truth on 802.15.4 node lifetime". In : *Computer Communications* 35.14. Special issue : Wireless Green Communications and Networking, p. 1759-1767. ISSN : 0140-3664. DOI : https://doi.org/10.1016/j.comcom.2012.05.008. URL : https://www.sciencedirect.com/science/article/pii/S0140366412001703.

GALLUCCIO, Laura et al. (2015). "SDN-WISE : Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks". In : *2015 IEEE conference on computer communications (INFOCOM)*. IEEE, p. 513-521.

GARTNER (2023). *Emerging Technologies : Revenue Opportunity Projection of Digital Twins*. URL : https://www.gartner.com/en/documents/4011590.

GHORY, Imran (2004). "Reinforcement learning in board games". In : *Department of Computer Science, University of Bristol, Tech. Rep* 105.

GIUA, Alessandro et Manuel SILVA (2018). "Petri nets and Automatic Control : A historical perspective". In : *Annual Reviews in Control* 45, p. 223-239. ISSN : 1367-5788. DOI : https://doi.org/10.1016/j.arcontrol.2018.04.006. URL : https://www.sciencedirect.com/science/article/pii/S1367578818300117.

GLAESSGEN, Edward et David STARGEL (p. d.). "The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles". In : *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. DOI : 10.2514/6.2012-1818. eprint : https://arc.aiaa.org/doi/pdf/10.2514/6.2012-1818. URL : https://arc.aiaa.org/doi/abs/10.2514/6.2012-1818.

GNAWALI, Omprakash et P LEVIS (sept. 2012). *The Minimum Rank with Hysteresis Objective Function*. RFC 6719. DOI : 10.17487/RFC6719. URL : https://www.rfc-editor.org/info/rfc6719.

GRIEVES, Michael (2014). "Digital twin : manufacturing excellence through virtual factory replication". In : *White paper* 1, p. 1-7.

HAFIDI, Samy et Nawel GHARBI (2019). "Colored Petri Nets for Modeling Congestion Control in Wireless Sensor Networks with Retrials". In : *2019 IEEE Symposium on Computers and Communications (ISCC)*, p. 998-1003. DOI : 10.1109/ISCC47284.2019.8969705.

HE, Yuan, Junchen GUO et Xiaolong ZHENG (2018). "From surveillance to digital twin : Challenges and recent advances of signal processing for Industrial Internet of Things". In : *IEEE Signal Processing Magazine* 35.5, p. 120-129.

HEIDRICH-MEISNER, Verena et al. (jan. 2007). "Reinforcement learning in a Nutshell". In : p. 277-288.

Henry, Matthew H., Ryan M. Layer et David R. Zaret (2010). "Coupled Petri nets for computer network risk analysis". In : *International Journal of Critical Infrastructure Protection* 3.2, p. 67-75. issn : 1874-5482. doi : https://doi.org/10.1016/j.ijcip.2010.05.002. url : https://www.sciencedirect.com/science/article/pii/S1874548210000211.

Hoi-Sheung et al. (2007). "McMAC : A Parallel Rendezvous Multi-Channel MAC Protocol". In : *2007 IEEE Wireless Communications and Networking Conference*, p. 334-339. doi : 10.1109/WCNC.2007.67.

Hong, H. et al. (2021). "NetGraph : An Intelligent Operated Digital Twin Platform for Data Center Networks". In : *Proceedings of the ACM SIGCOMM 2021 Workshop on Network-Application Integration*, p. 26-32.

Hui, Linbo et al. (2022). "Digital Twin for Networking : A Data-driven Performance Modeling Perspective". In : *IEEE Network*, p. 1-8. doi : 10.1109/MNET.119.2200080.

IEEE (2002). "IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN - Specific Requirements - Part 15 : Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)". In : *IEEE Std 802.15.1-2002*, p. 1-473. doi : 10.1109/IEEESTD.2002.93621.

— (déc. 2016). *IEEE SA - IEEE Standard for Information Technology–Telecommunications and information exchange between systems local and metropolitan area networks–specific requirements - part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*. url : https://standards.ieee.org/ieee/802.11/5536/.

— (déc. 2020). *IEEE Standard for Low-Rate Wireless Personal Area Networks (LR-WPANs) : Amendment 2 : Physical Layer and Medium Access Control for Visible Light Communication*. Rapp. tech. 802.15.4-2020. IEEE Standards Association. url : https://standards.ieee.org/ieee/802.15.4/7029/.

IEEE802.15WG (2011). "IEEE standard for local and metropolitan area networks—Part 15.4 : Low-rate wireless personal area networks (lr-wpans)". In : *IEEE Std* 802, p. 4-2011.

Industrial-Shields (p. d.). *Openmote home page - OpenMote*. url : https://openmote.com/wp-content/uploads/2021/06/content.pdf.

Insights, Fortune Business (2023). *Internet of Things [IoT] Market Size, Share & COVID-19 Impact Analysis, By Component (Platform, Solution & Services), By End-use Industry (BFSI, Retail, Government, Healthcare, Manufacturing, Agriculture, Sustainable Energy, Transportation, IT & Telecom, and Others), and Regional Forecast, 2023-2030*. https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307. [Accessed 26-07-2023].

ISA (2011). *ANSI/ISA-100.11a-2011 Wireless systems for industrial automation : Process control and related applications — isa.org*. https://www.isa.org/products/ansi-isa-100-11a-2011-wireless-systems-for-industr. [Accessed 28-07-2023].

ITU (juin 2012). url : https://www.itu.int/rec/T-REC-Y.2060-201206-I.

Jeong, Seungbeom et al. (2020). "OST : On-Demand TSCH Scheduling with Traffic-Awareness". In : *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, p. 69-78. doi : 10.1109/INFOCOM41043.2020.9155496.

Jin, Yichao et al. (2016). "A centralized scheduling algorithm for IEEE 802.15. 4e TSCH based industrial low power wireless networks". In : *2016 IEEE Wireless Communications and Networking Conference*. IEEE, p. 1-6.

Jindal, Vandana (2018). "History and architecture of Wireless sensor networks for ubiquitous computing". In : *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 7.2, p. 214-217.

KANDRIS, Dionisis et al. (2020). "Applications of Wireless Sensor Networks : An Up-to-Date Survey". In : *Applied System Innovation* 3.1. ISSN : 2571-5577. DOI : 10.3390/asi3010014. URL : https://www.mdpi.com/2571-5577/3/1/14.

KARAAGAC, Abdulkadir, Ingrid MOERMAN et Jeroen HOEBEKE (2018). "Hybrid schedule management in 6TiSCH networks : The coexistence of determinism and flexibility". In : *IEEE Access* 6, p. 33941-33952.

KHAN, I.H. et M. JAVAID (2022). "Role of Internet of Things (IoT) in Adoption of Industry 4.0". In : *Journal of Industrial Integration and Management* 07.04, p. 515-533. DOI : 10.1142/S2424862221500068. eprint : https://doi.org/10.1142/S2424862221500068. URL : https://doi.org/10.1142/S2424862221500068.

KHAN, Md Al-Masrur et al. (2020). "A systematic review on reinforcement learning-based robotics within the last decade". In : *IEEE Access* 8, p. 176598-176623.

KHANAFER, Mounib, Mouhcine GUENNOUN et Hussein T. MOUFTAH (2014). "A Survey of Beacon-Enabled IEEE 802.15.4 MAC Protocols in Wireless Sensor Networks". In : *IEEE Communications Surveys & Tutorials* 16.2, p. 856-876. DOI : 10.1109/SURV.2013.112613.00094.

KHERBACHE, Mehdi, Arsalan AHMED et al. (2023). "Constructing a Network Digital Twin through formal modeling : Tackling the virtual–real mapping challenge in IIoT networks". In : *Internet of Things* 24, p. 101000. ISSN : 2542-6605. DOI : https://doi.org/10.1016/j.iot.2023.101000. URL : https://www.sciencedirect.com/science/article/pii/S2542660523003232.

KHERBACHE, Mehdi, Moufida MAIMOUR et Eric RONDEAU (2021). "When Digital Twin Meets Network Softwarization in the Industrial IoT : Real-Time Requirements Case Study". In : *Sensors* 21.24. ISSN : 1424-8220. DOI : 10.3390/s21248194. URL : https://www.mdpi.com/1424-8220/21/24/8194.

— (2022a). "Digital Twin Network for the IIoT using Eclipse Ditto and Hono". In : *IFAC-PapersOnLine* 55.8. 6th IFAC Symposium on Telematics Applications TA 2022, p. 37-42. ISSN : 2405-8963. DOI : https://doi.org/10.1016/j.ifacol.2022.08.007. URL : https://www.sciencedirect.com/science/article/pii/S2405896322010801.

— (2022b). "Network Digital Twin for the Industrial Internet of Things". In : *2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, p. 573-578. DOI : 10.1109/WoWMoM54355.2022.00089.

— (2023). "QL-TSCH-plus : A Q-learning distributed scheduling algorithm for TSCH networks". In : *9th World Forum on Internet of Things, WFIoT2023*.

KHERBACHE, Mehdi, Otabek SOBIROV et al. (2022). "Reinforcement Learning TDMA-Based MAC Scheduling in the Industrial Internet of Things : A Survey". In : *IFAC-PapersOnLine* 55.8. 6th IFAC Symposium on Telematics Applications TA 2022, p. 83-88. ISSN : 2405-8963. DOI : https://doi.org/10.1016/j.ifacol.2022.08.014. URL : https://www.sciencedirect.com/science/article/pii/S2405896322010874.

— (2023). "Decentralized TSCH scheduling protocols and heterogeneous traffic : Overview and performance evaluation". In : *Internet of Things* 22, p. 100696. ISSN : 2542-6605. DOI : https://doi.org/10.1016/j.iot.2023.100696. URL : https://www.sciencedirect.com/science/article/pii/S2542660523000197.

KIM, Hyojoon et Nick FEAMSTER (2013). "Improving network management with software defined networking". In : *IEEE Communications Magazine* 51.2, p. 114-119. DOI : 10.1109/MCOM.2013.6461195.

KIM, Nae-Soo, Kyeseon LEE et Jae-Hong RYU (2015). "Study on IoT based wild vegetation community ecological monitoring system". In : *2015 Seventh International Conference on Ubiquitous and Future Networks*, p. 311-316. DOI : 10.1109/ICUFN.2015.7182556.

Kim, Seohyang, Hyung-Sin Kim et Chongkwon Kim (2019). "ALICE : Autonomous Link-based Cell Scheduling for TSCH". In : *2019 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, p. 121-132. DOI : 10.1145/3302506.3310394.

Kiviharju, Mikko, Teijo Venäläinen et Suna Kinnunen (2009). "Towards Modelling Information Security with Key-Challenge Petri Nets". In : *Identity and Privacy in the Internet Age*. Sous la dir. d'Audun Jøsang, Torleiv Maseng et Svein Johan Knapskog. Berlin, Heidelberg : Springer Berlin Heidelberg, p. 190-206.

Kopetz, Hermann et Wilfried Steiner (2022). "Internet of Things". In : *Real-Time Systems : Design Principles for Distributed Embedded Applications*. Cham : Springer International Publishing, p. 325-341. DOI : 10.1007/978-3-031-11992-7_13. URL : https://doi.org/10.1007/978-3-031-11992-7_13.

Koubaa, A., M. Alves et E. Tovar (2006). "A comprehensive simulation study of slotted CSMA/CA for IEEE 802.15.4 wireless sensor networks". In : *2006 IEEE International Workshop on Factory Communication Systems*, p. 183-192. DOI : 10.1109/WFCS.2006.1704149.

Kouvatsos, Demetres D (2013). *Performance modelling and evaluation of ATM networks*. Springer.

Kreutz, D. et al. (2015). "Software-Defined Networking : A Comprehensive Survey". In : *Proceedings of the IEEE*.

Kurunathan, Harrison et al. (2018). "IEEE 802.15.4e in a Nutshell : Survey and Performance Evaluation". In : *IEEE Communications Surveys Tutorials* 20.3, p. 1989-2010. DOI : 10.1109/COMST.2018.2800898.

Lample, Guillaume et Devendra Singh Chaplot (2017). "Playing FPS games with deep reinforcement learning". In : *Proceedings of the AAAI Conference on Artificial Intelligence*. T. 31. 1.

Langendoen, Koen, Aline Baggio et Otto Visser (2006). "Murphy loves potatoes : Experiences from a pilot sensor network deployment in precision agriculture". In : *Proceedings 20th IEEE international parallel & distributed processing symposium*. IEEE, 8-pp.

Li, Hongcheng et al. (2022). "Data-driven hybrid petri-net based energy consumption behaviour modelling for digital twin of energy-efficient manufacturing system". In : *Energy* 239, p. 122178. ISSN : 0360-5442. DOI : https://doi.org/10.1016/j.energy.2021.122178. URL : https://www.sciencedirect.com/science/article/pii/S0360544221024269.

Li, Hua et al. (2015). "Development of a Remote Monitoring System for Henhouse Environment Based on IoT Technology". In : *Future Internet* 7.3, p. 329-341. ISSN : 1999-5903. DOI : 10.3390/fi7030329. URL : https://www.mdpi.com/1999-5903/7/3/329.

Light, Roger A (2017). "Mosquitto : server and client implementation of the MQTT protocol". In : *Journal of Open Source Software* 2.13, p. 265.

Lingling, Hu et al. (2011). "An Intelligent Vehicle Monitoring System Based on Internet of Things". In : *2011 Seventh International Conference on Computational Intelligence and Security*, p. 231-233. DOI : 10.1109/CIS.2011.59.

Liu, Zhenzhen et I. Elhanany (2006). "RL-MAC : A QoS-Aware Reinforcement Learning based MAC Protocol for Wireless Sensor Networks". In : *2006 IEEE International Conference on Networking, Sensing and Control*, p. 768-773. DOI : 10.1109/ICNSC.2006.1673243.

Ma, Jiali et al. (2022). "Digital Twin-Based Zero-Touch Management for IoT". In : *Electronics* 11.24, p. 4104.

Mainetti, Luca, Luigi Patrono et Antonio Vilei (2011). "Evolution of wireless sensor networks towards the Internet of Things : A survey". In : *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*, p. 1-6.

Mandolla, Claudio et al. (2019). "Building a digital twin for additive manufacturing through the exploitation of blockchain : A case analysis of the aircraft industry". In : *Computers in Industry* 109, p. 134-152.

McKeown, Nick et al. (mar. 2008). "OpenFlow : Enabling Innovation in Campus Networks". In : *SIGCOMM Comput. Commun. Rev.* 38.2, p. 69-74. ISSN : 0146-4833. DOI : 10.1145/1355734.1355746. URL : https://doi.org/10.1145/1355734.1355746.

Mihaylov, Mihail et al. (2011). "Distributed Cooperation in Wireless Sensor Networks". In : *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*. AAMAS '11. Taipei, Taiwan : International Foundation for Autonomous Agents et Multiagent Systems, p. 249-256. ISBN : 0982657153.

Mihaylov, Mihail Emilov et al. (2011). "Self-Organizing Synchronicity and Desynchronicity using Reinforcement Learning". English. In : *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*. Sous la dir. de Joaquim Filipe et Ana Fred. T. 2. Proceedings of the 3rd International Conference on Agents and Artificial Intelligence 2. Joaquim Filipe and Ana Fred, p. 94-103. ISBN : 978-989-8425-41-6.

Mitchell, Melanie (2009). *Complexity : A guided tour*. Oxford University Press.

Neto, José Maurıcio et al. (2017). "Modeling CSMA-CA protocol with coloured petri nets for wireless sensor networks applications". In : *17 International conference on software telecommunications and computer networks (SofCOM)*.

Nguyen-Duy, Hung et al. (2019). "RL-TSCH : A Reinforcement Learning Algorithm for Radio Scheduling in TSCH 802.15.4e". In : *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, p. 227-231. DOI : 10.1109/ICTC46691.2019.8939833.

Oikonomou, George et al. (2022). "The Contiki-NG open source operating system for next generation IoT devices". In : *SoftwareX* 18, p. 101089. ISSN : 2352-7110. DOI : https://doi.org/10.1016/j.softx.2022.101089.

Onasanya, Adeniyi et Maher Elshakankiri (2021). "Smart integrated IoT healthcare system for cancer care". In : *Wireless Networks* 27, p. 4297-4312.

Orfanidis, Charalampos et al. (2021). "TSCH Evaluation under Heterogeneous Mobile Scenarios". In : *IoT* 2.4, p. 656-668. ISSN : 2624-831X. DOI : 10.3390/iot2040033. URL : https://www.mdpi.com/2624-831X/2/4/33.

Orozco-Santos, Federico, Víctor Sempere-Payá et al. (2021). "Enhancing sdn wise with slicing over tsch". In : *Sensors* 21.4, p. 1075.

Orozco-Santos, Federico, Víctor Sempere-Payá et al. (2022). "TSCH Multiflow Scheduling with QoS Guarantees : A Comparison of SDN with Common Schedulers". In : *Applied Sciences* 12.1. ISSN : 2076-3417. DOI : 10.3390/app12010119. URL : https://www.mdpi.com/2076-3417/12/1/119.

Osterlind, Fredrik et al. (2006). "Cross-Level Sensor Network Simulation with COOJA". In : *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, p. 641-648. DOI : 10.1109/LCN.2006.322172.

Oztemel, Ercan et Samet Gursev (2020). "Literature review of Industry 4.0 and related technologies". In : *Journal of intelligent manufacturing* 31, p. 127-182.

Palattella, Maria Rita, Nicola Accettura et al. (2012). "Traffic Aware Scheduling Algorithm for reliable low-power multi-hop IEEE 802.15.4e networks". In : *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, p. 327-332. DOI : 10.1109/PIMRC.2012.6362805.

PALATTELLA, Maria Rita, Thomas WATTEYNE et al. (2016). "On-the-Fly Bandwidth Reservation for 6TiSCH Wireless Industrial Networks". In : *IEEE Sensors Journal* 16.2, p. 550-560. DOI : 10.1109/JSEN.2015.2480886.

PARK, Huiung, Haeyong KIM, Kyeong Tae KIM et al. (2019). "Frame-Type-Aware Static Time Slotted Channel Hopping Scheduling Scheme for Large-Scale Smart Metering Networks". In : *IEEE Access* 7, p. 2200-2209. DOI : 10.1109/ACCESS.2018.2886375.

PARK, Huiung, Haeyong KIM, Seon-Tae KIM et al. (juil. 2020). "Multi-Agent Reinforcement-Learning-Based Time-Slotted Channel Hopping Medium Access Control Scheduling Scheme". In : *IEEE Access* PP, p. 1-1. DOI : 10.1109/ACCESS.2020.3010575.

PARK, Kyu Tae et al. (avr. 2019). "Design and implementation of a digital twin application for a connected micro smart factory". In : *International Journal of Computer Integrated Manufacturing* 32, p. 1-19. DOI : 10.1080/0951192X.2019.1599439.

PHUNG, Kieu-Ha, Truong Thu HUONG et al. (2018). "A Scheduler for Time Slotted Channel Hopping Networks supporting QoS Differentiated Services". In : *2018 International Conference on Advanced Technologies for Communications (ATC)*, p. 232-236. DOI : 10.1109/ATC.2018.8587569.

PHUNG, Kieu-Ha, Bart LEMMENS et al. (jan. 2013). "Adaptive Learning Based Scheduling in Multichannel Protocol for Energy-Efficient Data-Gathering Wireless Sensor Networks". In : *International Journal of Distributed Sensor Networks* 2013. DOI : 10.1155/2013/345821.

POLLIN, Sofie et al. (2008). "Performance Analysis of Slotted Carrier Sense IEEE 802.15.4 Medium Access Layer". In : *IEEE Transactions on Wireless Communications* 7.9, p. 3359-3371. DOI : 10.1109/TWC.2008.060057.

POMMEREAU, F. (2015). "SNAKES : A flexible high-level petri nets library (tool paper)". In : *Application and Theory of Petri Nets and Concurrency : 36th International Conference, PETRI NETS 2015, Brussels, Belgium, June 21-26, 2015, Proceedings 36*. Springer, p. 254-265.

RIGHETTI, Francesca et al. (juil. 2020). "An Evaluation of the 6TiSCH Distributed Resource Management Mode". In : *ACM Trans. Internet Things* 1.4. ISSN : 2691-1914. DOI : 10.1145/3395927. URL : https://doi.org/10.1145/3395927.

RODRIGUEZ, Alejandro, Lars Michael KRISTENSEN et Adrian RUTLE (2018). "On Modelling and Validation of the MQTT IoT Protocol for M2M Communication." In : *PNSE@ Petri Nets/ACSD*, p. 99-118.

RODRÍGUEZ, Alejandro, Lars Michael KRISTENSEN et Adrian RUTLE (2019). "Formal modelling and incremental verification of the MQTT IoT protocol". In : *Transactions on Petri Nets and Other Models of Concurrency XIV*, p. 126-145.

ROSATI, Riccardo et al. (2023). "From knowledge-based to big data analytic model : a novel IoT and machine learning based decision support system for predictive maintenance in Industry 4.0". In : *Journal of Intelligent Manufacturing* 34.1, p. 107-121.

SAFARIC, Stanislav et Kresimir MALARIC (2006). "ZigBee wireless standard". In : *Proceedings ELMAR 2006*, p. 259-262. DOI : 10.1109/ELMAR.2006.329562.

SAHAR, Gul et al. (2021). "Recent Advancement of Data-Driven Models in Wireless Sensor Networks : A Survey". In : *Technologies* 9.4. ISSN : 2227-7080. DOI : 10.3390/technologies9040076. URL : https://www.mdpi.com/2227-7080/9/4/76.

SAVAGLIO, Claudio et al. (2019). "Lightweight Reinforcement Learning for Energy Efficient Communications in Wireless Sensor Networks". In : *IEEE Access* 7, p. 29355-29364. DOI : 10.1109/ACCESS.2019.2902371.

SHELBY, Zach, Klaus HARTKE et Carsten BORMANN (juin 2014). *The Constrained Application Protocol (CoAP)*. RFC 7252. DOI : 10.17487/RFC7252. URL : https://www.rfc-editor.org/info/rfc7252.

Sisinni, Emiliano et al. (2018). "Industrial internet of things : Challenges, opportunities, and directions". In : *IEEE transactions on industrial informatics* 14.11, p. 4724-4734.

Song, Jianping et al. (2008). "WirelessHART : Applying Wireless Technology in Real-Time Industrial Process Control". In : *2008 IEEE Real-Time and Embedded Technology and Applications Symposium*, p. 377-386. DOI : 10.1109/RTAS.2008.15.

Soua, Ridha, Pascale Minet et Erwan Livolant (2012). "MODESA : An optimized multichannel slot assignment for raw data convergecast in wireless sensor networks". In : *2012 IEEE 31st international performance computing and communications conference (IPCCC)*. IEEE, p. 91-100.

— (2014). "A Distributed Joint Channel and Slot Assignment for Convergecast in Wireless Sensor Networks". In : *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, p. 1-5. DOI : 10.1109/NTMS.2014.6813995.

— (2015). "DiSCA : A distributed scheduling for convergecast in multichannel wireless sensor networks". In : *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, p. 156-164. DOI : 10.1109/INM.2015.7140288.

Stark, R., S. Kind et S. Neumeyer (2017). "Innovations in digital modelling for next generation manufacturing system design". In : *CIRP Annals* 66.1, p. 169-172. ISSN : 0007-8506. DOI : https://doi.org/10.1016/j.cirp.2017.04.045. URL : https://www.sciencedirect.com/science/article/pii/S0007850617300458.

std., IEEE (avr. 2012). *802.15.4e, Part. 15.4 : Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendament 1 : MAC sublayer*. Standard. IEEE standard for Information Technology, p. 1-225. DOI : 10.1109/IEEESTD.2012.6185525.

Sutton, Richard S et Andrew G Barto (2018). *Reinforcement learning : An introduction*. MIT press.

Tan, Weng Chun et Manjit Singh Sidhu (2022). "Review of RFID and IoT integration in supply chain management". In : *Operations Research Perspectives* 9, p. 100229. ISSN : 2214-7160. DOI : https://doi.org/10.1016/j.orp.2022.100229. URL : https://www.sciencedirect.com/science/article/pii/S2214716022000070.

Tao, Fei et al. (2019). "Digital Twin in Industry : State-of-the-Art". In : *IEEE Transactions on Industrial Informatics* 15.4, p. 2405-2415. DOI : 10.1109/TII.2018.2873186.

Thubert, Pascal (mar. 2012). *Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)*. RFC 6552. DOI : 10.17487/RFC6552. URL : https://www.rfc-editor.org/info/rfc6552.

— (mai 2021). *An Architecture for IPv6 over the Time-Slotted Channel Hopping Mode of IEEE 802.15.4 (6TiSCH)*. RFC 9030. DOI : 10.17487/RFC9030. URL : https://www.rfc-editor.org/info/rfc9030.

Thubert, Pascal et Tim Winter (mai 2021). *RPL : IPv6 Routing Protocol for Low-Power and Lossy Networks*. Internet-Draft draft-pthubert-roll-rfc6550bis-01. Work in Progress. Internet Engineering Task Force. 154 p. URL : https://datatracker.ietf.org/doc/html/draft-pthubert-roll-rfc6550bis-01.

Tmote Sky : Datasheet (2006). Rapp. tech., p. 1-28.

Tolio, Tullio (2008). *Design of flexible production systems*. Springer.

Tsinarakis, George, Nikolaos Sarantinoudis et George Arampatzis (2022). "A Discrete Process Modelling and Simulation Methodology for Industrial Systems within the Concept of Digital Twins". In : *Applied Sciences* 12.2. ISSN : 2076-3417. DOI : 10.3390/app12020870. URL : https://www.mdpi.com/2076-3417/12/2/870.

Tsinarakis, George J. et al. (2020). "Implementation of a Petri-net based Digital Twin for the development procedure of an Electric Vehicle". In : *2020 28th Mediterranean Conference on Control and Automation (MED)*, p. 862-867. DOI : 10.1109/MED48518.2020.9182784.

Urke, Andreas Ramstad, Oivind Kure et Knut Ovsthus (2022). "A Survey of 802.15.4 TSCH Schedulers for a Standardized Industrial Internet of Things". In : *Sensors* 22.1. ISSN : 1424-8220. DOI : 10.3390/s22010015. URL : https://www.mdpi.com/1424-8220/22/1/15.

Veisi, Farzad, Julien Montavont et Fabrice Theoleyre (2023). "Enabling Centralized Scheduling Using Software Defined Networking in Industrial Wireless Sensor Networks". In : *IEEE Internet of Things Journal*, p. 1-1. DOI : 10.1109/JIOT.2023.3302994.

Vilajosana, Xavier, Kris Pister et Thomas Watteyne (mai 2017). *Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration*. RFC 8180. DOI : 10.17487/RFC8180. URL : https://www.rfc-editor.org/info/rfc8180.

Vitor, Rafael Ferreira et al. (2021). "Synchronous and Asynchronous Requirements for Digital Twins Applications in Industry 4.0." In : *ICEIS (2)*, p. 637-647.

Wang, Haozhe et al. (2022). "A Graph Neural Network-Based Digital Twin for Network Slicing Management". In : *IEEE Transactions on Industrial Informatics* 18.2, p. 1367-1376. DOI : 10.1109/TII.2020.3047843.

Wang, Qin, Xavier Vilajosana et Thomas Watteyne (nov. 2018). *6TiSCH Operation Sublayer (6top) Protocol (6P)*. RFC 8480. DOI : 10.17487/RFC8480. URL : https://www.rfc-editor.org/info/rfc8480.

Watkins, Christopher et Peter Dayan (mai 1992). "Technical Note : Q-Learning". In : *Machine Learning* 8, p. 279-292. DOI : 10.1007/BF00992698.

Wu, Jiaju et al. (2020). "The Development of Digital Twin Technology Review". In : *2020 Chinese Automation Congress (CAC)*, p. 4901-4906. DOI : 10.1109/CAC51589.2020.9327756.

Xu, Y. et X. Xie (2011). "Modeling and Analysis of Security Protocols Using Colored Petri Nets." In : *J. Comput.* 6.1, p. 19-27.

Yang, Wei et al. (2023). "Optimizing Federated Learning With Deep Reinforcement Learning for Digital Twin Empowered Industrial IoT". In : *IEEE Transactions on Industrial Informatics* 19.2, p. 1884-1893. DOI : 10.1109/TII.2022.3183465.

Ye, Jiuyan et al. (2013). "A precision agriculture management system based on Internet of Things and WebGIS". In : *2013 21st International Conference on Geoinformatics*, p. 1-5. DOI : 10.1109/Geoinformatics.2013.6626173.

Ye, Wei, J. Heidemann et D. Estrin (2004). "Medium access control with coordinated adaptive sleeping for wireless sensor networks". In : *IEEE/ACM Transactions on Networking* 12.3, p. 493-506. DOI : 10.1109/TNET.2004.828953.

Zhao, Liang et al. (2020). "Intelligent digital twin-based software-defined vehicular networks". In : *IEEE Network* 34.5, p. 178-184.

Zhou, Cheng et al. (avr. 2023). *Digital Twin Network : Concepts and Reference Architecture*. Internet-Draft draft-irtf-nmrg-network-digital-twin-arch-03. Work in Progress. Internet Engineering Task Force. 29 p. URL : https://datatracker.ietf.org/doc/draft-irtf-nmrg-network-digital-twin-arch/03/.

Zhu, Yanhong et al. (2021). "A knowledge graph based construction method for Digital Twin Network". In : *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*, p. 362-365. DOI : 10.1109/DTPI52967.2021.9540177.

Zia, Tanveer, Peng Liu et Weili Han (2017). "Application-Specific Digital Forensics Investigative Model in Internet of Things (IoT)". In : *Proceedings of the 12th International Conference on Availability, Reliability and Security*. ARES '17. Reggio Calabria, Italy : Association for

Computing Machinery. ISBN : 9781450352574. DOI : 10.1145/3098954.3104052. URL : https://doi.org/10.1145/3098954.3104052.

ZOUINKHI, A. et al. (2009). "Petri Nets Modelling of active products cooperation for active security management". In : *2009 6th International Multi-Conference on Systems, Signals and Devices*. IEEE, p. 1-6.

ZUBEREK, W.M. (1991). "Timed Petri nets definitions, properties, and applications". In : *Microelectronics Reliability* 31.4, p. 627-644. ISSN : 0026-2714. DOI : https://doi.org/10.1016/0026-2714(91)90007-T. URL : https://www.sciencedirect.com/science/article/pii/002627149190007T.