This paper has been accepted for publication on IEEE Future Networks World Forum (FNWF):
WS5: Federated Testbed as a Service for Future Networks: Challenges & the State of the Art, Montreal, Canada, October 2022

NeXT: A Software-Defined Testbed with Integrated Optimization, Simulation and Experimentation

Jiangqi Hu[1], Maxwell McManus[1], Sabarish Krishna Moorthy[1], Yuqing Cui[1],
Zhangyu Guan[1], Nicholas Mastronarde[1], Elizabeth Serena Bentley[2], and Michael Medley[2]
[1]Department of Electrical Engineering, University at Buffalo, Buffalo, NY 14260, USA
[2]Air Force Research Laboratory (AFRL), Rome, NY 13440, USA
Email: {jiangqih, memcmanu, sk382, yuqingcu, guan, nmastron}@buffalo.edu,
{elizabeth.bentley.3, michael.medley}@us.af.mil

***Abstract*—To support rigorous and repeatable experimental evaluation of wireless networked systems, the community has made significant efforts to develop experimentation platforms. However, existing platforms primarily focus on the data plane, i.e., the forwarding infrastructure, without explicitly considering the control plane. To fill this gap, in this work we develop *NeXT*, a software-defined testbed with integrated wireless network simulation, experimentation and optimization capabilities. We first design the data plane, which integrates an event-driven broadband wireless network simulator called *UBSim* and a software-defined wireless network testing facility called *RoboNet*. We then design NeXT's control plane, where a software toolchain is developed and deployed to support both traditional model-based optimization and new data-driven control techniques. Finally, we validate the effectiveness of NeXT by considering a series of wireless network optimization and control problems.**

I. Introduction

In the past decades, the evolution of wireless network systems has significantly changed and will continue to change the way we live and work, our commercial activities as well as national security. However, as of today the wireless research community is still lacking a mature ecosystem to support rigorous and repeatable experimental evaluation of wireless networked systems. To fill this gap, significant efforts have been made by the community. A recent milestone is the NSF Platforms for Advanced Wireless Research (PAWR) program, which attempts to develop four large-scale outdoor experimentation platforms for advanced wireless research [1]. As of today, three of them have already been developed and are available to the wireless community. These are POWDER-RENEW for experiments in the sub-6 GHz frequency bands [2], COSMOS for experiments in both sub-6 GHz and mmWave frequency bands as well as edge computing [3], and AERPAW for experiments for wireless UAV systems [4].

While existing community shared facilities have significantly advanced experimental research for new wireless systems, it is still challenging to fully meet the needs of experimental wireless research in the era of data-driven networking. First, to simplify the modeling, control and optimization of heterogeneous NextG networks, data-driven control based on Artificial Intelligence (AI) and Machine Learning (ML) has attracted significant research attention [5]. However, the effectiveness of AI/ML algorithms largely relies on sufficient well-labeled data for policy training. It is typically time consuming and sometimes unsafe to collect training data in real-world environments. Second, the design, prototyping and verification of new network control algorithms require engineers to grapple simultaneously with mathematical modeling, distributed control, protocol design across different layers of the protocol stack, as well as their implementation and deployment. This process is typically complex, tedious and error-prone.

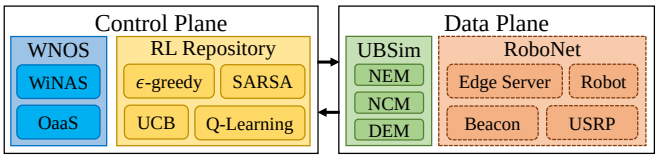


Fig. 1: NeXT testbed architecture.

To address these challenges, *in this paper we present NeXT, a software-defined wireless Network X-Control Testbed, where "X" refers to optimization, simulation and experimentation.* In a nutshell, NeXT provides an integrated testing framework, in which researchers are allowed to generate in an automated manner distributed cross-layer network optimization algorithms, simulate the generated algorithms in software, and then validate the simulation results based on testbed experiments. The overall architecture of NeXT is illustrated in Fig. 1, where there are two planes, *Data Plane* and *Control Plane*. The former provides simulation and experimentation capabilities, and the latter implements network optimization and control functionalities.

The main contributions of this work are as follows:

- We first design the data plane for the NeXT testbed. In this plane, we first integrate UBSim with NeXT for software-based network simulation. UBSim is an event-driven simulator that has been developed at University at Buffalo for broadband (microwave, mmWave and ter-

ACKNOWLEDGMENT OF SUPPORT AND DISCLAIMER: (a) This material is based upon work supported by United States Air Force under Contract No. FA8750-20-C-1021; (b) Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.
Distribution A. Approved for public release: Distribution unlimited AFRL-2022-3290 on 12 July 2022.

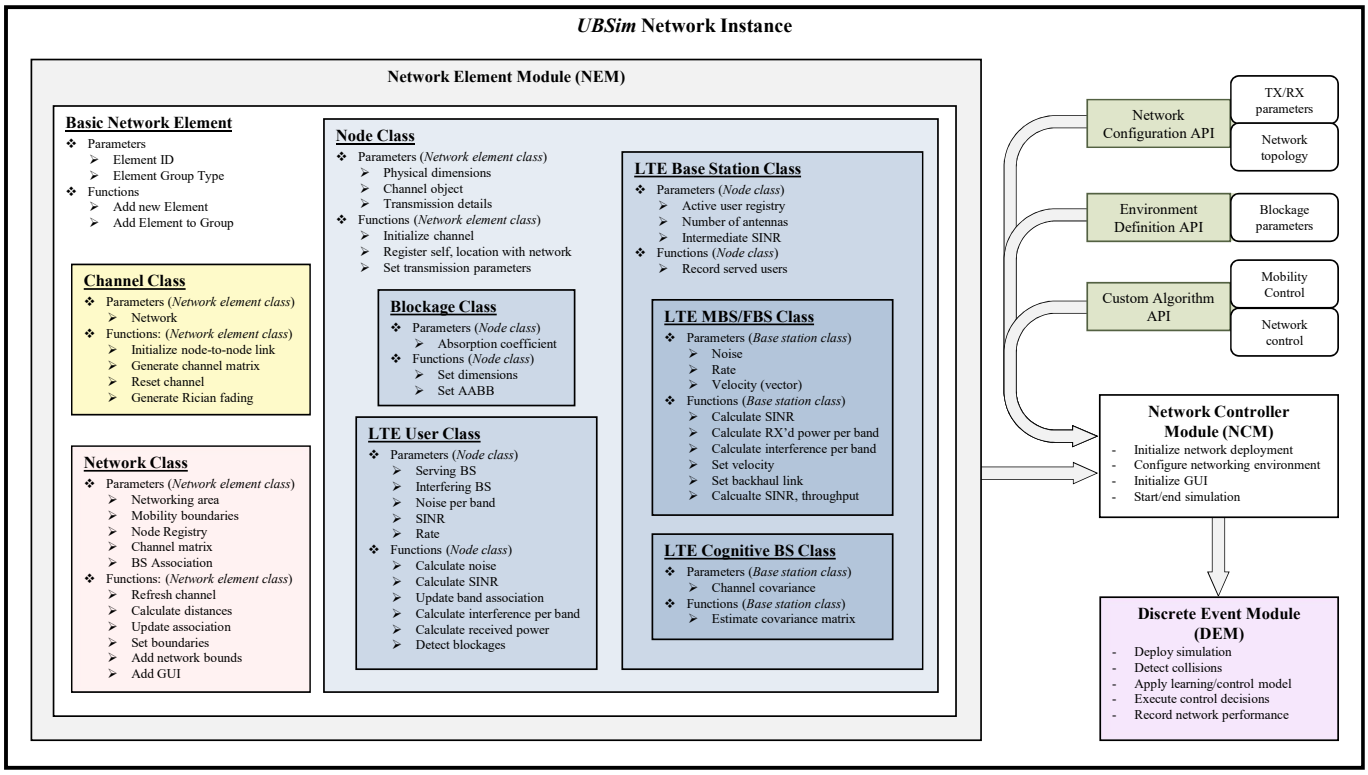


Fig. 2: Architectural overview of UBSim network simulator.

ahertz bands) aerial and ground wireless networking. We also develop a testing facility for mobile networks based on software-defined radios.

- We then design NeXT's control plane, which supports traditional model-based control and new data-driven control techniques. For the former, Wireless Network Operating System (WNOS) [6] has been deployed to enable automated generation of distributed cross-layer control algorithms. For the latter, a reinforcement learning (RL) repository is developed supporting various RL algorithms.
- We showcase the optimization, simulation and experimentation capabilities of the NeXT testbed considering a series of wireless network control problems. A set of APIs have been designed to simplify access to NeXT's data and control planes.

II. Data Plane Design

The data plane provides the forwarding infrastructure for the NeXT testbed. As illustrated in Fig. 1, two forwarding infrastructures have been designed: *UBSim* for software-based network simulations and *RoboNet* for experiments based on software-defined radios.

A. Software Simulations based on UBSim

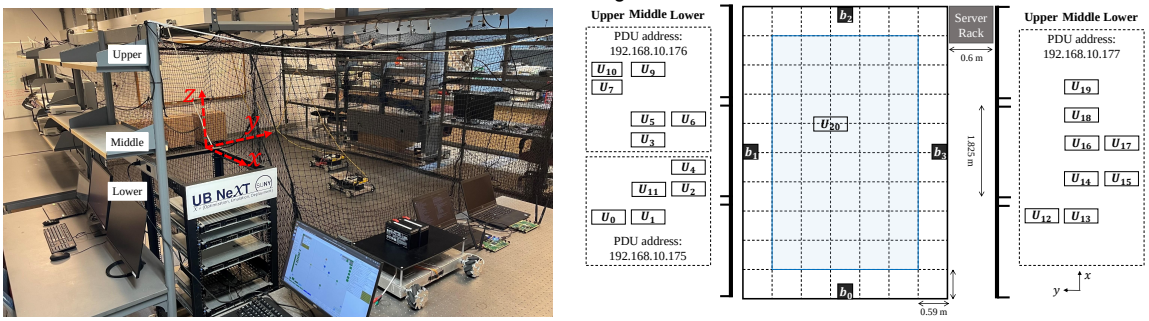
UBSim is a new wireless network simulator written in Python and based on the SimPy discrete event simulation framework [7]. As depicted in Fig. 2, UBSim comprises three primary modules to handle the behavior definition of various network elements, as well as three APIs to support a wide range of custom networking scenarios. Specifically, the network element module (NEM) defines the behaviors of all types of communication nodes, environmental blockages, the channels, and the network as a whole. The network controller module (NCM) organizes the information from the NEM and each user API to define the network topology, environment, and control objective. The discrete event module (DEM) then takes the resulting full scenario definition and starts the discrete event-driven simulation process.

The simulator APIs offer full configuration over network behaviors, environment specification, and control specification. Specifically, the network configuration API provides control over parameters such as frequency, bandwidth, mobility, and location of nodes, as well as networking area and propagation characteristics. The environmental definition API provides control over the locations and sizes of blockages as well as their RF absorption coefficients over different frequency bands. In general, all environmental features are modeled as blockages within the networking area. Finally, the custom algorithm API provides access to the runtime behavior of all the nodes, such as mobility, transmission patterns, band association, among others. Particularly, this API module provides direct support for experimental applications of ML for tasks such as network automation and self-configuration.

B. Software-Defined Forwarding Infrastructure: RoboNet

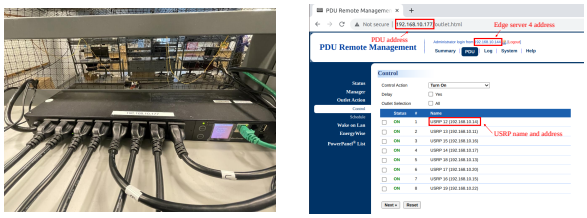
The design objective of RoboNet is to support experiments in wireless networks with mobile robots, such as mobile hotspots [8] and wireless UAVs [9]. The testbed is located in

This paper has been accepted for publication on IEEE Future Networks World Forum (FNWF):
WS5: Federated Testbed as a Service for Future Networks: Challenges & the State of the Art, Montreal, Canada, October 2022



(a) (b)

Fig. 3: (a) Snapshot of the RoboNet testbed; (b) RoboNet network topology.



(a) (b)

Fig. 4: (a) Snapshot of PDU setup; (b) PDU remote management interface.

238 Davis Hall on the University at Buffalo's North Campus. Fig. 3 shows a snapshot of RoboNet and the corresponding topology. At the center of RoboNet is a netted enclosure of dimension $6 \times 4 \times 2.1$ m^3, providing a safe space for robot navigation. For mobile nodes, three wireless robots have been designed based on SuperDroid vehicles and USRP SDRs. An indoor navigation system is also designed based on Marvelmind beacons to provide indoor localization for the robots. For static nodes, a set of USRP SDRs have been deployed over the shelves on the left and right sides of the net enclosure. All the static software radios are controlled by a server rack of five Dell workstations. The mobile software radios are controlled by the onboard computing hosts.

Static Nodes. The static nodes consist of 19 USRP N210 and 5 USRP B210 SDRs. Each USRP N210 operates at frequency from DC to 6 GHz and can process up to 50 mega samples per second (MS/s). Each USRP N210 is equipped with a CBX daughterborad and two VART900/VART2450 antennas. These USRP SDRs are connected via two switches to a server rack, comprising four Dell EMC R340 PowerEdge workstations for baseband signaling processing. Each USRP B210 is designed for low-cost experimentation with continuous frequency coverage from 70 MHz to 6 GHz. Each USRP B210 is also equipped with two VART2450 antennas.

The USRP SDRs are powered via three remotely accessible CyberPower Power-Distribution-Units (PDUs), as shown in Fig. 4(a). These PDUs are assigned Ethernet LAN IP addresses 192.168.10.175, 192.168.10.176 and 192.168.10.177 and connected to edge servers via switches. Figure 4(b) shows the PDU remote management interface, via which experimenters can power on/off USRPs in real time or at scheduled times.

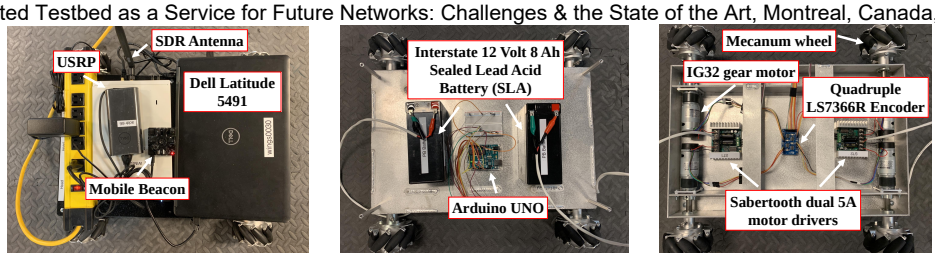
Mobile Nodes. Three software-defined robot vehicles have been designed for RoboNet based on a combination of SuperDroid robots and USRP SDRs. Snapshots of the robot vehicles are shown in Fig. 5. The SuperDroid robot serves as the mobile carrier of the software radios. A programmable Mecanum wheel vectoring robot has been used in the current design of the mobile nodes. Each robot comprises 4 Mecanum wheels, 4 IG32 gear motors, 2 Sabertooth dual 5A motor drivers, 1 Quadruple LS7366R Encoder and 1 Arduino UNO controller. Each robot is powered by two 18V/2.4A PB (lead-acid) batteries. This allows each robot vehicle to carry up to 50 lbs of payload, including the USRP SDRs and their controlling host. Each robot is equipped with USRP SDRs for programmable wireless communications. Currently, both USRP N210 and B210 can be supported by mobile nodes.

A Dell Latitude 5491 laptop with Intel Core^{TM} i7-8850H CPU @ 2.6GHz*12 is used for robot control, USRP SDR control and baseband signal processing. The connection between the controlling laptop and the robot vehicle is established by an Arduino via USB port "/dev/ttyACM0". The mobile beacon is connected to the laptop via USB port "/dev/ttyACM1". Finally, the movement of the robot is controlled and navigated by the Arduino and the beacon via serial communications.

Indoor Positioning System. Because of the poor reception of GPS signals in indoor environments, an indoor positioning system has been deployed, as shown in Fig. 6. The system consists of a controller modem (Fig. 6(a)) and 7 precise (with accuracy of ±2 cm) Marvelmind Super-Beacons (Fig. 6(b)). Based on this system, the location of the mobile beacon can be calculated using trilateration based on the propagation delay of ultrasonic signals to a set of stationary beacons.

The 7 super beacons are divided into two groups: 4 static and 3 mobile beacons. As shown in Fig. 3(b), the 4 static beacons, b_1, b_2, b_3 and b_4, are attached to the four sides of the protecting net. For example, Fig. 6(b) shows the deployment of b_1, which can communicate with the controller modem, its neighbour beacons and the mobile beacon using the selected frequency (19/25/31/37 kHz). According to the exchanged information among the static beacons, the mobile beacon and the modem, the robot locations will be updated in real time. We adopt a Non-Inverse Architecture to set up the navigation system and 31 kHz is used as the communication frequency.

Finally, the controller modem is connected to the edge server via a USB port. Through the control dashboard at the server, experimenters can define a network map by assigning the origin point of the 3D network, configure beacon



(a) (b) (c)

Fig. 5: Snapshots of mobile node. (a) USRP software radio, control host, laptop, and mobile beacon; (b) Power unit and Arduino controller; and (c) Bottom view: motors, motor drivers and encoder.

(a) (b)

Fig. 6: (a) Controller Modem; (b) Super Beacon.

parameters (e.g., beacon address and mode), and monitor the movements of the mobile beacons mounted on the robots.

III. Control Plane Design

The control plane supports both traditional model-based control and emerging data-driven control. To this end, WNOS has been deployed over NeXT for automated generation of distributed cross-layer control algorithms, and an RL repository has been designed supporting various RL algorithms. The control plane is deployed over the edge servers.

A. Network Modeling and Optimization Support

It is tedious and error-prone to manually model and optimize forwarding infrastructure in the data plane. To address this challenge, WNOS [6] has been deployed over NeXT. The goal of WNOS is to abstract the data plane forwarding infrastructure and automatically generate distributed solution algorithms that can be deployed on the NeXT's data plane, i.e., *UBSim* and *RoboNet*. The network abstraction provides a set of APIs, based on which experimenters can characterize in a centralized manner the desired network behaviors before actual deployment. The automated control program generation is enabled by *disciplined instantiation (DI)* [6], based on which user-defined abstract centralized network control problems can be decomposed into a set of distributed subproblems.

WNOS supports a wide set of network control problems in both static and mobile networks. These include, but are not limited to, *rate maximization, power minimization, end-to-end delay minimization, and movement optimization*. WNOS also provides a rich set of APIs, based on which experimenters are allowed to define more sophisticated control problems in next-generation broadband networks spanning across multiple frequency bands, e.g., microwave, mmWave as well as THz bands. Below are some examples of the APIs.

Example APIs: Experimenters can use `attach(·)` to add elements to the network and `connect(·)` to link one or more network elements. With `install_model(·)`, one can install an expression model for a network element attribute. Based on `get_expr(·)`, `mkexpr(·)` and `record_expr(·)`, one can get the expression of a network element, construct the new expression and store the expression in the database, respectively. `set_para(·)` can be used to designate a specific expression as a utility function, constraint, or optimization variable, and `set_soln(·)` can be used to select the solution method to optimize the designated variables.

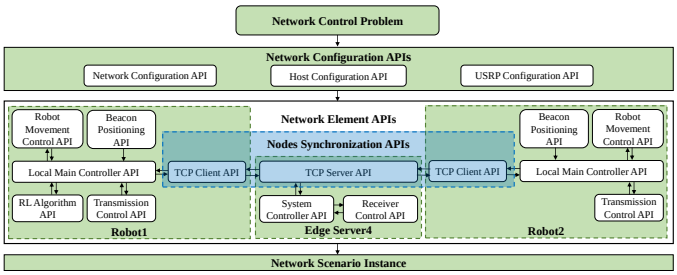


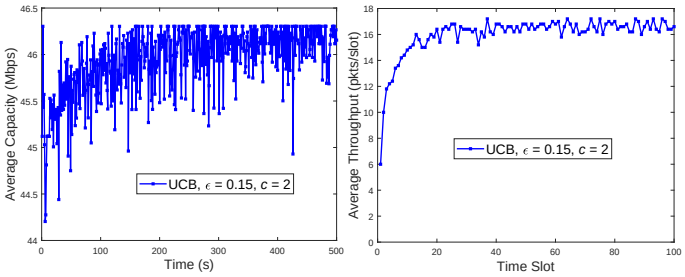
Fig. 7: Network element control interface and experiment management APIs.

B. Data-Driven Network Control Repository

This repository consists of two classes of APIs for data-driven control, i.e., *Basic Class* and *Advanced Class*. The basic class is responsible for network initialization. Examples include the *Environment Initialization API, Variable Initialization API* and *Feedback List Initialization API*. The *Advanced Class* APIs are designed based on *Basic Class* and are used for policy training, including updating states, actions and a value table. Given the number of states and actions specified using the *Configuration API*, the environment can be initialized using the *Environment Initialization API*. Key variables involved in learning algorithms, such as the current state and next state, can be initialized via the *Variable Initialization API*. One is also allowed to choose the *Reward Type* and *Calculator Mode* through the *Configuration API*. Based on these APIs, four classes of RL algorithms have been implemented in the advanced class and can be called via the *RL Algorithm API*. These are epsilon-greedy search, upper confidence bound (UCB) action selection, Q-learning and SARSA. Different reward types and calculator modes have been defined in advance, while experimenters can define custom reward type and calculator mode for their own experiments.

C. Experiment Management APIs

A set of experiment management APIs have also been provided in NeXT's control plane to help experimenters define various network environments, as illustrated in Fig. 7. These include *Network Configuration API*, *Host Configuration API*



(a) (b)

Fig. 8: (a) Average capacity obtained over UBSim; and (b) average number of transmitted packets using RoboNet.

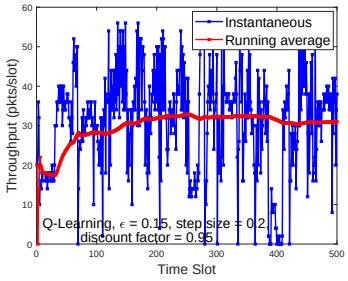


Fig. 9: Instantaneous and running average number of transmitted packets.

and *USRP Configuration API*. Parameters that can be configured via network configuration APIs include network area, center frequency, bandwidth, transmission power, modulation type, slot duration, number of robots, etc. Through host and USRP configuration APIs, experimenters can manage Ethernet address, wireless network address, and port numbers for the SDRs and their controlling hosts.

After the experiment profile has been configured, one can further control various network components via a set of system control APIs deployed at the edge server. These include the *Transmission Control API*, which can be used to control the transmissions of the USRP N210 carried by the robot vehicle; the *Receiver Control API* for controlling data receiving; the *Robot Movement Control API* for controlling robot movement; and finally the *Beacon Positioning API*, based on which experimenters can obtain the real-time positions of the robots.

IV. Example Experiments over NeXT

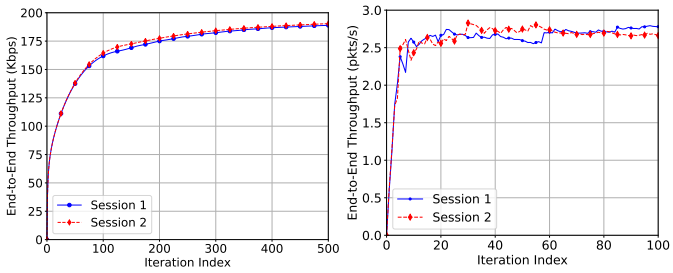
We now test NeXT and showcase its capabilities of optimization, simulation and experimentation considering several network control problems. These include user scheduling in a cellular network, trajectory optimization for a mobile hotspot, and joint rate and power control in a multi-hop network.

User Scheduling. In the first experiment, we consider a wireless network with a hotspot serving a set of users. The transmission time is divided into a set of consecutive time slots. In each time slot, we consider that the hotspot can serve at most one user. The objective of the hotspot is to maximize the aggregate throughput by selecting a user to serve in each time slot. We design control algorithms for the hotspot based on the data-driven network control repository as discussed in Section III-B. Specifically, we consider the UCB action selection algorithm and test it over both UBSim and RoboNet developed in Section II. Firstly, we test the effectiveness of the UCB algorithm in UBSim. Fig. 8(a) plots the achievable capacity averaged over 20 episodes each with 500 time slots. It can be seen that the average capacity improves over time, and this validates the effectiveness of the data-driven network control repository.

Then we further test the data-driven network control repository over RoboNet considering software-defined radios and real-world wireless channels. USRP20 is selected as the transmitter and five USRPs (USRP2, USRP5, USRP9, USRP11 and USRP19) are selected as receivers (confer Fig. 3). The time slot duration is set to 3 seconds. The exploration parameter ϵ and UCB control parameter c are set to 0.15 and 2, respectively. We run 10 episodes of robot navigation, with each episode consisting of 100 time slots. We calculate the average number of received packets in each time slot and the results are shown in Fig. 8(b). It can be seen that the highest throughput can be achieved in around 20 time slots. This further validates the effectiveness of the data-driven network control repository.

Mobile Hotspot Navigation. In the second experiment, we consider a wireless network where a robot carrying a mobile hotspot moves around to serve a set of users. The objective is to maximize the users' aggregate throughput by controlling the robot's trajectory. The network is divided into a set of grid cells, each corresponding to a state of the environment. In each grid cell, the robot has five action options, i.e., move forward, move backward, move left, move right and stay. The reward for each state-action pair is defined as the sum throughput of users. Q-learning is considered in this experiment with exploration probability ϵ set to 0.15, step size of 0.2 and discount factor 0.95. Each episode consists of 500 time-slots, corresponding to 3 hours. We measure the number of received packets and calculate the corresponding running average in each time slot. The experimental results are reported in Fig. 9. It can be seen that the running average converges to around 30 packets/slot. The drop of instantaneous throughput near time slot 400 is caused by the imperfection of the wireless link, which may get disconnected as the robot moves.

Multi-hop Network Optimization. In this experiment we further consider a multi-session multi-hop networks with two sessions and eight nodes. Each session consists of four nodes, namely one source node, two relay nodes and one destination node. The objective is to maximize the network throughput while minimizing the interference between the two sessions. The optimization algorithms are generated with the help of WNOS, which has been deployed over the control plane of NeXT, as described in Section III. The resulting algorithms are deployed over the data plane. Similar to the User Scheduling experiment, we conduct this experiment over both UBSim and RoboNet. The results are reported in Fig. 10. We can see that the control algorithms converge over both UBSim and RoboNet. It is worth pointing out that different link models have been considered in UBSim and RoboNet in their current implementations. In future research we will create a digital twin of RoboNet based on UBSim and test the gap between simulated and real-world performance.



(a) (b)

Fig. 10: Average End-to-End Throughput with (a) UBSim Simulator and (b) NeXT Testbed.

V. Enabled New Research Topics

In this section we discuss the new research topics that NeXT can enable, including *sim-to-real transfer learning*, *robust wireless network control*, *online digital twin construction and optimization*, and *multi-agent reinforcement learning*.

Sim-to-real transfer learning: Towards zero-touch wireless network self-configuration, the proposed framework will connect accelerated learning in the virtual domain with performance evaluation in the real domain. This will enable experimentation towards policy transfer across domains, starting with experimental benchmarks to quantify the reality gap between UBSim and RoboNet and then designing methods to minimize the impact of this gap through an experimental campaign of domain adaptation and novel twin-domain learning algorithms.

Robust wireless network control: The use of robust learning for domain adaptation in the wireless domain has been introduced in [10]. We plan to build on this idea by applying the enabled sim-to-real capabilities of our framework, exploring robust learning as a method of accounting for performance degradation expected from sim-to-real policy transfer.

Online digital twin construction and optimization: Existing methods for generating a virtual model for digital twin applications can be tedious and error-prone. This motivates autonomous virtual environment construction based on mobile sensing techniques such as simultaneous localization and mapping (SLAM). With integrated simulation and experimentation capabilities, the NeXT testbed can enable research of online digital twin construction by providing configurable network simulation environments in UBSim, and verifying the accuracy of the autonomously generated digital twin with ground truth obtained through testbed experiments.

Multi-agent Reinforcement Learning (MARL): The NeXT testbed can support MARL research for development and evaluation of algorithms such as REINFORCE policy gradient (PG), gradient-based partially observable MDP (G(PO)MDP), actor-critic (A2C), or asynchronous actor-critic (A3C). The architecture of UBSim and its supporting APIs can significantly simplify the simulation design process, and enable rapid deployment of environments such as RoboNet through user-configurable parameters such as the number of nodes, distributed or centralized control algorithms, and reward function related to the environment. Additionally, debugging MARL algorithms for can be complicated and extremely challenging due to the distributed nature of data collection and processing. Using UBSim, all information needed for each node will be accessible using one command. This can save time, provide configurable online feedback to display only target data points, and limit redundancy in coding for large-scale MARL problems. Finally, with support of the NeXT testbed, simulation results obtained in virtual environments (e.g., UBSim) can be verified through real-world experiments (e.g., RoboNet).

VI. Conclusions and Future Work

In this work, we have introduced the software-defined testbed *NeXT*, which enables integrated simulation, experimentation and optimization for wireless research. We verified the effectiveness and flexibility of *NeXT* considering both simulation and testbed experiments. We also discussed the new research topics that can be enabled by *NeXT*. In future research, we will i) support experiments in flying networks by integrating UAVs into *NeXT*; ii) support the use of a digital twin for testing self-optimizing networks; and iii) enable remote access to our platform via CloudRAFT, a cloud-based framework for remote access of experimentation platforms that has been developed at the University at Buffalo [11].

References

[1] A. Gosain, "Platforms for Advanced Wireless Research: Helping Define a New Edge Computing Paradigm," in *Proc. of Technologies for the Wireless Edge Workshop*, New Delhi, India, 2018.

[2] https://powderwireless.net/.

[3] D. Raychaudhuri, I. Seskar, G. Zussman, T. Korakis, D. Kilper, T. Chen, J. Kolodziejski, M. Sherman, Z. Kostic, X. Gu, H. Krishnaswamy, S. Maheshwari, P. Skrimponis, and C. Gutterman, "Challenge: COSMOS: A City-Scale Programmable Testbed for Experimentation with Advanced Wireless," in *Proc. of the 26th Annual International Conference on Mobile Computing and Networking*, London, United Kingdom, Sept. 2020.

[4] M. L. Sichitiu, I. Guvenc, R. Dutta, V. Marojevic, and B. Floyd, "AERPAW Emulation Overview," in *Proc. of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation Characterization*, London, United Kingdom, September 2020.

[5] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and Learning in O-RAN for Data-Driven NextG Cellular Networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, Oct. 2021.

[6] Z. Guan, L. Bertizzolo, E. Demirors, and T. Melodia, "WNOS: Enabling Principled Software-Defined Wireless Networking," *IEEE/ACM Transactions on Networking*, vol. 29, no. 3, pp. 1391–1407, June 2021.

[7] "Simpy." [Online]. Available: https://pypi.org/project/simpy/

[8] S. Barrachina-Muñoz, B. Bellalta, and E. W. Knightly, "Wi-Fi Channel Bonding: An All-Channel System and Experimental Study From Urban Hotspots to a Sold-Out Stadium," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2101–2114, Oct. 2021.

[9] J. Buczek, L. Bertizzolo, S. Basagni, and T. Melodia, "What is A Wireless UAV? A Design Blueprint for 6G Flying Wireless Nodes," in *Proc. of the 15th ACM Workshop on Wireless Network Testbeds, Experimental evaluation CHaracterization (WiNTECH'21)*, New Orleans, LA, USA, January 2022.

[10] M. McManus, Z. Guan, N. Mastronarde, and S. Zou, "On the Source-to-Target Gap of Robust Double Deep Q-Learning in Digital Twin-Enabled Wireless Networks," in *Proc. of SPIE Conference Big Data IV: Learning, Analytics, and Applications*, Orlando, Florida, April 2022.

[11] S. K. Moorthy, C. Lu, Z. Guan, N. Mastronarde, G. Sklivanitis, D. Pados, E. S. Bentley, and M. Medley, "CloudRAFT: A Cloud-based Framework for Remote Experimentation for Mobile Networks," in *Proc. of IEEE International Workshop on Communication and Networking for Swarms Robotics (RoboCom)*, Virtual Conference, January 2022.