Horizon Europe Framework Programme

CENTRIC

Towards an AI-native, user-centric air interface for 6G networks

D4.1 Protocol emergence challenges

Contractual Delivery Date:	31st March 2024
Actual Delivery Date:	
Editor: *(name, organization)*	Alvaro Valcarce Rial, Nokia Networks France
Deliverable nature:	OTHER (Open-source software)
Dissemination level:	Public
Version:	1.0
Keywords: MAC, channel access, in-factory subnetworks, MARL, DCI	

ABSTRACT

This report describes four open-source software repositories developed as part of Work Package 4 of project CENTRIC and delivered as deliverable D4.1. These repositories address various challenges in protocol learning and emergence. Each repository focuses on a specific problem, such as multiple access with MuJoCo robots, random channel access with MARL, DCI learning for reducing the length of control messages, and 6G in-factory subnetworks for industrial applications. The report provides detailed descriptions of the purpose, background, features, functionality, and usage examples for each repository. These repositories offer a starting point for exploring alternative solutions, benchmarking, and customization, and can be integrated with machine learning algorithms for further research and development.

This project has received funding from the European Union's Smart Networks and Services Joint Undertaking (SNS JU) under grant agreement No. 101096379.

Disclaimer

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101096379. This publication reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Impressum

Full project title: Towards an AI-native, user-centric air interface for 6G networks

Short project title: CENTRIC

Number and title of the work package: WP4, AI-AI protocols and radio resource management

Number and title of task: Deliverable 4.1, Task 4.1

Document title: Protocol emergence challenges

Editor: Alvaro Valcarce Rial, Nokia Networks France

Work-package leader: Alvaro Valcarce Rial, Nokia Networks France

Executive summary

This report provides an overview of four open-source repositories developed as part of Work Package 4 of project CENTRIC. These repositories aim to address various challenges in protocol learning and emergence, offering valuable resources for researchers and developers in the field.

The first repository focuses on multiple access with MuJoCo robots. It provides a framework for simulating and studying the behavior of multiple robots in a shared environment. Researchers can use this repository to explore different strategies for coordinating the actions of multiple robots, enabling them to develop more efficient and robust protocols.

The second repository addresses random access with MARL (Multi-Agent Reinforcement Learning). It offers a platform for studying the behaviour of multiple agents in a random access scenario. By utilizing reinforcement learning techniques, researchers can investigate how agents can effectively share limited resources and optimize their access strategies.

The third repository focuses on DCI (Downlink Control Information) learning for reducing the length of 5GNR control messages. This repository provides a DCI simulator, that can be used to generate artificial DCI messages for protocol model training. By reducing the DCI length, researchers can improve the efficiency and scalability of communication protocols, leading to more efficient resource utilization.

The fourth repository explores 6G in-factory subnetworks for industrial applications. It offers a framework for simulating and analysing the performance of subnetworks in an industrial setting. Researchers can use this repository to evaluate the effectiveness of different communication protocols and network configurations, enabling them to design more reliable and efficient subnetworks for industrial applications.

Each repository in this report is accompanied by detailed descriptions of its purpose, background, features, functionality, and usage examples. These repositories serve as valuable starting points for researchers and developers, providing alternative solutions, benchmarking capabilities, and opportunities for customization. Furthermore, they can be integrated with machine learning algorithms to facilitate further research and development in the field of protocol learning and emergence.

In conclusion, the open-source repositories described in this report offer a wealth of resources for researchers and developers interested in protocol learning and emergence. By providing frameworks, tools, and algorithms, these repositories enable the exploration of alternative solutions, benchmarking, and customization. They also facilitate the integration of machine learning techniques, opening up new avenues for research and development in this field.

List of authors

Company	Author	Contribution
Oulun Yliopisto	Septimia Sarbu	Contributor and lead of task 4.1
Interdigital Europe Ltd	Muhammad Awais Jadoon	Contributor
Nokia Networks France	Bryan Liu	Contributor
Aalborg Universitet	Ramoni Ojekunle Adeogun	Contributor and project technical lead
Nokia Networks France	Alvaro Valcarce	Contributor and lead of WP4
Interdigital Europe Ltd	Ahmet Serdar Tan	Contributor

Table of Contents

List of figures and tables

List of figures:

List of tables:

Abbreviations

5G	Fifth Generation
5G-ACIA	5G Alliance for Connected Industries and Automation
6G	Sixth Generation
BS	Base Station
DCI	Downlink Control Information
L2	Layer 2
MAC	Medium Access Control
MARL	Multi-Agent Reinforcement Learning
MuJoCo	Multi-Joint dynamics with Contact
mMTC	Massive Machine Type Communications
MTD	Machine-Type Device
NR	New Radio
PDCCH	Physical Downlink Control Channel
RA	Random Access
RL	Reinforcement Learning
SDU	Service Data Unit
TTI	Transmission Time Interval
UE	User Equipment

1 Introduction

CENTRIC's deliverable D4.1 is an open-source library containing simulators for various problems in protocol learning/emergence. The problems included in this library address the first three WP4-specific objectives as illustrated in Table 1.

These simulators are software implementations of the problems to be addressed during the next phase of the project. Their purpose is to engage the open-source community at large (beyond project CENTRIC) into exploring alternative solutions to protocol learning challenges, such as the customization of L2 signaling schemes and channel access policies to particular scenarios, the reduction of control-plane overhead, and the objective quantification of protocol performance.

Table 1: WP4 objectives addressed by the various protocol learning open-source simulators

		WP4 objectives		
		1 – To develop AI techniques for emerging Layer 2 protocols that are customized to specific 6G use-cases and scenarios	2 – To minimize the signaling overhead of the emerged protocols	3 – To demonstrate the feasibility of the protocol emergence techniques via numerical simulations
Open-source repositories	Multiple-access-with-MuJoCo-robots	✓		✓
	Random-Access-with-MARL	✓		✓
	DCI-Learning		✓	✓
	6G-infactory-subnetworks	✓		✓

This short report describes the purpose of each and all the delivered repositories, the background to each of them and their features. However, this report is not a software documentation. It does not contain installation instructions, nor an API guide. The repositories themselves are self-contained and include README files addressing these points.

2 Repository 1: Multiple access with MuJoCo robots

https://github.com/CENTRIC-WP4/Multiple-access-with-MuJoCo-robots

2.1 Background

The purpose of the repository titled "Multiple-access with MuJoCo robots" is to provide a minimal working example of a communication network between a base station (BS) and several user equipments (UEs), instantiated as gymnasium environments (CartPole and MuJoCo). The BS and the UEs are connected through data and control channels. The data channels are much fewer than the number the UEs that need to access them to send their data to the BS, hence the multiple-access scenario modelled in this repository. It offers users a starting point of a tractable model of a communication network. This model may be employed as a simple environment for learning and testing novel communication protocols and easily illustrating the advantages of the proposed methods. In addition, it facilitates benchmarking of novel communication protocols emerged with various methods, some of which fall under the paradigm of reinforcement learning.

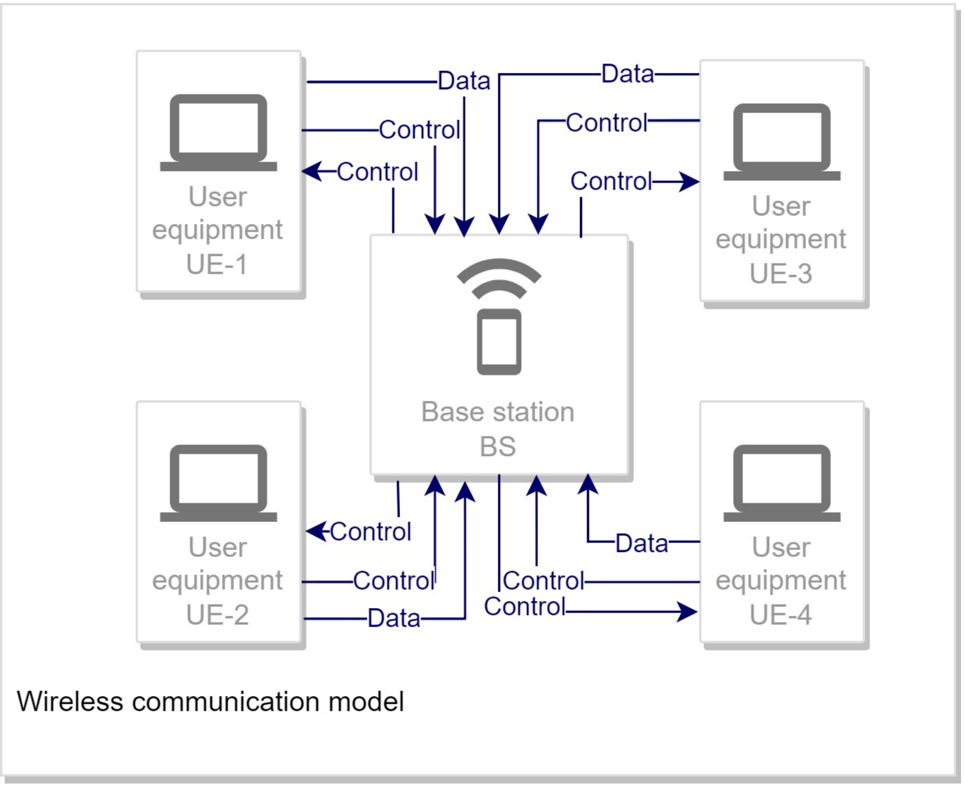

Figure 1: Wireless communication model between a BS and several UEs via the control and data planes.

The wireless communication model implemented in this repository is depicted in Figure 1.1. Supported by a basic version of the contention free protocol with random scheduling, the orchestration of the communication between the BS and the UEs and the channel access is performed via the control plane, while the data packets are sent to the BS via the data plane. A UE state vector is counted as an SDU (service data unit). Corrupted reception of an SDU by the BS is modelled as a Bernoulli distribution with the block error rate probability specified by the user. The communication round ends when a given number of SDUs have been sent by the UEs and received by the BS.

The CartPole and the MuJoCo robots, together with the communication network orchestrated by the BS, provide toy-model scenarios for various 6G use-cases, such as collaborative robots on a smart factory floor, networks of heterogeneous URLLCs (ultra-low-latency) entities, massive machine-type communications (mMTC), Communication and Control Co-design applications etc. Here, the user may design, implement, optimize existing protocols or emerge new ones via AI techniques in general, and reinforcement learning in particular.

2.2 Features and functionality

Users may benefit from this repository because it provides a parallel communication framework between a base-station (BS) and several user equipments (UEs) with a basic version of the contention-free MAC protocol with random scheduling. The communication network is decoupled from the particular physical characteristics of the UEs and BS. To ensure real-time synchronization, the UE-BS communication is simulated using the Multiprocessing feature of python. The BS and UEs exchange messages and data via the Queue data structure, which has the blocking feature. The BS and the UEs represent concurrent processes, which, once they are launched, will communicate through the contention-free protocol with random scheduling. This multiprocessing mechanism ensures the correct synchronization and reception of data packets between the network entities via the control messages. The communication messages are sent and received at the correct time points and the goodput, that is the number of data packets received in each time interval, is computed correctly at the BS. Packets received later by the BS will not be counted erroneously and, thus, higher goodput will not be reported. At each UE, for all the time steps of the simulation, the user can track the signalling messages exchanged between the UEs and the BS and can visualized the data received by the BS. While the communication is running, these two functionalities can be monitored for each time step. As a result, the user can track how often and which UEs access the channel and how much data the BS is receiving, as well as the type of data received.

Due to its modular design, the source code in the repository is easily customizable to the user's needs. For example, replacing the gymnasium agents with any type of entities implemented by the user does not entail any changes in the underlying communication infrastructure. Likewise, the communication protocol between the UEs and the BS may be replaced with one developed by the user, without the need of modifying the network agents or the synchronization process of the communication. The contention-free communication protocol with random scheduling acts as the baseline for the above scenarios. The user may modify the provided source code, to replace the baseline and add their own solution to the protocol emergence challenge.

2.3 Unique capabilities

- **Parallel implementation** of the control message exchanges -> correct computation of the goodput and latency
- **Modularity** -> the number of users, the data traffic model, the communication protocol, the type of UE can be changed independently from other features of the model
- **Versatility ->** if offers a high degree of flexibility in its implementation, that is by tuning the above parameters, it can be applied to many types of 6G use-cases such as URLLC robotic control, mMTC etc.
- **Easily customizable** to the user's needs

2.4 Usage examples

The provided repository implements tractable simulation environments and communication protocols for real-world 6G use-cases. For example, the network of CartPoles may be used to simulate 6G URLLC entities with various latency requirements, the network of MuJoCo robots may be used to simulate collaborative navigation and manufacturing on a smart factory floor. Both types of agents may be integrated in Communication and Control Co-design applications for indoor smart factory scenarios.

2.5 Remarks

In this repository, we have provided a versatile and tractable simulation platform that captures the essential communication features and functionalities desired in 6G use-cases. Easily customizable to the user's needs, it offers a common benchmarking environment for testing novel communication protocols emerged with machine learning. In addition, it provides simulation models characterized by a high degree of adaptability to the requirements of various URLLC and mMTC use-cases.

3 Repository 2: Random access with MARL

https://github.com/CENTRIC-WP4/Random-Access-with-MARL

3.1 Background

The paradigm of massive machine type communications (mMTC) requires rethinking and redesigning of how machine-type devices (MTDs) in the network share the medium. This is because of the peculiar traffic characteristics and MTD characteristics. The traffic is usually sporadic where a few devices may wake up to transmit in the uplink. Moreover, during an event such as fire, the devices may have correlation among them, either spatial or temporal. Furthermore, some packets to be transmitted by the devices may have more priority over the others. Similarly, the users are low-power low-complexity devices in mMTC

For these reasons, conventional multiple access schemes, including grant-free access or scheduling schemes, are not enough to meet the demands of the mMTC. Grant-free random access (RA) makes more sense as scheduled access incurs signaling overhead and when the traffic is mostly sporadic in nature, RA is more efficient. However, RA schemes are known to be inefficient when the traffic load is high because of the number of collisions. There are a few challenges in terms of designing multiple access techniques for mMTC and some of them are listed as follow:

1- The scheme should be adaptive to different traffic changes in the network
2- The scheme should be aware of the different QoS requirements of the devices and different traffic priority requirements
3- The scheme should be scalable to a large number of devices
4- The scheme should be fair

Can we design a multiple access policy using multi-agent reinforcement learning (MARL) for mMTC?

This repository contains RA environment where multiple users are accessing the medium following a random policy. Users are encouraged to design their MARL algorithms for better RA policies considering traffic characteristics and for heterogeneous requirements for the devices.

3.2 Features and functionality

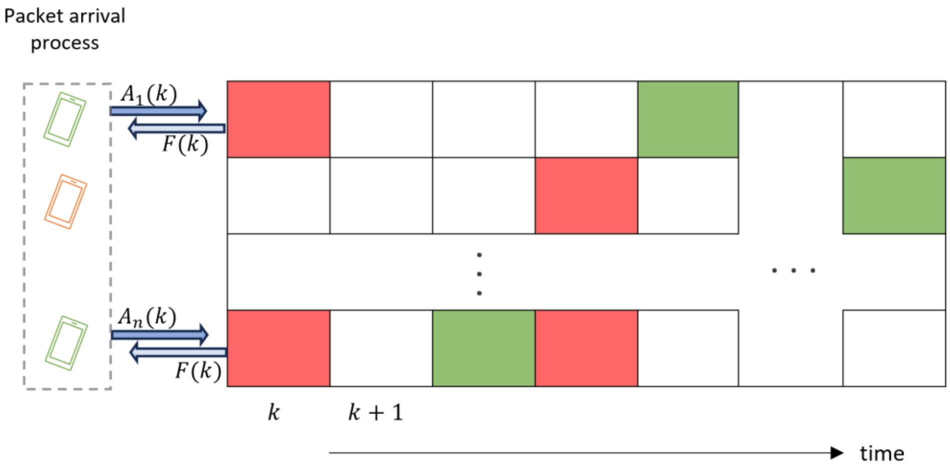

Figure 2: Interaction of agents with the environment. Each agent takes action and receives feedback before calculating the reward and updating the state space.

The core purpose of this repository is to design an RA scheme using MARL that considers the above-mentioned challenges and characteristics of mMTC system. The repository provides an RL-gym environment for RA, in which we consider, *N* agents (MTDs) and and *K* orthogonal resources (channels) as shown in Figure 2.1. Each user may or may not have a packet in its buffer $B_n(k) \in \{0,1\}$. Every user receives M bit feedback from environment at each time slot (discrete time slots, *k*).

Observation Space: the observation space for each agents includes: - IDs of agents (this can also be excluded) - Previous action $A_n(k-1)$, $B_n(k)$, feedback for *M* channels $F_m(k) \in \{0,1\}$

Action Space: Two actions per agent {transmit or not transmit} over the m-*th* channel or silent.

Reward: Several rewards may be used depending on the requirements of the task or objective; The objective might be to increase the throughput, or/and to have fairness among users or decrease packet delay.

Each MTD is considered as an agent. An expectation from the repository is to design RA policy or policies in such a way that each MTD can take action of transmission on the channel in a distributed manner. The repository is flexible in terms of designing your own state-space, reward signals and traffic model.

3.3 Usage examples

Let us consider an environment with *N* number of users accessing a wireless channel (m=1). The traffic is generated using random Poisson process with an average arrival rate = 0.01 packets per device (The arrival rate may be customized inside the repository). Design a reward signal which not only provides better average throughput but also provides fairness among users. This means that the overall average success rate among users shouldn't have a high variance. The reward is also customizable, but one may use the existing reward function to measure the performance of their proposed algorithm.

Another example can be, using existing environment, compute average throughput of the system with correlated traffic model where certain users wake up when an event is activated. The devices should be able to resolve collisions in this case in an efficient manner. In this case, the reward should reflect the purpose of the environment; penalize the users when they collide and reward them when they transmit their packets successfully.

3.4 Remarks

Users can benefit from this repository if they want to design MARL-based policies with different traffic models that are well-suited for mMTC. For instance, event-driven traffic models, correlated traffic models or any other customized traffic model, policies considering different priorities of the packets in the buffer. This repository may be used to extend the environment of RA using MARL. Scalability is a big issue in MARL and this repository may be used to design scalable solutions for MARL for RA. Moreover, users may also use this repository to compare different MARL algorithms for centralized and decentralized training and decentralized execution.

4 Repository 3: DCI learning

https://github.com/CENTRIC-WP4/DCI-Learning

4.1 Background

Downlink control information (DCI) learning is about training ML models to understand the potential reduction of the length of a DCI message when the control bits are temporally correlated. Radio resources are scarce, needing to accommodate massive numbers of Eser Equipments (UEs). With the growing requirements of various wireless services, the management of radio resources becomes important. In particular, control messages are essential for a base station to manage the connection status for a user equipment. However, the number of radio resources is limited. To increase the control channels capacity (the number of users that can be served by a base station) without occupying additional radio resources, reducing the length of a DCI message is a direct and effective solution.

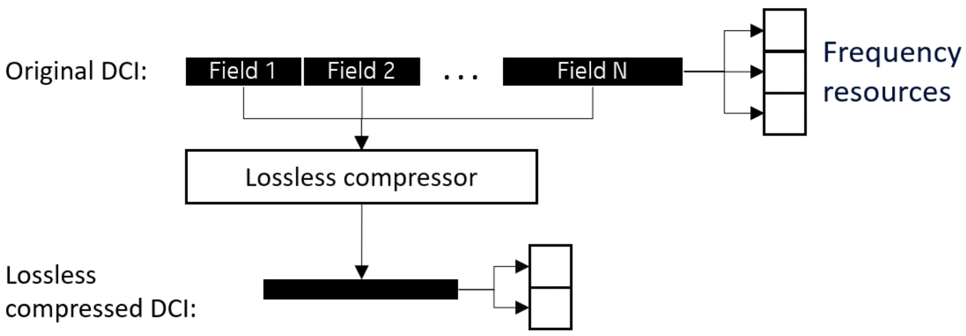

Figure 3: Implementation of a lossless compressor to reduce the DCI length.

A lossless compressor can accomplish the task of reducing the length of a DCI message. Nevertheless, conventional lossless compression algorithms are designed through a look-up table. The implementation is simple but correspondingly, the compression ratio is limited. Our repository generates temporal correlated binary bits for the interested developer to design lossless compression algorithms that reduce the length of DCI messages. Two baseline lossless compression algorithms, Huffman coding and Lempel-Ziv-Welch, are provided for the developers to evaluate the performances. Additional features such as 5G NR PDCCH encoding and decoding follow the open sources from Sionna and Matlab open codes.

4.2 Features and functionality

The main feature of the repository is the generation of correlated binary sequences. There are 3 ways of generating the correlated binary data.

- *K-dependent*: given the sequence of correlation coefficients and the activation ratio per bit, the adjacent K bits are correlated.
- *One-dependent*: given the sequence of correlation coefficients and the activation ratio per bit, the adjacent 2 bits are correlated.

- *Decaying Product*: the closer of the indices of the bits, the stronger the correlation. Given a correlation coefficient γ, the bits with indices i and j are correlated by a coefficient of $\gamma^{|i-j|}$.

On top of that, a simple even resource scheduler and modulation and coding scheme selector are implemented to form a portion of the fields in a DCI message. With a preset number of transmission time intervals, user equipment and the type of channel model, the main function of the repository generates DCI messages for each user equipment.

Additionally, there are 2 baseline algorithms provided for the developers to compare performance against other lossless compression methods.

- *Huffman coding*: it collects the frequencies of symbols in the input data and generates a tree-based binary representation for each symbol. Apparently, the more accurate of the tree-based representation from the actual frequency of each symbol, the greater compression ratio that Huffman coding can achieve.
- *Lempel-Ziv-Welch*: instead of creating a tree-based representation for each symbol as in Huffman coding, Lempel-Ziv-Welch generates a dictionary to store the possible combination of symbols and encode the input data by choosing the longest substring to achieve lossless compression. Lempel-Ziv-Welch is easy to implement but may not perform well when there are fewer repeating patterns, and the underlining correlation structure needs additional refinement.

Please note that the preset parameters for the baseline algorithms such as dictionary size can influence the final performance. The parameters may need to be optimized according to the system setup.

4.3 Usage examples

A DCI to be sent at the current TTI might be correlated to the control messages that have been sent in the previous TTIs. Exploring the temporal correlation provides the chance to greatly reduce the length of a DCI message. As a result, the code rate could be decreased, which results in a better decoding performance and the reliability of a wireless system can be improved. Baseline algorithms of Huffman coding and Lempel-Ziv-Welch are provided in this repository. An example of compression result from Huffman coding is shown below, where the dark blue point indicates the null space, while the white and light blue points refer to the binary bit of 0 and 1, respectively.

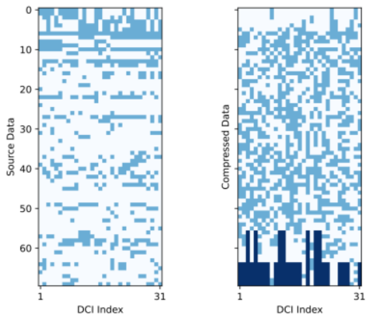

Figure 4: An example of lossless compressed data using Huffman coding

4.4 Remarks

Massive connectivity plays an important role in future 6G networks and to achieve a better capacity of a wireless system, improvement on the standards of control messages is necessary. This repository serves as a great start for researchers and developers to investigate characteristics of DCI message and it is a wonderful tool to generate training and test data for lossless compression techniques.

5 Repository 4: 6G in-factory subnetworks

https://github.com/CENTRIC-WP4/6G-infactory-subnetworks

5.1 Background

6G in-X subnetworks are envisioned as low-power short-range cells to be installed inside entities such as robots, production modules, vehicles, or even human bodies for the support of potential life-critical control operations. Potential applications of such in-X subnetworks are depicted in Figure 4.1. In CENTRIC, we focus on industrial applications where in-X subnetworks are used to replace control services that currently run over wired networks translating to enhanced flexibility, modularity, and ease of re-configuration.

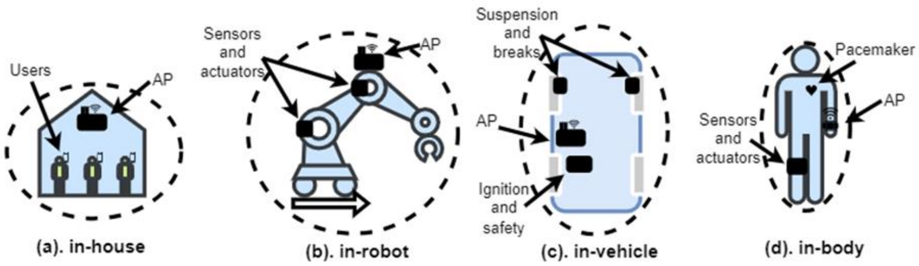

Figure 5: Selected use cases for 6G in-X subnetworks

In this deliverable, we focus on the industrial use case of in-X subnetworks with example applications including motion, force/torque, and position/proximity control. To guarantee the stability of these heterogeneous control applications, a subnetwork may need to support varying communication and/or control requirements. This raises the challenge of how to efficiently support communication to/from devices (e.g., sensors and actuators) driving heterogeneous control applications over a limited number of time/frequency resources without sacrificing the stability of the control operations. For example, transmissions between sensors and the access point within a subnetwork may necessitate different traffic characteristics, end-to-end latency, survival time, and/or payload size. There is therefore the need for a protocol to manage transmissions within a subnetwork such that intra-subnetwork interference is avoided. Another important challenge that needs to be addressed for in-X subnetworks is that of efficiently managing the limited radio resources such that interference is minimized.

In this repository, we present software implementations of 6G in-X subnetworks for industrial applications in a multi-agent reinforcement learning environment to drive research into the challenges of developing RL techniques that solve both the problem of learning a protocol for wireless driving heterogeneous control services and radio resource allocation.

5.2 Features and functionality

The 6G in-X subnetwork repository has been implemented in such a manner that machine-learning algorithms can be easily integrated and evaluated. The multi-subnetwork industrial scenario has been implemented using the GYM environment from OpenAI making it compatible with most RL tools. The main features of the repository are described next:

1. **5G-ACIA defined indoor factory deployment**: The multi-subnetwork implementation is based on an indoor industrial factory scenario inspired by existing production facilities of manufacturing companies as identified by the industry initiative - 5G alliance for connected industries, and automation. Such a layout is depicted in Figure 4.2, as a 180 m×80 m hall containing several separate areas for production, assembly, storage, and human work zones. Multiple in-robot subnetworks can be deployed with the task of transporting materials or tools around the facility. The alleys separating laboring areas are 5 m wide taking up ∼ 1600 m2 of the factory area and are outlined as two-lane roads in a right-handed traffic setting. As a special case, a single-subnetwork environment can be simulated by setting the number of subnetworks to 1. This is to allow for the development and evaluation of machine learning algorithms for supporting heterogeneous control applications within a single subnetwork.
2. **Standardized propagation models**: The wireless characteristics calculations in this implementation are based on models specified by 3GPP for in-factory environments.
3. **Benchmark algorithms**: The repository contains the implementation of selected baseline algorithms that can be used to benchmark the performance of novel algorithms for heterogeneous service multiplexing and/or subband allocation in 6G in-X subnetworks. The implemented benchmark algorithms include:
   - **Random subband allocation**: allocate a subband to each subnetwork randomly.
   - **Greedy subband selection:** a distributed algorithm involving the selection of the least interfered subband by each subnetwork based on sensing measurement of aggregate interference power or signal-to-interference power ratio (SIR)
   - **Centralized coloring**: implementation of improper coloring for subband allocation using measurements of the mutual interference matrix.

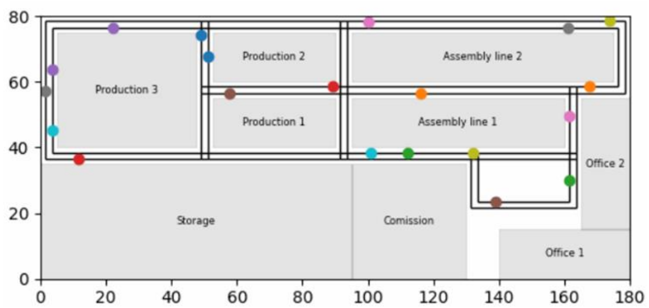

Figure 6: Example of the implemented in-factory deployment of 6G in-X subnetworks.

5.3 Usage examples

As an example, the included “main.py” file can be used to train and evaluate two multi-agent reinforcement learning policies (multi-agent DDQN and multi-agent PPO) for subband selection.

The repository contains a *config* file with default in-factory settings based on 3GPP specifications for indoor factory environments. Users can specify input parameters such as the number of subnetworks, subnetwork speed, type of problem, number of sub-bands, etc. in the *config* file. Once the main file is executed, relevant statistics including subnetworks trajectory information, state measurements and corresponding actions, reward signals and other metrics are generated and stored. An example of the reward signals extracted from stored data after training of MARL agents using the repository is shown in Figure 7.

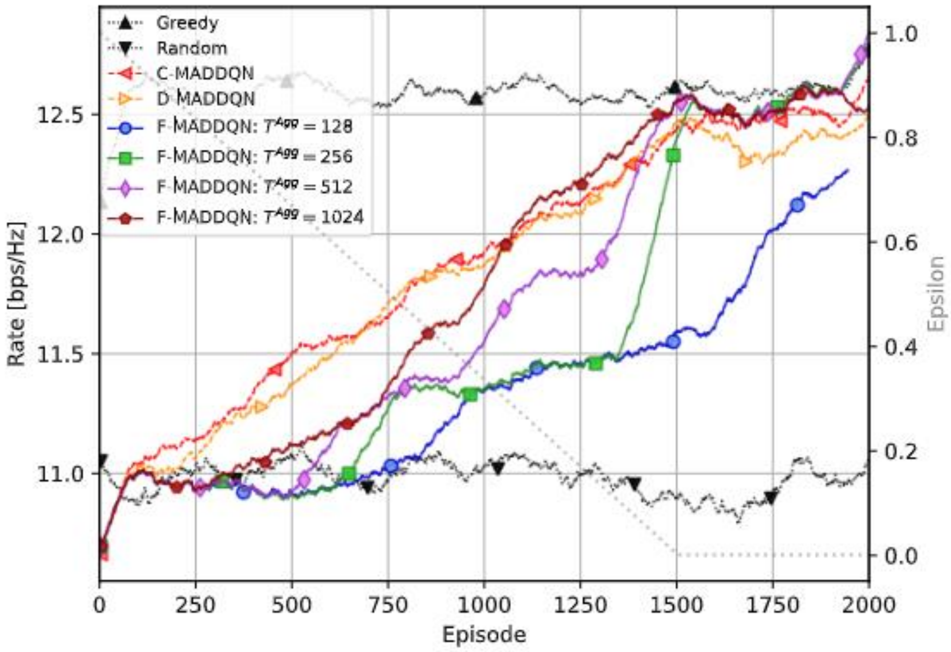

Figure 7: Example of training curves generated using the in-factory subnetwork repository for a scenario with 20 subnetworks.

5.4 Remarks

This repository provides a tool for the development and evaluation of layer 2 and RRM algorithms for 6G in-X subnetworks supporting industrial control operations. The repository is structured in such a way that new machine-learning algorithms can be seamlessly integrated and evaluated.