Learning, Computing, and Trustworthiness in Intelligent IoT Environments: Performance-Energy Tradeoffs

Beatriz Soret, *Member, IEEE*, Lam D. Nguyen, *Graduate Student Member, IEEE*, Jan Seeger, Arne Bröring, Chaouki Ben Issaid, *Member, IEEE*, Sumudu Samarakoon, *Member, IEEE*, Anis El Gabli, *Member, IEEE*, Vivek Kulkarni, Mehdi Bennis, *Fellow, IEEE*, and Petar Popovski, *Fellow, IEEE*

(Invited Paper)

Abstract—An Intelligent IoT Environment (iIoTe) is comprised of heterogeneous devices that can collaboratively execute semi-autonomous IoT applications, examples of which include highly automated manufacturing cells or autonomously interacting harvesting machines. Energy efficiency is key in such edge environments, since they are often based on an infrastructure that consists of wireless and battery-run devices, e.g., e-tractors, drones, Automated Guided Vehicle (AGV)s and robots. The total energy consumption draws contributions from multiple iIoTe technologies that enable edge computing and communication, distributed learning, as well as distributed ledgers and smart contracts. This paper provides a state-of-the-art overview of these technologies and illustrates their functionality and performance, with special attention to the tradeoff among resources, latency, privacy and energy consumption. Finally, the paper provides a vision for integrating these enabling technologies in energy-efficient iIoTe and a roadmap to address the open research challenges.

Index Terms—Edge IoT, wireless AI, distributed learning, distributed ledger technology, autonomous IoT, trustworthiness.

I. INTRODUCTION

A. Towards the Edge

DURING the last decade, the need for connecting billions of Internet of Things (IoT) devices has driven a significant part of the design of computing and communication networks. The number of use cases is countless, ranging from smart home to smart city, industrial automation or smart farming. Many of the applications involve huge amounts of data, and the need for fast, trustworthy and reliable processing of this data is oftentimes infeasible with a cloud-centric paradigm [1], [2]. Moreover, typical hierarchical setups of IoT cloud platforms hinder use cases with dynamically changing context due to lacking self-awareness of the individual sub-systems and the overall system they usher. Alternatively, the architectures are evolving towards edge solutions that place compute, networking, and storage in close proximity to the devices. At the same time, the introduction of machine-driven intelligence has led to the term *edge intelligence*, referring to the design of distributed IoT systems with latency-sensitive learning capabilities [3].

Although the edge-centric approach solves the fundamental limitations in terms of latency and dynamism, it also induces new challenges to the edge system: 1) the system has to deal with complex IoT applications which include functions for sensing, acting, reasoning and control, to be collaboratively run in heterogeneous devices, such as edge computers and resource-constrained devices, and generating data from a huge number of data sources; 2) trustworthiness is a big concern for edge and IoT systems where devices communicate with other devices belonging to potentially many different parties, without any pre-established trust relationship among them; and 3) all those functionalities are increasingly based on a resource-limited wireless infrastructure that introduces latency and packet losses in dynamically changing channels.

Another huge concern for the exponential growth of IoT is its scalability and contribution to the carbon footprint. On the one hand, IoT is key in deploying a huge amount of applications that will reduce the emissions of numerous sectors and industries (e.g., smart farming or energy) [4]. On the other hand, although many of these devices are low-power, the total energy consumption of the infrastructure that support such systems does have a contribution to the digital carbon footprint and cannot be overlooked [5], [6].

B. Intelligent IoT Environments

We coin the term *Intelligent IoT Environment (iIoTe)* to refer to autonomous IoT applications endowed with intelligence based on an efficient and reliable IoT/edge- (computation) and network- (communication) infrastructure that

Beatriz Soret and Petar Popovski are with the Connectivity Section, Department of Electronic System, Aalborg University, 9220 Aalborg, Denmark (e-mail: bsa@es.aau.dk; petarp@es.aau.dk).

Lam D. Nguyen was with Siemens AG, 81739 Munich, Germany. He is now with the Connectivity Section, Department of Electronic System, Aalborg University, 9220 Aalborg, Denmark (e-mail: ndl@es.aau.dk).

Jan Seeger, Arne Bröring, and Vivek Kulkarni are with Siemens AG, 81739 Munich, Germany (e-mail: jan.seeger@thenybble.de; arne.broering@siemens.com; vivekkulkarni@siemens.com).

Chaouki Ben Issaid, Sumudu Samarakoon, Anis El Gabli, and Mehdi Bennis are with the Centre for Wireless Communications, University of Oulu, 90570 Oulu, Finland (e-mail: chaouki.benissaid@oulu.fi; sumudu.samarakoon@oulu.fi; anis.elgabli@oulu.fi; mehdi.bennis@oulu.fi).

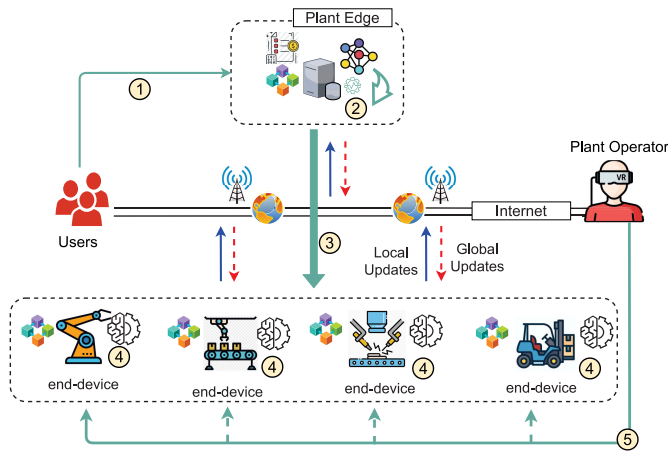Digital Object Identifier 10.1109/TGCN.2021.3138792

Fig. 1. iIoTe in a manufacturing plant.

dynamically adapts to changes in the environment and with built-in and assured trust. Besides the wireless (and wired) networking to interconnect all IoT devices and infrastructure, there are other three key (and power-hungry) technologies that enable iIoTe. The first one is Machine Learning (ML) and Artificial Intelligence (AI), and therefore we talk about *intelligent* IoT environments, comprising heterogeneous devices that can collaboratively execute autonomous IoT applications. Given the distributed nature of the system, distributed ML/AI solutions are better suited for multi-node (multi-agent) learning. *Edge computing* is another defining technology that provides the computation side of the infrastructure and allocates computing resources for complex IoT applications that need to be distributed over multiple, connected IoT devices (e.g., machines and Automated Guided Vehicle (AGV)s). The third pillar is the Distributed Ledger Technology (DLT): rather than traditional security mechanisms, DLT has been identified as the most flexible solution for trustworthiness in a fully decentralized and heterogeneous scenario. Combined with smart contracts, it is possible for the system to autonomously control the transactions from parties without the need for human intervention. All these ingredients are necessary for a fully functional iIoTe, but they have inevitably a significant contribution to the total energy footprint. Our goal is to understand the role of each technology in the performance and energy consumption of an iIoTe.

C. Example: A Manufacturing Plant

A representative use case for iIoTe is a manufacturing plant like shown in Fig. 1, with autonomous collaboration between industrial robot arms, machinery and AGVs. This relies on real-time data analysis and adaptability and intelligence in the manufacturing process, which is only feasible with the edge paradigm. The wireless infrastructure interconnects all the machines and robots to the edge network and enables reliable and safe operation. In the figure, the following scene is depicted: a customer (the end-user) of a shared manufacturing plant orders a product by specifying a manufacturing goal (step 1). In step (2), the needed machine orchestration and associated process plan is determined to manufacture the

desired product taking into account the available computation and communication resources. The event-based process planner at the edge node is responsible for observing the manufacturing process and reacting when the health state of a concerned machine changes. For example, by re-scheduling a given task in a non-responding machine. In step (3), the manufacturing process data is sent to the involved machines, which can include, e.g., mobile robots or an AGV to transport the work-pieces between production points, robotic arms, laser engravers, assembly stations, etc. Let us assume that the task requires a robot to pick up a work-piece and place it in different machines for its processing. As these machines may be operated by the plant owner or a third-party operator, contractual arrangements need to be set up, for which a distributed ledger is used. The ledger registers the details of each task for future accountability. In step (4), the local AI on board of the different end devices comes into play. For example, in the case of the robot as an end device, its AI decides how to pick up a work-piece and place it in the next machine. In case the local AI of the robot cannot complete its task (e.g., because it has not been trained for a similar situation yet), a human takes over remote control (this can be, e.g., a plant operator). After the human intervention, the local AI can be re-trained based on the data captured from the human input. This scene captures the role and interaction of the three technologies mentioned above: edge computing, ML/AI and DLTs/smart contracts. Similar examples can be defined in other domains, such as agriculture (e.g., autonomously interacting harvesting machines), healthcare (e.g., remote patient monitoring and interventions) and energy (e.g., wind plant monitoring and maintenance).

D. Contributions and Outline

In this paper, we analyze the key technologies for the next generation of IoT systems, and the tradeoffs between performance and energy consumption. We notice that characterizing the energy efficiency of these complex systems is a daunting task. The conventional approach has been to characterize every single device or link. Nevertheless, the energy expenditure of an IoT device will strongly depend on the context in which it is put, in terms of, e.g., goal of the communication or traffic behavior. Therefore, we go beyond the conventional single-device approach and use the iIoTe as the basic building block in the energy budget. Contrary to the single device, the iIoTe is able to capture the complex interactions among devices for each of the technologies. The total energy footprint is not just a simple sum of an average per-link or per-transaction consumption of an isolated device, and scaling the number of iIoTe to a large number of instances will give a more accurate picture of the overall energy consumption.

The rest of the paper is organized as follows. In Section II we provide the state-of-the-art of the enabling technologies. Section III analyzes the performance and energy consumption of each enabling technology, and Section IV provides the vision for integrating the enabling technologies in energy-efficient iIoTe. Concluding remarks and a roadmap to address the open research challenges are given in Section V.

II. BACKGROUND AND RELATED WORK

A. Edge Wireless Communications

Edge computing enables the processing of the received data closer to the sensor that generated them. This means a full re-design of the communication infrastructure that must implement additional functionality at the cellular base stations or other edge nodes. The design and performance of communication networks for edge computing has been widely studied in the last years, and an overview can be found in [7] and [8]. One example is the term Mobile Edge Computing (MEC), adopted in 5G to refer to the deployment of cloud servers in the base stations to enable low latency, proximity, high bandwidth, real time radio network information and location awareness. Specifically, the concept was defined in late 2014 by the European Telecommunications Standards Institute (ETSI): *As a complement of the C-RAN architecture, MEC aims to unite the telecommunication and IT cloud services to provide the cloud-computing capabilities within radio access networks in the close vicinity of mobile users* [9]. One of the areas of more research has been the network virtualization and slicing with the MEC paradigm [10]. In the Radio Access Network, several authors have looked at the potential of edge computing to support Ultra-Reliable Low-Latency Communication (URLLC) [11]–[13]. Another research area is the use of machine learning, particularly deep learning techniques, to unleash the full potential of IoT edge computing and enable a wider range of application scenarios [14], [15]. However, most previous works address the communications separately. Even though several papers address the joint communication and computation resource management [16], they represent only the first step towards a holistic design of iIoTe and its defining technologies, as well as the integration with the communication infrastructure.

To optimize the energy efficiency of iIoTe, it is interesting to choose a communication technology that ensures low power consumption and massive connections of devices. In this regard, 3GPP introduced narrowband Internet of Things (NB-IoT), a cellular technology to utilize limited licensed spectrum of existing mobile networks to handle a limited amount of bi-directional IoT traffic. Although it uses LTE bands or guard-bands, it is usually classified as a 5G technology. It can achieve up to 250 kbps peak data rate over 180 kHz bandwidth on a LTE band or guard-band [17], [18].

Compared to other low-power technologies, NB-IoT is interesting for IoT applications with more frequent communications. This is the case for the ones considered in iIoTe, where the intelligent end devices share the updated models frequently and must record new transactions in the ledger. At the same time, NB-IoT keeps the advantages of Low-Power Wide Area (LPWA) technologies: low power consumption and simplicity. Throughout the rest of the paper, we use NB-IoT as a representative wireless technology for our analyses of iIoTe. Other wireless technologies will follow similar access procedures and energy-performance trade-offs.

For an analysis of the energy consumption and battery lifetime of NB-IoT under different configurations we refer the reader to [19]. A key point for this analysis is the study
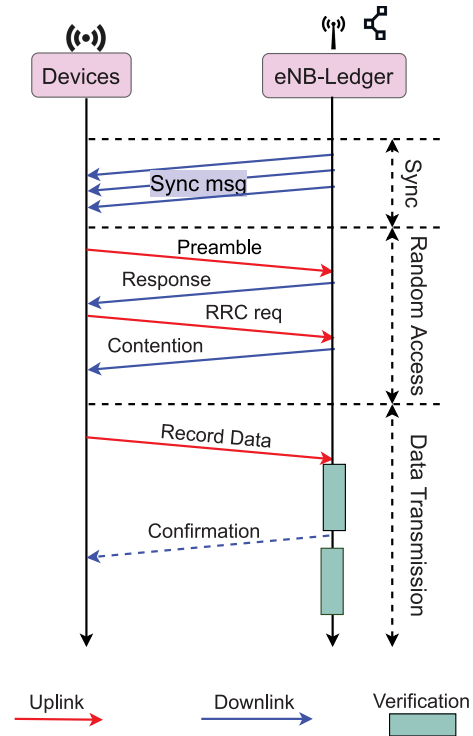

Fig. 2. Random Access procedure in NB-IoT.

of the communication exchange during the access procedure: The devices that attempt to communicate through a base station must first complete a Random Access (RA) procedure to transit from Radio Resource Control (RRC) idle mode to RRC connected mode. Only in RRC connected mode data can be transmitted in the uplink through the Physical Uplink Shared Channel (PUSCH) or in the downlink through the Physical Downlink Shared Channel (PDSCH). The standard 3GPP RA procedure consists of four message exchanges: preamble (*Msg1*), uplink grant (*Msg2*), connection request (*Msg3*), and contention resolution (*Msg4*) (see Figure 2 where the example of recording some data, e.g., a DLT transaction is depicted). Out of these, *Msg3* and *Msg4* are scheduled transmissions where no contention takes place.

The NB-IoT preamble are orthogonal resources transmitted in the Narrowband Physical Random Access Channel (PRACH) (NPRACH) and used to perform the RA request (*Msg1*). A preamble is defined by a unique single-tone and pseudo-random hopping sequence. The NPRACH is scheduled to occur periodically in specific subframes; these are reserved for the RA requests and are commonly known as Random Access Opportunitys (RAOs). To initiate the RA procedure, the devices select the initial subcarrier randomly, generate the hopping sequence, and transmit it at the next available RAO. The orthogonality of preambles implies that multiple devices can access the base station in the same RAO if they select different preambles. Next, the grants are transmitted to the devices through the Narrowband Physical Downlink Control Channel (PDCCH) (NPDCCH) within a predefined period known as the *RA response window*. However, the number of preambles is finite and collisions

can happen. In case of collision, each collided device may retransmit a preamble after a randomly selected backoff time.

The specification provides sufficient flexibility in the configuration of the RA process, which makes it feasible to adjust the protocol and find the right balance between reliability, latency, and energy consumption for a given application. Specifically, the network configures the preamble format and the maximum number of preamble transmission depending on the cell size, and this has an impact on the preamble and the total duration [20]. Increasing the number of preamble transmissions reduces the erasure probability, but at the cost of higher energy consumption and larger latency. The same energy-reliability-latency tradeoff applies to other messages, including the RA response. Moreover, scheduling the NPRACH and NPDCCH consumes resources that would otherwise be used for data transmission. Therefore, each implementation must find an adequate balance between the amount of resources dedicated to NPRACH, NPDCCH, PUSCH, and PDSCH.[1]

B. Distributed Learning Over Wireless Networks

Implementing intelligent IoT systems with distributed ML/AI over wireless networks (e.g., NB-IoT) needs to consider the impact of the communication network (latency and reliability under communication overhead and channel dynamics) and on-device constraints (access to data, energy, memory, compute, and privacy, etc.). Obtaining high-quality trained models without sharing raw data is of utmost importance, and redounds to the trustworthiness of the system. In this view, Federated Learning (FL) has received a groundswell interest in both academia and industry, whose underlying principle is to train a ML model by exchanging model parameters (e.g., Neural Network (NN) weights and/or gradients) among edge devices under the orchestration of a federation server and without revealing raw data [21]. Therein, devices periodically upload their model parameters after their local training to a parameter server, which in return does model averaging and broadcasting the resultant global model to all devices. FL has been proposed by Google for its predictive keyboards [22] and later on adopted in different use cases in the areas of intelligent transportation, healthcare and industrial automation, and many others [23], [24]. While FL is designed for training over homogeneous agents with a common objective, recent studies have extended the focus towards personalization (i.e., multi-task learning) [25], training over dynamic topologies [26] and robustness guarantees [27], [28]. In terms of improving data privacy against malicious attackers, various privacy-preserving methods including injecting fine-tuned noise into model parameters via a differential privacy mechanism [29]–[32] and mixing model parameters over the air via analog transmissions [33], [34] have been recently investigated. Despite of the advancements in FL design, one main drawback in the design of FL is that its communication overhead is proportional to the number of model parameters calling

for the design of communication-efficient FL. In an edge setup with limited resources in communication and computation, this introduces training stragglers degrading the overall training performance. In this view, client scheduling [35]–[37] and computation offloading [12], [38], [39] with the focus on guaranteeing target training/inference accuracy have been identified as a promising research direction.

With client scheduling, the number of communication links are reduced (known as *link sparcification*) and thus, the communication bandwidth and energy consumption of distributed learning can be significantly decreased. Additional temporal link sparsity can be introduced by enforcing model sharing policies that account model changes and/or importance within consecutive training iterations such as the Lazy Aggregated Gradient Descent (LAG) method [40]. Sparsity can be further exploited by adopting sparse network topologies, which rely on communications within a limited neighborhood in the absence of a central coordinator/helper. While such sparsification improves energy and communication efficiencies, it could yield higher learning convergence speed as well as lower training and inference accuracy, in which sparsity needs to be optimized in terms of the trade-off between communication cost and convergence speed. In this view, several sparse-topology-based distributed learning methods including decentralized Gradient Descent (GD), dual averaging [41], learning over graphs [42], [43] and GADMM algorithms [44], [45] have been investigated.

C. Optimizing IoT Application Deployments in IoT Environments

IoT applications typically consist of multiple *components*. For instance, an IoT application could comprise components for secure data acquisition (e.g., based on Blockchain), data pre-processing, feeding the data into a neural network (or even through multiple ones) before it acts upon the outcome of the ML inference, etc. In many cases, such composed IoT applications need to be distributed over multiple, connected intelligent IoT devices. An important aspect is then to optimize this allocation of application components to devices. The result of the allocation is an assignment of components to devices, that fulfills the constraints, and optimizes the performance of the system in some metric. This metric could, for example, maximize the responsiveness of the application or minimize the overall energy consumption, where the latter is reasonable in battery-run wireless systems. An overview of existing allocation approaches is given in [46].

Previous work [47] used Constraint Programming to describe an approach for the efficient distribution of actors to IoT devices. The approach resembles the Quadratic Assignment Problem (QAP) and is NP-hard, resulting in long computation times when scaling up. Samie *et al.* [48] present another Constraint Programming-based approach that takes into account the bandwidth limitations and minimizing energy consumption of IoT nodes. The system optimizes computation offloading from an IoT node to a gateway, however, it does not consider composed computations that can be distributed to multiple devices.

---

[1]It is worth mentioning that 5G has not defined a RA procedure yet but it is expected that, when this happens, it will be heavily based on the described procedure for LTE and the energy consumption/latency tradeoff will follow similar principles.

A Game Theory-based approach is presented in [49] that aims at the joint optimization of radio and computational resources of mobile devices. However, the system local optimum for multiple users only aims at deciding whether to fully offload a computation or to fully process it on device.

Based on Non-linear Integer Programming, Sahni *et al.* [50] present their Edge Mesh algorithm for task allocation that optimizes overall energy consumption and considers data distribution, task dependency, embedded device constraints, and device heterogeneity. However, only basic evaluation and experimentation are done, without performance comparison. Based on Integer Linear Programming (ILP), Mohan and Kangasharju [51] propose a task assignment solver that first minimizes the processing cost and secondly optimizes the network cost, which stems from the assumption that Edge resources may not be highly processing-capable. An intermediary step reduces the sub-problem space by combining tasks and jobs with the same associated costs. This reduces the overall processing costs.

Cardellini *et al.* [52] describe a comprehensive ILP-based framework for optimally placing operators of distributed stream processing applications, while being flexible enough to be adjusted to other application contexts. Different optimization goals are considered, e.g., application response time and availability. They propose their solution as a unified general formulation of the optimal placement problem and provide an appropriate theoretical foundation. The framework is flexible so that it can be extended by adding further constraints or shifted to other optimization targets. Finally, our previous work [53] has leveraged Cardellini's framework and has extended it by incorporating further constraints for the optimization goal, namely the overall energy usage of the application.

D. Distributed Ledger Technologies Over Wireless Networks

In recent years, DLT has been the focus of large research efforts spanning several application domains. Starting with the adoption of Bitcoin and Blockchain, DLT has received a lot of attention in the realm of IoT, as the technology promises to help address some of the IoT security and scalability challenges [54]. For instance, in IoT deployments, the recorded data are either centralized or spread out across different heterogeneous parties. These data can be both public or private, which makes it difficult to validate their origin and consistency. In addition, querying and performing operations on the data becomes a challenge due to the incompatibility between different Application Programming Interfaces (APIs). For instance, Non-Governmental Organizations (NGOs), Public and Private sectors, and industrial companies may use different data types and databases, which leads to difficulties when sharing the data [55]. A DLT system offers a tamper-proof ledger that is distributed on a collection of communicating nodes, all sharing the same initial block of information, the genesis block [56]. In order to publish data to the ledger, a node includes data formatted in transactions in a block with a pointer to its previous block, which creates a chain of blocks, the so called Blockchain.

A smart contract [57] is a distributed app that lives in the Blockchain. This app is, in essence, a programming language class with fields and methods, and they are executed in a transparent manner on all nodes participating in a Blockchain [58]. Smart contracts are the main blockchain-powered mechanism that is likely to gain a wide acceptance in IoT, where they can encode transaction logic and policies, which includes the requirements and obligations of parties requesting access, the IoT resource/service provider, as well as data trading over wireless IoT networks [59]. With the aforementioned characteristics, the advantages of the integration of DLTs into wireless IoT networks consist of: 1) guarantee of immutability and transparency for recorded IoT data; 2) removal of the need for third parties; and 3) development of a transparent system for heterogeneous IoT networks to prevent tampering and injection of fake data from the stakeholders.

DLTs have been applied in various IoT areas such as healthcare [60], [61], supply chain [62], smart manufacturing [63], and vehicular networks [64]. In the smart manufacturing area, the work described in [63] investigates DLT-based security and trust mechanisms and elaborates a particular application of DLTs for quality assurance, which is one of the strategic priorities of smart manufacturing. Data generated in a smart manufacturing process can be leveraged to retrieve material provenance, facilitate equipment management, increase transaction efficiency, and create a flexible pricing mechanism.

One of the challenges of implementing DLT in IoT and edge computing is the limited computation and communication capabilities of some of the nodes. In this regard, the authors in [59], [65] worked on the communication aspects of integrating DLTs with IoT systems. The authors studied the trade-off between the wireless communication and the trustworthiness with two wireless technologies, LoRa and NB-IoT.

III. ENABLING TECHNOLOGIES FOR iIoTe

This section elaborates on the three enabling technologies for iIoTe: 1) distributed learning; 2) distributed computing; and 3) distributed ledgers.

A. Energy-Efficient Distributed Learning Over Wireless Networks

As shown in Figure 1, each end device in the iIoTe has local AI and the whole system relies on FL. We present learning frameworks that are suitable for iIoTe leveraging two techniques: 1) spatial and temporal sparsity and 2) quantization.

1) Dynamic GADMM: Standard FL requires a central entity, which plays the role of a parameter server (PS). At every iteration, all nodes need to communicate with the PS, which may not be an energy-efficient solution especially for a large distributed network of agents/workers, as in the manufacturing use case. Furthermore, a PS-based approach is vulnerable to a single point of attack or failure. To overcome this problem and ensure a more energy-efficient solution, we propose a variant of the standard Alternative Direction Method of Multipliers (ADMM) [66] method that decomposes the problem into a set of subproblems that are solved in parallel, referred to as Group ADMM (GADMM) [44]. GADMM

extends the standard ADMM to decentralized topology and enables communication and energy-efficient distributed learning by leveraging spatial sparsity, i.e., enforcing each worker to communicate with at most two neighboring workers. In GADMM, the standard learning problem (**P1**) is re-formulated as the following learning problem (**P2**):

$$(\textbf{P1}) \quad \min_{\{\boldsymbol{\theta}_n\}_{n=1}^{N}} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) \tag{1}$$

$$(\textbf{P2}) \quad \min_{\{\boldsymbol{\theta}_n\}_{n=1}^{N}} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n)$$
$$\text{s.t.} \quad \boldsymbol{\theta}_n = \boldsymbol{\theta}_{n+1}, \text{ for } n = 1, \dots, N-1. \tag{2}$$

To this end, GADMM divides the set of workers into two groups *head* and *tail*. Thanks to the equality constraint of (**P2**), each worker from the head/tail group exchanges model with only two workers from the tail/head group forming a chain topology. At iteration $k + 1$, giving the models of the tail workers and the dual variables at iteration k, all head workers update their models in parallel since they have no joint constraints. Once the head workers update their models, they transmit their updated model to their neighbors from the tail group. Then, following the same way, every tail worker updates its model. Finally, the dual variables are updated locally at each worker. Following this alternation, GADMM allows at most $N/2$ workers to compete over the available bandwidth compared to N workers for the PS-based approach. With that, GADMM can significantly increase the bandwidth available to each worker, which reduces the energy wasted in competition for communication resources. The energy expenditure for communication is further reduced by including only two neighboring workers. The detailed algorithm is described in [44].

One drawback of GADMM is attributed to its slow convergence compared to standard ADMM. In other words, due to the sparsification of the graph, workers require more iterations for the convergence. To alleviate this issue and combine the fast convergence of standard ADMM with the communication-efficiency of GADMM, we have proposed Dynamic GADMM (D-GADMM) [44]. Not only D-GADMM improves the convergence speed of GADMM, but it also copes with dynamic (time-variant) networks, in which the workers are moving (e.g., the AGVs in the manufacturing plant or the tractors in the agriculture use case), while inheriting the theoretical convergence guarantees of GADMM. In a nutshell, every couple of iterations in D-GADMM, i.e., system coherence time, two things are changing: i) workers assignment to head/tail group, which follows a predefined assignment mechanism and ii) neighbors of each worker from the other group. The idea at high level as is follows: the workers are given fixed IDs, and they share a pseudo-random code that is used every τ seconds, where τ is the system coherence time to generate a set of random integers with cardinality $N/2 - 2$. If n belongs to the set, then worker n is a head worker for this period. The assumption is that workers 1 and N do not change their assignment. i.e., worker 1 is always a head and worker N is always a tail. Head workers broadcast their IDs
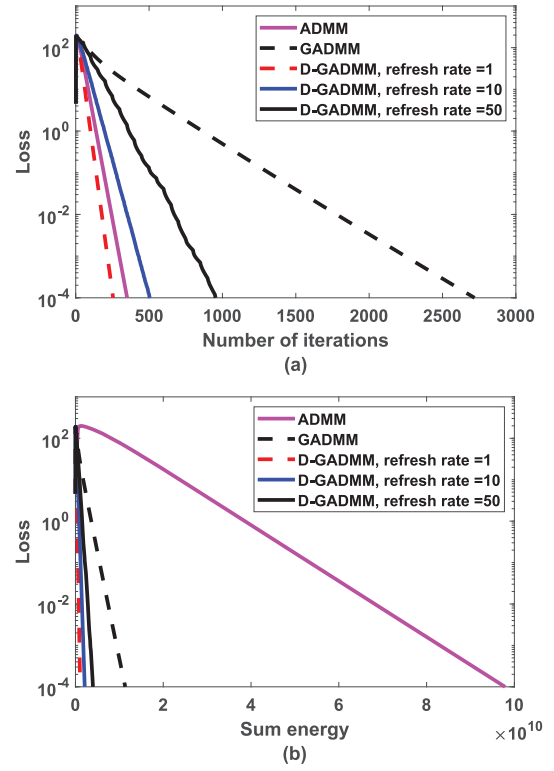

Fig. 3. D-GADMM: loss as a function of (a) number of iterations and (b) total energy consumption.

alongside a pilot signal, then tail workers compute their communication cost to all head workers, and share the cost vector with the neighboring heads. If a tail does not receive a signal from a certain head, the cost to that head is ∞, the same applies to heads. Subsequently, every head locally computes the communication-efficient chain using a predefined heuristic and share it with its neighboring tails. This approach requires two communication rounds and guarantees that every head will compute the same chain. Once the chain information is calculated, each worker will share its right dual variable with its right neighbor to be used by both workers and GADMM continues for τ seconds. It is worth mentioning that we could, e.g., start with a chain $1 - 2 - 3 - \cdots - N$ and move to $1 - 5 - 7 - 4 - \cdots - N$, so only nodes 1 and N preserve their assignments. For further details, the reader is referred to [44] where a comprehensive explanation of the steps of D-GADMM can be found.

In Fig. 3, we plot the objective error in terms of the number of iterations (left) and in terms of sum energy (right) for D-GADMM as well as GADMM and standard ADMM. As we can see from Fig. 3, D-GADMM greatly increases the convergence speed of GADMM and thus decreases the overall communication cost for fixed topology. As a consequence, D-GADMM achieves convergence speed comparable to the PS-based ADMM while maintaining GADMM's low communication cost per iteration.

2) Censored Quantized Generalized GADMM: As pointed out earlier, each worker, in the GADMM framework, exchanges its model with up to two neighboring workers only, which slows down convergence. To reduce the communication
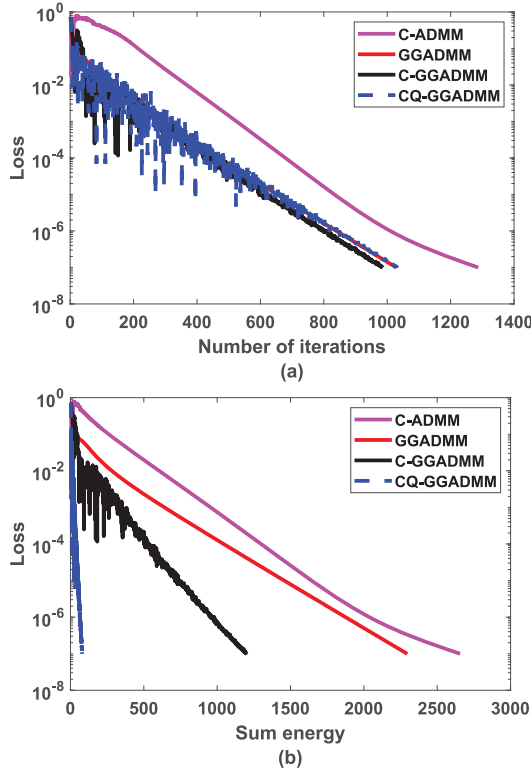
Fig. 4. CQ-GGADMM: loss as a function of (a) number of iterations and (b) total energy consumption.

overhead while generalizing to more generic network topologies, we propose the Generalized GADMM (GGADMM) [45]. Under this generalized framework, the workers are still divided into two groups: 1) head and 2) tail, with possibly different sizes. In other words, the topology is generalized from a chain topology to any bipartite graph where the number of neighbors, that each worker can communicate with can be any arbitrary number and not necessarily limited to two. By leveraging the censoring idea, i.e., temporal sparsity, we introduce the Censored GGADMM (C-GGADMM) where each worker exchanges its model only if the difference between its current and previous models is greater than a certain threshold. To make the algorithm more communication-efficient, censoring is applied on the quantized version of the worker's model instead of the model itself to get the Censored Quantized GGADMM (CQ-GGADMM) [45], [67]. CQ-GGADMM can significantly reduce the communication overhead, particularly for large model size d, since its payload size is $(bd + 32)$ bits compared to the payload size of $32d$ bits for the full precision GGADMM. Since according to the Shannon's capacity theorem, more bits consume more transmission energy for the same bandwidth, transmission duration, and noise spectral density, the communication energy of CQ-GGADMM, compared to the original GADMM, is significantly reduced. Theoretically, CQ-GGADMM inherits the same performance and convergence guarantees of vanilla GGADMM, provided that the censoring threshold sequence is non-increasing and non-negative.

Fig. 4 compares CQ-GGADMM with Censored ADMM (C-ADMM), GGADMM, as well as C-GGADMM

in terms of the loss versus the number of iterations (left) and versus the total sum energy (right) for a system of 18 workers on a linear regression task using the Body Fat dataset [68]. We can observe, from Fig. 4, that CQ-GGADMM exhibits the lowest total communication energy, followed by C-GGADMM, then GGADMM and finally C-ADMM, while having similar convergence speed to GGADMM. This observation validates the benefits of censoring the quantized version of the models before sharing, which makes the proposed algorithm (CQ-GGADMM) more communication and energy efficient.

Finally, it is worth mentioning that motivated by the fact that, in FL, the parameter server is interested in the aggregated output of all workers rather than the individual output of each worker, analog over the air aggregation schemes such as [34], [69]–[71] were proposed. Such schemes were shown to achieve high scalability and significant savings in energy consumption owing to their ability to allow non-orthogonal access to the bandwidth.

B. Optimizing Energy Consumption of Wireless IoT Environments

The next pillar in iIoTe is edge computing. Specifically, we consider the problem of allocating the application components to the available end devices. As first presented in [53], we extend the Integer Linear Programming (ILP) based framework defined by Cardellini *et al.* [52] (Section II-C). In [53] the goal was to minimize the overall energy consumption needed for executing an IoT application. The formulated ILP model is described below. The optimality can be determined with this by feeding it into a solver, such as IBM CPLEX.[2]

We define optimality of the allocation by total energy use over one execution of an IoT application. Energy during the application's execution is consumed in two phases: 1) *device energy*, consumed by a device when executing a component and 2) *edge network energy*, consumed by the device when sending the result of the calculation over the network. Note that "optimal" in this case only describes optimality in the integer model. Given the constraints and the model, we find the optimal assignment, i.e., the one with minimal energy usage.

The optimal network configuration is the assignment of application components to devices that result in the lowest total consumption of energy and satisfies the constraints. The constraints concern the requirements that an assignment must satisfy: Each component should only be allocated once and resource requirements for assigned components should not exceed the resources of the node. This problem is a form of the quadratic assignment problem, and thus is NP-hard.

1) System Model: The application consists of a set of components and edges that interconnect them, modeled as a weighted undirected graph $\mathcal{G}_{app} = (\mathcal{V}_{app}, \mathcal{E}_{app})$. Graph $\mathcal{G}_{app}$ is multi-partite, with vertex set $\mathcal{V}_{app}$ containing the application components, $|\mathcal{V}_{app}| = N$, and edge set $\mathcal{E}_{app} \subset \{t_1 t_2 : t_i \in \mathcal{V}_{app}, i = 1, 2\}$ representing the logical connections between components t_i.

[2]https://www.ibm.com/analytics/cplex-optimizer

TABLE I
PARAMETERS OF ENERGY-AWARE ALLOCATION ALGORITHM

Symbol	Description
R_t	Resources required for the evaluation of component $t \in \mathcal{V}_{\text{app}}$.
O_t	Output of component $t \in \mathcal{V}_{\text{app}}$ for a single received input.
S_t	Computation time required for completing component $t \in \mathcal{V}_{\text{app}}$ once.
P_n	Processing power of node $n \in \mathcal{V}_{\text{net}}$.
R_n	Resources available on node $n \in \mathcal{V}_{\text{net}}$.
C_n	Energy consumption of node $n \in \mathcal{V}_{\text{net}}$ for one unit of computation.
T_l	Energy use for the transfer of one data packet over link $l \in E_{\text{net}}$.
$D_{(n_1, n_2)}$	Energy cost of the shortest path between n_1 and n_2.

Analogously, the network infrastructure where the components can be evaluated is modeled with the multi-partite graph $\mathcal{G}_{\text{net}} = (\mathcal{V}_{\text{net}}, \mathcal{E}_{\text{net}})$ with vertex set $\mathcal{V}_{\text{net}}$ containing the communicating nodes, with cardinality $|\mathcal{V}_{\text{net}}| = M$, and edge set $\mathcal{E}_{\text{net}} \subset \{t_1 t_2 : t_i \in \mathcal{V}_{\text{net}}, n = 1, 2\}$ representing the wireless and wired links among nodes n_i. The result of the allocation is a matrix $X = \mathcal{V}_{\text{app}} \times \mathcal{V}_{\text{net}}$ where $X[t, n] = 1$ if and only if component t is allocated to node n. We also define E_d to be the device energy and E_n the network energy. E_t is then the total energy, and we put the constraint $E_d + E_n \leq E_t$. Components, nodes and links have properties that are relevant for the energy consumption of the application once allocated. These parameters are described in Table I. S_t, P_n, R_n and C_n are defined as multiples of some reference node. The resources of a node are expressed as a single scalar, but additional resource requirements can easily be introduced into the model.

2) Problem Formulation: For calculating the network energy, we need to know whether a link between two components is assigned to a link between two nodes. For this, we introduce a matrix $Y = \mathcal{V}_{\text{app}} \times \mathcal{V}_{\text{app}} \times \mathcal{V}_{\text{net}} \times \mathcal{V}_{\text{net}}$, where $Y[t_1, t_2, n_1, n_2] = 1$ if and only if the communication between component t_1 and component t_2 is allocated on the network link between nodes n_1 and n_2. This corresponds to $X[t_1, n_1] = 1 \land X[t_2, n_2]$. Unfortunately, this is not a linear constraint, and thus we need to linearize the formulation. For this, we follow the formulation presented in [52] and define an ILP model as:

$$\forall t_1, t_2 \in \mathcal{V}_{\text{app}} : \forall n_1, n_2 \in \mathcal{V}_{\text{net}} : Y[t_1, t_2, n_1, n_2] \leq X[t_1, n_1] \tag{3}$$

$$\forall t_1, t_2 \in \mathcal{V}_{\text{app}} : \forall n_1, n_2 \in \mathcal{V}_{\text{net}} : Y[t_1, t_2, n_1, n_2] \leq X[t_2, n_2] \tag{4}$$

$$\forall t_1, t_2 \in \mathcal{V}_{\text{app}} : \forall n_1, n_2 \in \mathcal{V}_{\text{net}} : Y[t_1, t_2, n_1, n_2]$$
$$\geq X[t_1, n_1] + X[t_2, n_2] - 1 \tag{5}$$

$$\forall t \in \mathcal{V}_{\text{app}} : \sum_{n \in \mathcal{V}_{\text{net}}} X[t, n] = 1 \tag{6}$$

$$\forall n \in \mathcal{V}_{\text{net}} : \sum_{t \in \mathcal{V}_{\text{app}}} X[t, n] \cdot R_t \leq R_n \tag{7}$$

$$\sum_{t \in \mathcal{V}_{\text{app}}} \sum_{n \in \mathcal{V}_{\text{net}}} C_n \cdot (S_t / P_n) \cdot X[t, n] \leq E_d \tag{8}$$

$$\sum_{(t_1, t_2) \in \mathcal{E}_{\text{app}}} \sum_{n_1, n_2 \in \mathcal{V}_{\text{net}}} O_{n_1} \cdot P_{n_1, n_2} \cdot Y[t_1, t_2, n_1, n_2] \leq E_n \tag{9}$$

$$E_n + E_d \leq E_t \tag{10}$$

where equations (3) to (5) describe the linearization of the network matrix Y. Equations (6) and (7) are for ensuring that components are allocated only once and that resources are not exceeded, respectively. Equations (8) and (9) calculate network and device energy as described above. Finally, we calculate the total energy use of the assignment by adding both energies in (10). The objective of the optimization is the minimization of the total used energy.

3) A Linear Heuristic for Energy-Optimized Allocation: The presented QAP is NP-hard and thus compute intensive. The culprit for this is the network cost calculation and the linearization of Y resulting in a large number of constraints. By removing the Y matrix and the associated constraints, we create a linear problem that can be evaluated effectively by the simplex method [72]. The approach approximates the energy required for sending a packet of data by taking the average of a node's links. We introduce the parameter $\hat{T}_n = \frac{1}{|\text{outgoing}(n)|} \sum_{e \in \text{outgoing}(n)} T_e$ that describes the average transmission cost of a node's links.

$$\sum_{t \in \mathcal{V}_{\text{app}}} \sum_{n \in \mathcal{V}_{\text{net}}} C_n \cdot (S_t / P_n) \cdot X[t, n] + O_t \cdot \hat{T}_n \cdot X[t, n] \leq E_t \tag{11}$$

The complete model reuses constraints in equations (6) and (7) with the constraint (11). By transforming the QAP into a linear problem, we greatly increase the speed of finding a solution, and make the optimization feasible for on-line usage. The drawback is that by approximating the network energy the solution is no longer optimal, as it will be shown in the results.

4) Evaluation of Allocation Algorithm: We implemented the model using the PuLP[3] linear programming library. The evaluation was done by generating a random network and a random application, and letting the solver find the optimal allocation.
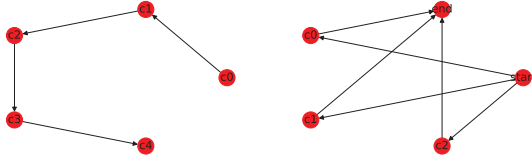
[3]https://pythonhosted.org/PuLP/

Fig. 5. "Long" (left) and "wide" (right) composed IoT applications.

The network configuration is generated with a variety of node configurations and capabilities, reflecting a heterogeneous computation and communication infrastructure that one could find in an industrial manufacturing plant (e.g., using Siemens range of industrial computers [73]). In the evaluated configuration, 60% of the nodes were generated as wired nodes, and the remaining 40% are wireless nodes. Nodes are connected to each other with a certain probability. That probability is 0.8 for wired-wired connections, 0.5 for wireless-wireless connections and 0.4 for wireless-wired connections. Wired connections use 0.2 units of energy, while wireless connections use 0.8 units of energy, which is similar to the power consumption of an Ethernet module [74] as compared to a WiFi module [75]. Nodes have a varying amount of memory resources uniformly distributed between a lower bound of 1 and an upper bound of 8 resource units. Nodes also have a varying processing speed between 1 and 3 speedup, roughly comparing to the Intel processor family i3, i5, and i7. Finally, nodes can use from 0.5 to 1.5 units of energy for a single unit of computation.

For the application, two classes with a certain number of components are generated, a "wide" and a "long" application. In a "wide" application, two components are designated the "start" and "end" components, and every other component needs input from the start node and sends output to the end node. In a long application, components are linked serially. Figure 5 shows two example applications. This method for generating recips is similar to [52]. Each application component has resource requirements randomly distributed between 1 and 8, an output factor randomly distributed between 0.5 and 1.5, and a computation size of 1 or 2.

As expected, the optimal allocation algorithm scales very badly (non-polynomially). Figure 6 shows the runtime of the algorithm for varying problem sizes. The shaded area shows the variance with the non-shown parameter (different application sizes for the network node graph, differing network sizes for the application node graph). The time needed for finding the optimal allocation grows unwieldy very quickly.

In comparison, the heuristic presented in equation (11) finds a solution much more quickly. Figure 7 shows the runtime of the heuristic for different network and application sizes. For the slowest case for the full allocation, the heuristic takes 8 seconds of CPU time, while the solver consumes 864104 seconds (about 10 days) of CPU time for finding the optimal allocation. The allocation evaluation was executed on an Amazon EC2 `m4.10xlarge` machine with 40 virtual cores and 160 GiB of memory. Peak memory use was 51 GiB. However, the heuristic loses about 30% of energy efficiency over the optimal algorithm. Specifically, 50% of the solutions
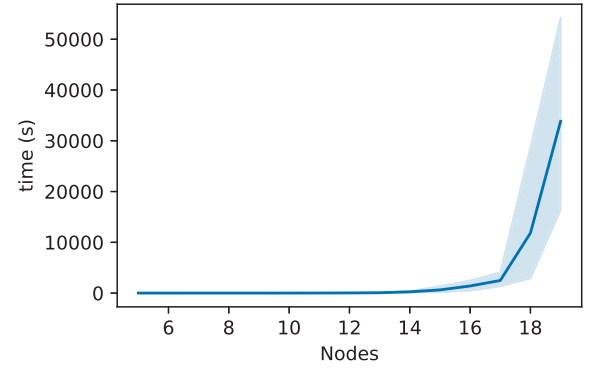
(a) CPU time of the optimal allocation algorithm vs. the number of nodes. Each experiment with n nodes was measured 5 times with 3 to n-1 components.
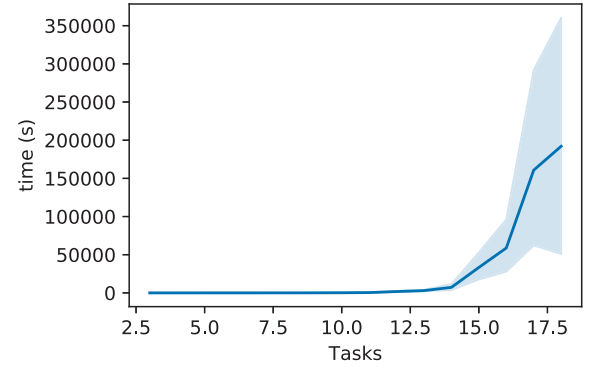
(b) CPU time of the optimal allocation algorithm vs. the number of components. Each experiment with n components was measured 5 times with 5 to 20 nodes.

Fig. 6. Runtime for optimal allocation.

achieve between 60% and 80% of the energy efficiency of the optimal case.

C. Energy-Efficient Blockchain Over Wireless Networks

The last enabling technology is DLT, which provides a tamper-proof ledger distributed for the nodes of the iIoTe. The energy and latency cost of implementing DLT over wireless links and with constrained IoT devices is oftentimes overlooked. In general, the latency and energy budgets are highly impacted by the wireless access protocol.

1) System Model: As introduced in [65], there are two architectural choices for IoT DLT. The conventional one is to have IoT devices that receive complete blocks from the Blockchain to which they are connected, and locally verify the validity of the Proof-of-Work (PoW) solution and the contained transactions. This configuration provides the maximum possible level of security. However, this requires high storage, energy and computation resources, since the node needs to store the complete Blockhain and to check all transactions. This makes it infeasible for many IoT applications. Instead, we consider the second option where the IoT device is a *light node* that receives only the headers from the Blockchain nodes. These headers contain sufficient information for the
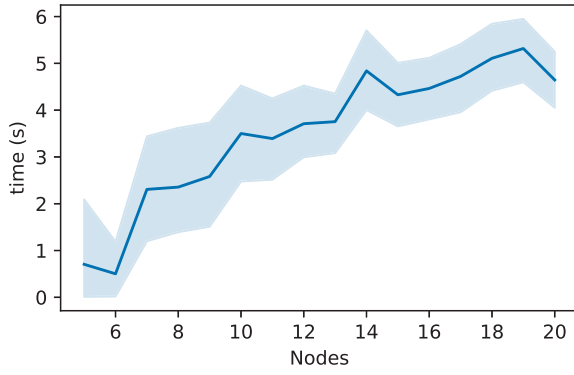
(a) CPU time of the allocation heuristic vs. the number of nodes. Each experiment with n nodes was measured 5 times with 3 to n-1 components.
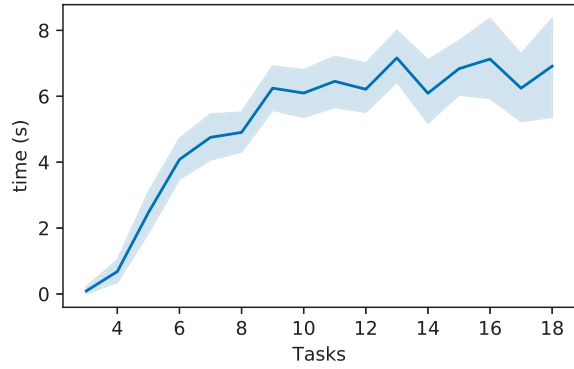
(b) CPU time of the allocation heuristic vs. the number of components. Each experiment with n components was measured 5 times with 5 to 20 nodes.

Fig. 7. Heuristic runtime.

Proof-of-Inclusion (PoI), i.e., to prove the inclusion of a transaction in the block without the need to download the entire block body. Furthermore, the device defines a list of (few) events of interest, such as modifications to the state of a smart contract or transactions from/to a particular address.

The communication model for this *lightweight* version is as follows. The IoT devices transmit data to the Blockchain using the edge infrastructure. Specifically, a NB-IoT cell with the base station located in its center is considered, with N devices uniformly distributed within the area. The base station, which is designated as a *full* DLT node connected to the Blockchain, is the DLT-anchor for the IoT devices. For the radio resource management, we adapt the queueing model of [19] to our scenario, where the uplink and downlink radio resources are modeled as two servers that visit and serve their respective inter-dependent traffic queues.

2) End-2-End (E2E) Latency: NB-IoT provides three coverage classes namely normal, extreme, and robust class for serving limited-resource devices and suffering various pathloss levels[76]. Minimum latency and throughput requirements need to be maintained in the extreme coverage class, whereas enhanced performance is ensured in the extended or normal coverage class. Without loss of generality, we consider only normal and extreme coverage class, i.e., the number of classes

$C = 2$. A class is assigned to a device based on the estimated path loss, with the base station informing the assigned device of the dedicated path between them. Class j and $\forall_j$ are supported by the replicas number c_j, which are transmitted based on data and the control packet [19]. Particularly, the reserved NPRACH period of class j is denoted by $c_j\tau$. The unit length τ of the NPRACH for the class of coverage is denoted by $c_j = 1$. t_j is the average time interval between two consecutive scheduling of NPRACH of class j, whereas the average time duration between two consecutive NPDCCH occurrences is denoted by d.

The total E2E latency includes two parts: 1) the latency L_{UeD} of transmissions of uplink and downlink between IoT devices and the base station (the wireless communication latency) and 2) and the latency L_{DLT} due to the DLT verification process. i.e., $L = L_{UeD} + L_{DLT}$.

The wireless communication latency of NB-IoT uplink and downlink can be formulated as:

$$L_{UeD} = L^u + L^d = L^u_{sync} + L^u_{rr} + L^u_{tx} + L^d_{sync}$$
$$+ L^d_{rr} + L^d_{rx}, \tag{12}$$

where L^u_{sync}, L^u_{rr}, L^u_{tx}, L^d_{synch}, L^d_{rr}, and L^d_{rx} are energy consumption of synchronization, resource reservation, and data transmission of uplink and downlink, respectively. L^u_{sync} has been defined in [77] with the values of $0.33s$. L_{rr} is given as:

$$L_{rr} = \sum_{l=1}^{N_{r_{max}}} (1 - P_{rr})^{l-1} P_{rr} l (L_{ra} + L_{rar}), \tag{13}$$

in which, $N_{r_{max}}$ is the maximum number of attempts, P_{rr} is the probability of successful resource reservation in an attempt, $L_{ra} = 0.5t + \tau$, is the expected latency in sending an RA control message, τ is the unit length and equal to the NPRACH period for the coverage class 1 which is varied from $40\,\text{ms}$ to $2.56\,\text{s}$ [77], and $L_{rar} = 0.5d + 0.5Qfu + u$, is the expected latency in receiving the RAR message, where Q are requests waiting to be served.

In the following, we provide a simple technique based on *drift approximation* [78] to calculate P_{rr} recursively. Therefore, we treat the mean of the random variables involved in the process as constants. Besides, we assume that sufficient resources are available in the NPDCCH so that failures only occur due to collisions in the NPRACH or to link outages.

Let $\lambda^a = \lambda^u + \lambda^d$ be the arrival rate of access requests per NPRACH period and $\lambda^a(l)$ be the mean number of devices participating in the contention with their l-th attempt. Note that in the steady state $\lambda^a(l)$ remains constant for all NPRACH periods. Next, let $\lambda^a_{tot} = \sum_{l=1}^{N_{r_{max}}} \lambda^a(l)$. The collision probability in the NPRACH can be calculated using the drift approximation for a given value of λ^a_{tot} and for a given number of available preambles K as:

$$P_{\text{collision}}(\lambda^a_{tot}) = 1 - \left(1 - \frac{1}{K}\right)^{\lambda^a_{tot} - 1} \approx 1 - e^{-\frac{\lambda^a_{tot}}{K}}. \tag{14}$$

From there, we approximate the probability of resource reservation as a function of λ^a_{tot} as $P_{rr}(\lambda^a_{tot}) \approx p_d\, e^{-\frac{\lambda^a_{tot}}{K}}$. This

allows us to define λ_{tot}^a as:

$$\lambda_{tot}^a = \lambda^a + (1 - P_{rr}(\lambda_{tot}^a)) \sum_{l=2}^{N_{r_{max}}} \lambda^a(l), \qquad (15)$$

since $\lambda^a(l) = (1 - P_{rr}(\lambda_{tot}^a))\lambda^a(l-1)$ for $l \geq 2$ and $\lambda^a(1) = \lambda^a$. Finally, from the initial conditions $\lambda^a(l) = 0$ for $l \geq 2$, the values of $\lambda^a(l)$ and λ_{tot}^a can be calculated recursively by: 1) applying (15); 2) calculating $P_{rr}(\lambda_{tot}^a)$ for the new value of λ_{tot}^a; and 3) updating the values of $\lambda^a(l)$. This process is repeated until the values of the variables converge to a constant value. The final value of $P_{rr}(\lambda_{tot}^a)$ is simply denoted as P_{rr} and used throughout the rest of the paper.

Assuming that the transmission time for the uplink transactions follows a general distribution with the first two moments l_1, l_2, the first two moments of the distribution of the packet transmission time are $s_1 = (f_1 l_1)/(\mathcal{R}w)$, and $s_2 = (f_1 l_2)/(\mathcal{R}^2 w^2)$. Applying the results from [79], considering L_{tx} as a function of scheduling of NPUSCH, we have:

$$L_{tx} = \frac{f\lambda^u s_1 s_2}{2s_1(1 - fGs_1)} + \frac{f\lambda^u s_1^2}{2(1 - f\lambda^u s_1)} + \frac{l_1}{\mathcal{R}^u w}, \quad (16)$$

where $\mathcal{R}^u$ is the average uplink transmission rate, $\lambda^u = \lambda_s + \lambda_b$, and $f(\lambda_s + \lambda_b)s_1$ is the mean batch-size. The latency of data reception is defined as:

$$L_{rx} = \frac{0.5Fh_1 t^{-1}}{h_1(1 - Fht^{-1})} + \frac{Fh_1}{1 - Fht^{-1}} + \frac{m_2}{\mathcal{R}^d y}, \quad (17)$$

in which, $h_1 = fm_1(\mathcal{R}^d y)^{-1}$, $h_2 = fm_2((\mathcal{R}^d)^2 y^2)^{-1}$ are two moments of distribution of the packet transmission time, assuming that the packet length follows a general distribution with moments m_1, m_2, $F = f\lambda^d t$, $\mathcal{R}^d$ is downlink data transmission rate.

Next, we calculate the second latency component, corresponding to the DLT verification process. Consider a DLT network that includes M miners. These miners start their Proof-of-Work (PoW) computation at the same time and keep executing the PoW process until one of the miners completes the computational task by finding the desired hash value [56]. When a miner executes the computational task for the POW of current block, the time period required to complete this PoW can be formulated as an exponential random variable W whose distribution is $f_W(w) = \lambda_c e^{-\lambda_c w}$, in which $\lambda_c = \lambda_0 P_c$ represents the computing speed of a miner, P_c is power consumption for computation of a miner, and λ_0 is a constant scaling factor. Once a miner completes its PoW, it will broadcast messages to other miners, so that other miners can stop their PoW and synchronize the new block.

$$L_{tM} = L_{newB} + L_{getB} + L_{transB} \qquad (18)$$

In (18), L_{newB}, L_{getB}, and L_{transB}, are latencies of sending hash of new mined block, requesting new block from neighboring nodes, and new block transmission, respectively. L_{newB} and L_{transB} are computed using uplink transmission, while L_{getB} is computed based on downlink transmission as described in previous section.

For the PoW computation, a miner $i*$, first finds out the desired PoW hash value, $i* = \min_{i \in M} w_i$. The fastest PoW computation among miners is W_{i*}, the complementary cumulative probability distribution of W_{i*} could be computed as $Pr(W_{i*} > x) = Pr(\min_{i \in M}(W_i) > x) = \prod_{i=1}^H Pr(W_i > x) = (1 - Pr(W < x))^M$. Hence, the average computational latency of miner $i*$ is described as:

$$L_{W_{i*}} = \int_0^\infty (1 - Pr(W \leq x))^M DDx$$
$$= \int_0^\infty e^{-\lambda_c Mx} DDx = \frac{1}{\lambda_c M} \qquad (19)$$

The total latency required from DLT verification process is $L_{DLT} = L_{tm} + L_{W_{i*}}$.

3) Energy Consumption: Analogously to the latency, the energy consumption is divided in the wireless communication (uplink/downlink) and the DLT verification.

The total energy consumption in the wireless communication is written as follows:

$$\begin{aligned} E_{UD} &= E^u + E^d \\ &= E_{sync}^u + E_{rr}^u + E_{tx}^u + E_s^u + E_{sync}^d + E_{rr}^d \\ &\quad + E_{rx}^d + E_s^d, \end{aligned} \qquad (20)$$

in which, E_{sync}^u, E_{rr}^u, E_{rr}^u, E_{sync}^d, E_{rr}^d, and E_{rx}^d are energy consumption of synchronization, resource reservation, and data transmission of uplink and downlink, respectively. Each of them are formally defined as follows:

$$E_{sync} = P_l \cdot L_{sync} \qquad (21)$$
$$E_{rar} = P_l \cdot L_{rar} \qquad (22)$$
$$E_{rr} = \sum_{l=1}^{N_{max}} (1 - P_{rr})^{l-1} \cdot P_{rr} \cdot (E_{ra} + E_{rar}) \qquad (23)$$
$$E_{ra} = (L_{ra} - \tau) \cdot P_I + \tau \cdot (P_c + P_e P_t) \qquad (24)$$
$$E_{tx} = \left(L_{tx} - \frac{l_a}{\mathcal{R}^u w}\right) \cdot P_I + (P_c + P_e P_t)\frac{l_a}{\mathcal{R}^u w} \qquad (25)$$
$$E_{rx} = \left(L_{rx} - \frac{m_1}{\mathcal{R}^d y}\right) \cdot P_I + P_l \frac{m_1}{\mathcal{R}^d y} \qquad (26)$$

where P_e, P_I, P_c, P_l, and P_t are the power amplifier efficiency, idle power consumption, circuit power consumption of transmission, listening power consumption, and transmit power consumption, respectively.

Following the PoW described above, the average energy consumption of DLT to finish a single PoW round is:

$$E_{DLT} = P_c L_{W_{i*}} + P_t L_{tm}. \qquad (27)$$

4) Results: The performance of DLT-based NB-IoT system is shown in Fig. 8. The experiments demonstrate the total latency Fig. 8(a) and energy efficient Fig. 8(b) of a DLT-based NB-IoT system, respectively. In Fig. 8(a), the E2E latency is defined as the time elapsed from the generation of a transaction at the NB-IoT device until its verification. This includes the latency at the NB-IoT radio link and at the DLT, which comprises the execution time of the smart contract and transaction verification. We observe that increasing t and d values at the first increases lifetime and decrease latency due to more

(a) End-to-End Latency (sec)
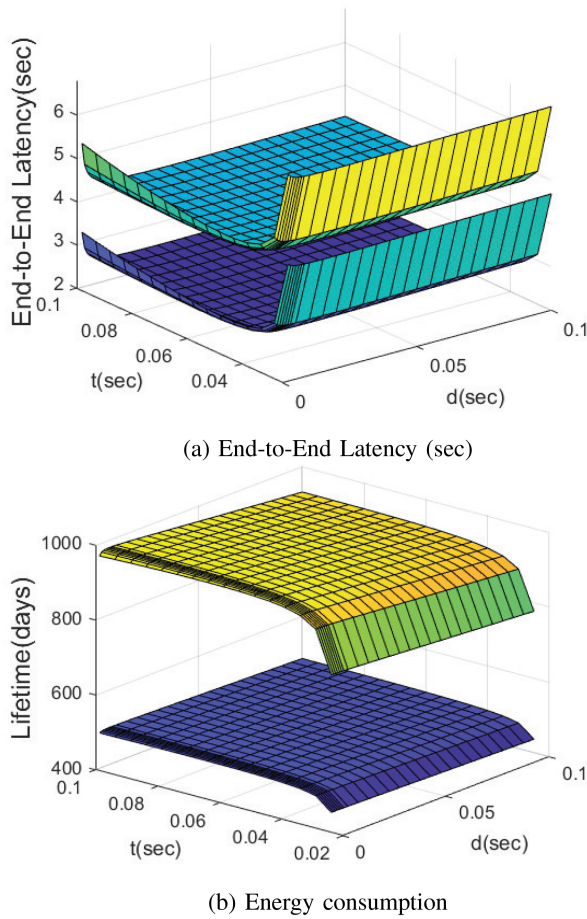

(b) Energy consumption

Fig. 8.   Latency and Energy Consumption.

resources for NPUSCH and NPDSCH, but after certain point increasing t and d decreases the lifetime by increasing the expected time for resource reservation. In comparison with the standard NB-IoT system in [55], [76], the DLT-based system introduces a slight latency because of addition time of consensus process and transaction verification. This is a latency and security trade-off between standard NB-IoT and DLT-based systems.

IV. TOWARDS ENERGY-EFFICIENT INTELLIGENT IoT ENVIRONMENTS

Having the energy-performance characterization for each of the enabling technologies (Section III), we describe next how they interact with each other in iIoTe. For this, we consider the scenario in Figure 9, where a given learning application is considered. We split the task into sub-tasks such as data processing, data training and model aggregation and distribute them in a decentralized way. Each of these sub-tasks (Section III-A) constitute the application components $(C_1, C_2, \ldots,)$ that can be run at the available edge nodes. The optimal allocation of sub-tasks to edge nodes is determined using the ILP-based algorithms presented in Section III-B. The required trustworthiness (i.e., assuring security, privacy, immutability and transparency) between sub-tasks is provided through DLT (Section III-C). The heterogeneity of

devices, capabilities and tasks is exploited accordingly: The edge servers with high computation capability are selected to operate the DLT activities, e.g., block mining, and aggregate the ML models (the head workers if the learning paradigms in Section III-A are applied), while more constrained edge devices or mobile devices are setup as DLT light clients that can participate in local training (the tail workers) and consensus. The involved network components can communicate via wireless long-range communication NB-IoT channels.

In detail, the communication workflow of the proposed scheme can be summarized as follows:

- *Step 1:* The data processing can be completed in different edge devices with limited resources. The selected data from the data provider is pre-processed and structured. This process includes both a data engineering and feature engineering sub-process, in which data engineering converts the raw data into prepared data and feature engineering tunes the prepared data to create features expected by ML models.
- *Step 2:* Then, the edge nodes or IoT devices, which are responsible for training, compute the local model based on its own private data and then publish the local model to its associated edge server via, e.g., NB-IoT by registering with active smart contracts to upload their result securely until the results are incorporated in the final aggregation and generation of DLT transactions.
- *Step 3:* Next, the edge servers with ML aggregation responsibility gather transactions and arrange them in blocks following the Merkle tree. The structure of a DLT involves the hash of the previous block, a timestamp, 'nonce' and the structure of hash tree. These edge servers with high computational capacity join in the DLT mining process to verify the created blocks and operate consensus in the edge network. After completing the mining process, the verified blocks are added to the ledger, and synchronized among the nodes. The local models are published in the distributed ledger. Hence, the powerful edge servers can compute the global model directly based on the aggregation rules defined in smart contracts.

The advantages of this integration are two-folds. First, by distributing the tasks to different edge nodes with different computing capacities, the IoT devices or edge nodes with limited resources can save significant amount of energy required for training or mining and they can achieve lower latency. Second, by leveraging the DLT, the updates of ML models are securely formed in encrypted transactions and hashed blocks, which significantly enhances the security and privacy of distributed learning in the edge networks. The DLT provides trust, transparency and immutability baseline for distributed learning to guarantee the security and privacy of data and ML models, and naturally addresses the single-point of failure problem of the current standard FL approach that relies on a centralized server to aggregate the models. Although the integration of enabling technologies introduces advantages, it also has some drawbacks, for example, the time required of DLT mining will increase the total latency of the system. This is a trade-off between trust and communication latency which we discussed in [55], [65].
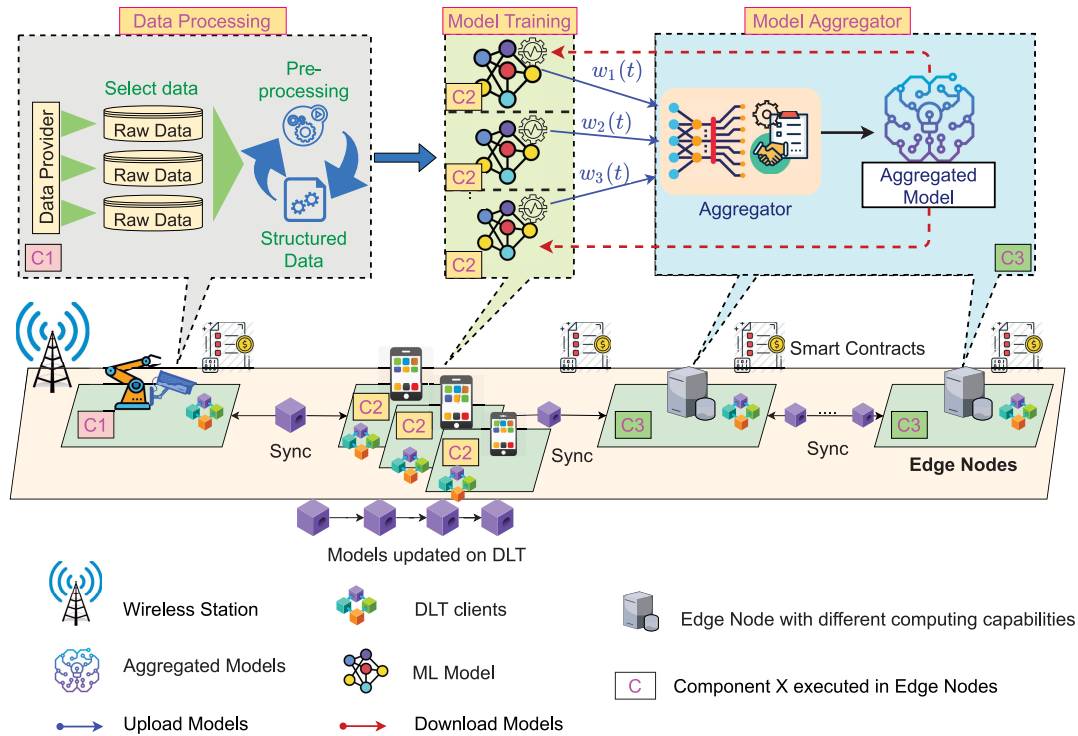
Fig. 9. Integration of enabling technologies.

V. Conclusion and Future Work

In this paper, we address the evolution of next-generation of IoT networks towards the edge, driven by the introduced intelligent IoT environments. We use the iIoTe as the basic building block to characterize the tradeoff energy-performance of the three key enabling technologies, learning, edge computing and distributed ledger. Edge intelligence must rely on distributed paradigms such as FL, and we have shown how exploiting spatial and temporal sparsity and quantization can significantly improve the performance and reduce the energy consumption. Moreover, we have discussed the distribution of the FL model aggregator and the rest of sub-tasks to make the framework more robust against failures. For edge computing, the optimal allocation of the application components to network resources is important to efficiently use the available infrastructure and optimize its energy consumption. DLT is a flexible solution for trustworthiness in these environments, but the energy and latency cost of implementing DLT over wireless and constrained devices is oftentimes overlooked. We have analyzed these parameters using NB-IoT as the baseline wireless technology.

In the integration of these technologies in iIoTe, we have shown the interactions among them, which provides the basis towards an energy model and evaluation that encompasses the contribution of each element. For instance, the learning and computation models can be easily broaden to consider the allocation of the different sub-tasks of the learning application in a representative topology, with each learning action and resource allocation playing the role of an action to be recorded in the DLT. Future work also includes extending the proposed solutions to dynamic environments where agents move and edge nodes are not always available. This is already supported by the presented dynamic *head/tail* learning paradigms but the integration of a dynamic resource allocation and DLT framework is pending. Another necessary direction is to investigate the joint optimization of the computing and communication resources from the energy perspective.

References

[1] L. Hou *et al.*, "Internet of Things cloud: Architecture and implementation," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 32–39, Dec. 2016.

[2] A. Alnoman, S. K. Sharma, W. Ejaz, and A. Anpalagan, "Emerging edge computing technologies for distributed IoT systems," *IEEE Netw.*, vol. 33, no. 6, pp. 140–147, Nov./Dec. 2019.

[3] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7457–7469, Aug. 2020.

[4] L. M. Hilty and B. Aebischer, *ICT Innovations for Sustainability*, vol. 310. Cham, Switzerland: Springer, 2015.

[5] "Lean ICT—Towards digital sobriety," Shift Project, Paris, France, Rep., 2019.

[6] D. Bol, G. de Streel, and D. Flandre, "Can we connect trillions of IoT sensors in a sustainable way? A technology/circuit perspective (invited)," in *Proc. IEEE SOI-3D-Subthreshold Microelectron. Technol. Unified Conf. (S3S)*, 2015, pp. 1–3.

[7] W. Yu *et al.*, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[8] N. Hassan, S. Gillani, E. Ahmed, I. Yaqoob, and M. Imran, "The role of edge computing in Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 110–115, Nov. 2018.

[9] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, and A. Neal, "Mobile-edge computing introductory technical white paper," Sophia Antipolis, France, ETSI, White Paper, 2014.

[10] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[11] M. S. Elbamby, M. Bennis, W. Saad, M. Latva-Aho, and C. S. Hong, "Proactive edge computing in fog networks with latency and reliability guarantees," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, p. 2019, 2018.

[12] M. S. Elbamby *et al.*, "Wireless edge computing with latency and reliability guarantees," *Proc. IEEE*, vol. 107, no. 8, pp. 1717–1737, Aug. 2019.

[13] Y. Hu and A. Schmeink, "Delay-constrained communication in edge computing networks," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2018, pp. 1–5.

[14] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.

[15] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan./Feb. 2018.

[16] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[17] M. Kanj, V. Savaux, and M. Le Guen, "A tutorial on NB-IoT physical layer design," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2408–2446, 4th Quart., 2020.

[18] Y.-P. E. Wang *et al.*, "A primer on 3GPP narrowband Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 117–123, Mar. 2017.

[19] A. Azari, Č. Stefanović, P. Popovski, and C. Cavdar, "On the latency-energy performance of NB-IoT systems in providing wide-area iot connectivity," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 1, pp. 57–68, Mar. 2020.

[20] "Radio resource control (RRC); protocol specification, v9. 3.0," 3GPP Standard TS 36. 331, 2010.

[21] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NIPS Workshop PMPML*, Barcelona, Spain, Dec. 2016. [Online]. Available: https://arxiv.org/abs/1610.05492

[22] T. Yang *et al.*, "Applied federated learning: Improving google keyboard query suggestions," 2018, *arXiv:1812.02903*.

[23] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 1146–1159, Feb. 2020.

[24] P. Kairouz *et al.*, "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[25] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. NIPS*, Dec. 2017, pp. 4424–4434.

[26] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," 2019, *arXiv: 1901.11173*.

[27] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," Jun. 2018, *arXiv::1806.00582*.

[28] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.

[29] Y. Zhao *et al.*, "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb. 2021.

[30] Y. Zhao *et al.*, "Local differential privacy-based federated learning for Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8836–8853, Jun. 2021.

[31] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020, *arXiv:2003.02133*.

[32] M. Yang, L. Lyu, J. Zhao, T. Zhu, and K.-Y. Lam, "Local differential privacy and its applications: A comprehensive survey," 2020, *arXiv:2008.03686*.

[33] Y. Koda, K. Yamamoto, T. Nishio, and M. Morikura, "Differentially private aircomp federated learning with power adaptation harnessing receiver noise," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6.

[34] A. Elgabli, J. Park, C. B. Issaid, and M. Bennis, "Harnessing wireless channels for scalable and privacy-preserving federated learning," *IEEE Trans. Commun.*, vol. 69, no. 8, pp. 5194–5208, Aug. 2021.

[35] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.

[36] M. M. Wadu, S. Samarakoon, and M. Bennis, "Federated learning under channel uncertainty: Joint client scheduling and resource allocation," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2020, pp. 1–6.

[37] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*.

[38] M. M. Amiri and D. Gündüz, "Computation scheduling for distributed machine learning with straggling workers," *IEEE Trans. Signal Process.*, vol. 67, no. 24, pp. 6270–6284, Dec. 2019.

[39] M. Polese, R. Jana, V. Kounev, K. Zhang, S. Deb, and M. Zorzi, "Machine learning at the edge: A data-driven architecture with applications to 5G cellular networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 12, pp. 3367–3382, Dec. 2021.

[40] T. Chen, G. B. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," 2018, *arXiv:1805.09965*.

[41] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 592–606, Mar. 2012.

[42] T. AlShammari, S. Samarakoon, A. Elgabli, and M. Bennis, "BayGo: Joint Bayesian learning and information-aware graph optimization," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2021, pp. 1–6.

[43] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar, "MATCHA: Speeding up decentralized SGD via matching decomposition sampling," in *Proc. 6th Indian Control Conf. (ICC)*, 2019, pp. 299–300.

[44] A. Elgabli, J. Park, A. S. Bedi, M. Bennis, and V. Aggarwal, "GADMM: Fast and communication efficient framework for distributed machine learning," *J. Mach. Learn. Res.*, vol. 21, no. 76, pp. 1–39, 2020.

[45] C. B. Issaid, A. Elgabli, J. Park, and M. Bennis, "Communication efficient distributed learning with censored, quantized, and generalized group ADMM," 2020, *arXiv:2009.06459*.

[46] G. T. Lakshmanan, Y. Li, and R. Strom, "Placement strategies for Internet-scale data stream systems," *IEEE Internet Comput.*, vol. 12, no. 6, pp. 50–60, Nov./Dec. 2008.

[47] A. M. Haubenwaller and K. Vandikas, "Computations on the edge in the Internet of Things," *Procedia Computer Science*, vol. 52, pp. 29–34, Jan. 2015.

[48] F. Samie, V. Tsoutsouras, L. Bauer, S. Xydis, D. Soudris, and J. Henkel, "Computation offloading and resource allocation for low-power IoT edge devices," in *Proc. IEEE 3rd World Forum Internet Things (WF-IoT)*, 2016, pp. 7–12.

[49] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[50] Y. Sahni, J. Cao, S. Zhang, and L. Yang, "Edge mesh: A new paradigm to enable distributed intelligence in Internet of Things," *IEEE Access*, vol. 5, pp. 16441–16458, 2017.

[51] N. Mohan and J. Kangasharju, "Edge-fog cloud: A distributed cloud for Internet of Things computations," in *Proc. Cloudification Internet Things (CIoT)*, 2016, pp. 1–6.

[52] V. Cardellini, V. Grassi, F. L. Presti, and M. Nardelli, "Optimal operator placement for distributed stream processing applications," in *Proc. 10th ACM Int. Conf. Distrib. Event-Based Syst.*, 2016, pp. 69–80.

[53] J. Seeger, A. Bröring, and G. Carle, "Optimally self-healing IoT choreographies," *ACM Trans. Internet Technol.*, vol. 20, no. 3, pp. 1–20, 2020.

[54] T. M. Fernández-Caramés and P. Fraga-Lamas, "A review on the use of blockchain for the Internet of Things," *IEEE Access*, vol. 6, pp. 32979–33001, 2018.

[55] L. D. Nguyen, A. E. Kalor, I. Leyva-Mayorga, and P. Popovski, "Trusted wireless monitoring based on distributed ledgers over NB-IoT connectivity," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 77–83, Jun. 2020.

[56] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Bus. Rev.*, Seoul, South Korea, Rep., 2008, Art. no. 21260.

[57] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Zug, Switzerland, Ethereum Project, Yellow Paper, 2014, pp. 1–32, 2014.

[58] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[59] L. D. Nguyen, I. Leyva-Mayorga, A. N. Lewis, and P. Popovski, "Modeling and analysis of data trading on blockchain-based market in IoT networks," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6487–6497, Apr. 2021.

[60] C. Esposito, A. De Santis, G. Tortora, H. Chang, and K.-K. R. Choo, "Blockchain: A Panacea for healthcare cloud-based data security and privacy?" *IEEE Cloud Comput.*, vol. 5, no. 1, pp. 31–37, Jan./Feb. 2018.

[61] K. N. Griggs, O. Ossipova, C. P. Kohlios, A. N. Baccarini, E. A. Howson, and T. Hayajneh, "Healthcare blockchain system using smart contracts for secure automated remote patient monitoring," *J. Med. Syst.*, vol. 42, no. 7, pp. 1–7, 2018.

[62] S. Saberi, M. Kouhizadeh, J. Sarkis, and L. Shen, "Blockchain technology and its relationships to sustainable supply chain management," *Int. J. Prod. Res.*, vol. 57, no. 7, pp. 2117–2135, 2019.

[63] Y. Zhang, X. Xu, A. Liu, Q. Lu, L. Xu, and F. Tao, "Blockchain-based trust mechanism for IoT-based smart manufacturing system," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1386–1394, Dec. 2019.

[64] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1495–1505, Apr. 2019.

[65] P. Danzi *et al.*, "Communication aspects of the integration of wireless IoT devices with distributed ledger technology," *IEEE Netw.*, vol. 34, no. 1, pp. 47–53, Jan./Feb. 2020.

[66] S. Boyd, N. Parikh, and E. Chu, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Hanover, MA, USA: Now, 2011.

[67] A. Elgabli, J. Park, A. S. Bedi, C. B. Issaid, M. Bennis, and V. Aggarwal, "Q-GADMM: Quantized group ADMM for communication efficient decentralized machine learning," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 164–181, Jan. 2021.

[68] D. Dua and C. Graff, 2017, "UCI machine learning repository (2020)," data set.

[69] M. M. Amiri and D. Gündüz, "Over-the-air machine learning at the wireless edge," in *Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2019, pp. 1–5.

[70] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 491–506, Jan. 2020.

[71] T. Sery and K. Cohen, "On analog gradient descent learning over multiple access fading channels," *IEEE Trans. Signal Process.*, vol. 68, pp. 2897–2911, 2020, doi: 10.1109/TSP.2020.2989580.

[72] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, pp. 308–313, Jan. 1965.

[73] "SIMATIC Box IPC." Siemens, 2021. [Online]. Available: new.siemens.com

[74] *ENC28J60—Stand-Alone Ethernet Controller with SPI Interface*, MICROCHIP, Chandler, AZ, USA, 2012.

[75] S. Chiaravalloti, F. Idzikowski, and Ł. Budzisz, "Power consumption of WLAN network elements," TKN, Berlin, Germany, TKN Rep. TKN-11-002, 2011.

[76] S. Popli, R. K. Jha, and S. Jain, "A survey on energy efficient narrowband Internet of Things (NBIoT): Architecture, application and challenges," *IEEE Access*, vol. 7, pp. 16739–16776, 2018.

[77] O. Liberg, M. Sundberg, E. Wang, J. Bergman, and J. Sachs, *Cellular Internet of Things: Technologies, Standards, and Performance*. Amsterdam, The Netherlands: Academic, 2017.

[78] C.-H. Wei, G. Bianchi, and R.-G. Cheng, "Modeling and analysis of random access channels with bursty arrivals in OFDMA wireless networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 4, pp. 1940–1953, Apr. 2015.

[79] H. Akimaru and K. Kawashima, *Teletraffic: Theory and Applications*. London, U.K.: Springer, Dec. 2012.

Lam D. Nguyen (Graduate Student Member, IEEE) received the bachelor's degree in telecommunication from the Hanoi University of Science and Technologies in 2015, and the master's degree in computer science from Seoul National University in 2019. He is currently pursuing the Ph.D. degree with Aalborg University. His research includes distributed systems, blockchain, smart contracts, the Internet of Things, and applying blockchain and federated learning to enhance the efficiency of blockchain-based IoT monitoring networks. He receives the Outstanding Paper Award for the research about scaling Blockchain in Massive IoT at the IEEE World Forum Internet of Things 2020, the Travel Grant from Linux Foundation 2020, the Best Research Award for a solution of Blockchain-Based CO_2 Emission Trading from VEHITS 2021. He is a Hyperledger Member and an IEEE ComSoc Student Member.

Jan Seeger received the M.Sc. degree in computer science from TU Munich. He is currently working as an Engineer with Siemens Technology, Munich, Germany. He is active in the areas of Internet of Things and automation research, and how semantic technologies can improve the engineering of automation systems.
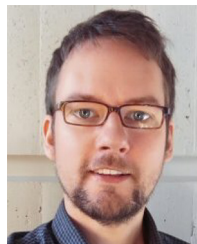
Arne Bröring received the Ph.D. degree from the University of Twente, The Netherlands, in 2012. He is a Senior Key Expert Research Scientist with Siemens Technology, Munich. He has contributed to over 90 publications in the field of distributed systems and has served on various program committees and editorial boards. At Siemens, he has been in charge of the technical and scientific coordination of large EU research projects (BIG IoT and IntellIoT). Before joining Siemens, he worked for the Environmental Systems Research Institute, Zurich, the 52° North Open Source Initiative, and led the Sensor Web and Simulation Lab, University of Münster. His research interests range from distributed system designs, over sensor networks, and semantic Web, to the Internet of Things.

Chaouki Ben Issaid (Member, IEEE) received the Diplôme d'Ingénieur degree in economics and financial engineering from the École Polytechnique de Tunisie in 2013, and the master's degree in applied mathematics and computational science and the Ph.D. degree in statistics from the King Abdullah University of Science and Technology in 2015 and 2019, respectively. He is currently a Postdoctoral Fellow of the Centre for Wireless Communications, University of Oulu. His current research interests include communication-efficient distributed learning and machine learning applications for wireless communication.

Beatriz Soret (Member, IEEE) received the M.Sc. and Ph.D. degrees in telecommunications from the University of Malaga, Spain, in 2002 and 2010, respectively. She is currently an Associate Professor with the Department of Electronic Systems, Aalborg University, and a Senior Research Fellow of the Communications Engineering Department, University of Malaga. Her research interests include LEO satellite communications, distributed and intelligent IoT, timing in communications, and 5G and post-5G systems.

Sumudu Samarakoon (Member, IEEE) received the B.Sc. degree (Hons.) in electronic and telecommunication engineering from the University of Moratuwa, Sri Lanka, in 2009, the M.Eng. degree from the Asian Institute of Technology, Thailand, in 2011, and the Ph.D. degree in communication engineering from the University of Oulu, Finland, in 2017. He is currently an Assistant Professor and a member of Intelligent Connectivity and Networks/Systems Group with the Centre for Wireless Communications, University of Oulu. His main research interests are in heterogeneous networks, small cells, radio resource management, machine learning at wireless edge, and game theory. He received the Best Paper Award at the European Wireless Conference, Excellence Awards for innovators, and the outstanding doctoral student in the Radio Technology Unit, CWC, University of Oulu, in 2016. He is also a Guest Editor of Telecom (MDPI) special issue on "Millimeter Wave Communications and Networking in 5G and Beyond."

Anis El Gabli (Member, IEEE) received the B.Sc. degree in electrical and electronic engineering from the University of Tripoli, Libya, in 2004, the M.Eng. degree from UKM, Malaysia, in 2007, and the M.Sc. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, in 2015 and 2018, respectively. He is currently a Postdoctoral Researcher with the Centre of Wireless Communications, University of Oulu. His main research interests include heterogeneous networks, radio resource management, vehicular communications, video streaming, and distributed/decentralized machine learning and optimization.

Mehdi Bennis (Fellow, IEEE) is a Full (tenured) Professor with the Centre for Wireless Communications, University of Oulu, Finland, an Academy of Finland Research Fellow, and the Head of the Intelligent Connectivity and Networks/Systems Group. His main research interests are in radio resource management, heterogeneous networks, game theory, and distributed machine learning in 5G networks and beyond. He has published more than 200 research papers in international conferences, journals and book chapters. He has been a recipient of several prestigious awards, including the 2015 Fred W. Ellersick Prize from the IEEE Communications Society, the 2016 Best Tutorial Prize from the IEEE Communications Society, the 2017 EURASIP Best paper Award for the Journal of Wireless Communications and Networks, the all-University of Oulu Award for Research, the 2019 IEEE ComSoc Radio Communications Committee Early Achievement Award, and the 2020 Clarviate Highly Cited Researcher by the Web of Science. He is an Editor of IEEE TRANSACTIONS ON COMMUNICATIONS and a Specialty Chief Editor for Data Science for Communications in the *Frontiers in Communications and Networks* journal.

Vivek Kulkarni received the Master of Science degree in communications engineering from the Technical University of Munich (TUM) in 2000, and the M.B.A. degree in innovation and business creation from the TUM School of Management in 2014. In more than 20 years of industrial experience, he worked in several industry domains, such as Industrial automation, Renewable energy, Carrier and Mobile networks with focus on communication protocols, future network architecture, MPLS, GMPLS, carrier-grade Ethernet, and Profinet. He is currently the Project Coordinator of the EU H2020 Project IntellIoT (Next-Generation IoT Project) and in the past he was the coordinator of EU-Project SEMIoTICS (IoT Project) as well as the EU H2020 Project VirtuWind, which was one of the prestigious vertical domain projects in the 5G PPP Phase-1 projects. Since 2013, his main focus is on software-defined networking, Industry 4.0, and next-generation IoT.

Petar Popovski (Fellow, IEEE) received the Dipl.-Ing. and M.Sc. degrees in communication engineering from the University of Sts. Cyril and Methodius, Skopje, and the Ph.D. degree from Aalborg University in 2005. He is a Professor with Aalborg University, where he heads the section on Connectivity and a Visiting Excellence Chair with the University of Bremen. He authored the book *Wireless Connectivity: An Intuitive and Fundamental Guide* (Wiley, 2020). He received an ERC Consolidator Grant in 2015, the Danish Elite Researcher Award in 2016, the IEEE Fred W. Ellersick Prize in 2016, the IEEE Stephen O. Rice Prize in 2018, the Technical Achievement Award from the IEEE Technical Committee on Smart Grid Communications in 2019, the Danish Telecommunication Prize in 2020, and the Villum Investigator Grant in 2021. He is currently an Area Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and from 2022 the Editor-in-Chief of IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. He was the General Chair for IEEE SmartGridComm 2018 and IEEE Communication Theory Workshop 2019. His research interests are in the area of wireless communication and communication theory. He is a member of the Large at the Board of Governors in IEEE Communication Society, a Vice-Chair of the IEEE Communication Theory Technical Committee and IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING.