IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 26, 2021, accepted August 8, 2021, date of publication August 18, 2021, date of current version August 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3105944

SDN-Enabled Resource Orchestration for Industrial IoT in Collaborative Edge-Cloud Networks

Application domain: Collaborative edge-cloud networks
Methodology: SDN - enabled resource orchestration

JUDE OKWUIBE[1], JUUSO HAAVISTO[2], IVANA KOVACEVIC[1], (Member, IEEE), ERKKI HARJULA[1], (Member, IEEE), IJAZ AHMAD[3], JOHIRUL ISLAM[1], (Graduate Student Member, IEEE), AND MIKA YLIANTTILA[1], (Senior Member, IEEE)

[1]Centre for Wireless Communication, University of Oulu, 90570 Oulu, Finland
[2]Center for Ubiquitous Computing, University of Oulu, 90570 Oulu, Finland
[3]VTT Technical Research Center of Finland, 02044 Espoo, Finland

Corresponding author: Jude Okwuibe (jude.okwuibe@oulu.fi)

This work was supported in part by the Academy of Finland through the projects: 6G Flagship and DigiHealth under Grant 318927 and Grant 326291, and in part by the AI Enhanced Mobile Edge Computing Project through the Future Makers Program of Jane and Aatos Erkko Foundation and Technology Industries of Finland Centennial Foundation. The work of Ijaz Ahmad was supported by Jorma Ollila Grant.

ABSTRACT Effective, long-lasting Industrial IoT (IIoT) solutions start with short-term gains and progressively mature with added capabilities and value. The heterogeneous nature of IIoT devices and services suggests frequent changes in resource requirements for different services, applications, and use cases. With such unpredictability, resource orchestration can be quite complicated even in basic use cases and almost impossible to handle in some extensively dynamic use cases. In this paper, we propose SDRM; an SDN-enabled Resource Management scheme. This novel orchestration methodology automatically computes the optimal resource allocation for different IIoT network models and dynamically adjust assigned resources based on predefined constraints to ensure Service Level Agreement (SLA). The proposed approach models resource allocation as a *Constraint Satisfaction Problem* (CSP) where optimality is based on the solution of a predefined *Satisfiability* (SAT) problem. This model supports centralized management of all resources using a software defined approach. Such resources include memory, power, bandwidth, and edge-cloud resources. SDRM aims at accelerating efficient resource orchestration through dynamic workload balancing and edge-cloud resource utilization, thereby reducing the cost of IIoT system deployment and improving the overall ROI for adopting IIoT solutions. We model our resource allocation approach on SAVILE ROW using ESSENSE PRIME modeling language, we then implement the network model on CloudSimSDN and PureEdgeSim. We present a detailed analysis of the system architecture and the key technologies of the model. We finally demonstrate the efficiency of the model by presenting experimental results from a prototype system. Our test results show an extremely low solver time ranging from 0.47 ms to 0.5 ms for nodes ranging from 100 to 500 nodes. With edge-cloud collaboration, our results show about 4 percent improvement in overall task success rates.

Simulation platforms:
* SAVILE ROW
* ESSENSE PRIME
* CloudSimSDN
* PureEdgeSim

INDEX TERMS Constraint satisfaction problem (CSP), software defined resource management (SDRM), software defined networking (SDN), edge computing, cloud computing, Industry 4.0, Internet of Things (IoT), Industrial IoT (IIoT), resource management.

I. INTRODUCTION

THE ongoing industrial revolution, the Industry 4.0, aims at realizing interconnected, responsive, and self-optimizing large-scale production of goods and assets through a seamless integration of advanced manufacturing techniques with Industrial Internet of Things (IIoT). IIoT is characterized by massive integration of smart objects, machines, sensors and Cyber-Physical Systems (CPS) across manufacturing and

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Quan.

management paraphernalia of modern day industries [1]. Manufacturers and businesses are harnessing emerging technologies, such as Software Defined Networking (SDN), Network Function Virtualization (NFV), cloud and edge computing technologies to drive operational efficiency at an industrial scale. It is no surprise that IIoT has received a lot of attention from both research and industry experts recently [2]–[6]. IIoT promises to revolutionize the industrial sector through the power of connected machines, sensors, and devices. The number of such integrated devices are estimated to run into tens of billions of devices over the next decade [7]. IIoT focuses heavily on machine-to-machine (M2M) communication, machine learning and big data. This enables organizations to unlock access to unprecedented amounts of data and rapidly extract insights for continuous improvement of products, services, and processes.

A recent survey by Microsoft found that 91% of companies have at least one IIoT project in the works [8], while a forecast by Million Insights predicted that the IIoT market could reach 992 billion dollar in global spending by the year 2025 [9]. This growth will be spurred by cost-effectiveness and the ease of availability of required devices like processors, sensors, and connected systems.

To set a better sail for the industries, erudite research and development works are going into developing and optimizing technologies and solutions that are aimed at addressing both current and foreseeable challenges that deploying such IIoT solutions face. Among these technologies are SDN [10], NFV [11], Multi-Access Edge Computing (MEC) [12], cloud computing [13], and containerization technologies [14]. Each of these technologies offer a unique value proposition towards the realization of the IIoT vision.

SDN and NFV are two of the latest technologies designed to introduce flexibility in network management and orchestration. SDN is mainly characterized by the decoupling of the control plane from the data plane and providing programmability for network application development [10]. SDN primarily consists of controllers and switches interacting through a standard communication protocol, with OpenFlow being the *de facto* standard [15]. When a new packet arrives, the switch will match the packet header to the flow table and forward the packet according the flow entry it matches. If such a packet is unmatched by any flow entry, it is then sent with a packet-in message to the controller where the controller then decides the appropriate action for such a packet. For undropped packets, the controller will compute the new routing path and update the flow table with a new flow entry which is then forwarded to the corresponding switches to ensure seamless forwarding of future matching packets. With this, SDN is able to maintain a simplified network design while optimizing resources.

NFV, on the other hand, enables the virtualization of network functions by installing such functions on general-purpose servers in the cloud [16]. Generally, software-defined solutions are fast becoming more prevalent in various aspects of modern day Information and Communication Technologies (ICT) and beyond. Companies like Steinberg,[1] a German musical software and hardware company is leveraging the SDN concept to develop modern music production kits by decoupling the musical notes and key functions from the underlying hardware and introducing a programmable software platform that allows users to control the functions of the different elements of the kit.

Virtualization has made network provisioning and automation more feasible and dynamic than with traditional approaches. This, on the one hand, is due to the intrinsic agility of virtualized environments and, on the other hand, due to the full software-based control of different network elements. Security automation is one of such instances where virtualization is leveraged to ease traditional time-consuming and error-prone security tasks on the network by configuring Network Security Functions (NSF) on the virtualized platform [16].

A key defining element of IIoT is the large amount of data generated from connected devices. This data needs to be stored and processed in a timely fashion to enable quick analytical decisions. Cloud computing provides and on-demand remote availability of computer system resources especially data storage and processing capabilities. The cloud is however limited in its ability to effectively support and manage the kind of data generated from IIoT systems [17]. A major limitation being lack of agility, i.e. the ability to process and share required data across nodes in a timely responsive fashion. With such a limitation, certain latency-critical IIoT applications won't operate effectively if they depend solely on the cloud for data and resource provisioning. As such, the computing solution for IIoT systems would require some more advanced and efficient techniques.

The emergence of edge computing together with other technologies like cloudlets, fog, and mist computing come as natural collaborators for the cloud. These technologies provide a pool of additional computational resources for data processing on traditional networks. The applicability of these collaborative technologies often depends on the application areas and individual use cases. The ecosystem of such collaborative edge-cloud solution in an SDN environment is depicted in Figure 1. Here SDN is providing intelligent packet steering and control functions for both cloud and edge applications.

Resource orchestration is another focal point for effective IIoT implementation. Considering that IIoT networks consist of a large number of diverse applications with diverse resource and SLA requirements. As such, there is a high tendency for multiple nodes to simultaneously apply to the IIoT gateway for the same resources, such as time slot to send data, transmission channel, power or other computational resources [18]. However, such resources are usually limited and the IIoT nodes on the network mostly have multiple functions and complicated service models, which makes for variable delay tolerance and service priorities [19]. With this,

[1] https://www.steinberg.net/

it becomes crucial to devise means to dynamically estimate the service priorities of various IIoT nodes and applications on the network, and allocate the corresponding resources based on these predetermined service priorities.

A. MOTIVATION

Despite the proven ability of SDN to offer dynamic traffic programmability and create policy-driven network supervision through the control plane functions, an equally important but much less investigated problem is ensuring a data-driven, dynamic, and conditionally automated control plane function for its various use cases especially in industrial automation.

In order to save the operational cost of implementing and running effective IIoT systems for such industrial automations, the sub-systems and individual processes that make up the IIoT system must have the available resources fairly divided according to the requirements of the processes or sub-systems. For system resources such as power, memory, and storage, we propose the use of SAT constraint modeling where optimal resource allocation is modeled based on predefined constraints to ensure satisfactory Quality of Service (QoS) as well as ensuring Service Level Agreement (SLA) requirement is met. For network resource management, we propose SDN, given its ability to support data-intensive applications like IIoT as well as its support for virtualization.

The optimal resource allocation strategy for each IIoT use case may vary depending on the resources involved as well as other critical components of the use case. However, regardless of the use case, a resource allocation algorithm has to be designed for optimal performances and best satisfaction.

B. CONTRIBUTION

Our main contributions are summarized as follows:

- We propose SDRM, an SDN-enabled scalable and optimal resource allocation scheme based on SMT and CSP modeling for IIoT applications and use cases.
- We implement a CSP model of the proposed scheme on Savil Row using Essence Prime constraint modelling language. We applied the model to various IIoT network instances with different constraints and SLA requirements.
- The solutions from the SAT *Solver* is then used to configure an SDN-enabled edge-cloud network model for IIoT. The system model is implemented using CloudSimSDN and PureEdgeSim.
- We finally demonstrate the efficiency of the proposed model through experimental results from a prototype system.

C. ORGANIZATION OF THE PAPER

The rest of this paper is organized as follows. Section II summarizes the background and related work of relevant technologies. Section III describes the proposed SDRM framework including the system model and the problem formulation. Practical implementation and validation of the proposed model is provided in Section IV. Section V presents the simulation results showing the efficiency of the proposed model as well as other validated benefits. Section VI provides the discussion and future research directions, while Section VII finally concludes the paper.

II. BACKGROUND AND RELATED WORK

A. RESOURCE ORCHESTRATION FOR IIoT

In literature, the automatic orchestration of resources for various IIoT application areas represents a central research area. In [20], authors presented a resource service model for an underground intelligent mine leveraging on IoT platforms. For this use case, authors proposed a resource service model based on Transparent Computing (TC). This model took a centralized approach to resource management, while distributed architecture was utilize for resource storage. In addition, the model provides a scalable approach to resource expansion, where increase in number of devices leads to a corresponding increase in the number of active servers set to handle various IIoT applications.

A heuristic approach to resource management for a satellite communication system is presented in [21]. Here, resource optimization is related to the classical multi-knapsack problem (MKP), and allocation is based on subscriber request, priority level, as well as the assured bandwidths. This approach also measures the satisfaction level of the allocation process. Similar to the approach in [20], some scaling schemes are proposed for this model to ensure that allocated sums of assigned resources does not exceed the available bandwidth budget. This approach is mainly limited in that it is centered on optimizing a specific resources, however current trends suggest that future IIoT systems would require a more centralized approach to resource management which not only optimizes resources, but also allows for a dynamic and seamless resource conversion from one form to another.

In [22], authors proposed a mixed-integer nonlinear stochastic optimization with long-term constraints based on Delay-Optimal Fog Configuration (D-optimal), Noncooperative Fog Configuration (NCOP), and Single-Slot Constraint (SSC). This approach was aimed at optimizing service hosting and task admission decisions using minimal system information while guaranteeing close-to-optimal performance.

Authors in [23] proposed an automatic resource provisioning approach for cloud environments based on the concept of control monitor-analyze-plan-execute (MAPE) loop. Here the main idea is to improve resource provisioning for cloud services by reducing the overall cost of resources, increasing resource utilization and minimizing SLA violations for cloud services. Table 1 shows a summary of the related work along with their key design objectives, resource allocation schemes and their key performance metrics.

A few papers, such as [24]–[27] and [28], consider learning algorithms based resource allocation in edge-cloud architecture. Semi-Markov decision process criterion is used in [24],

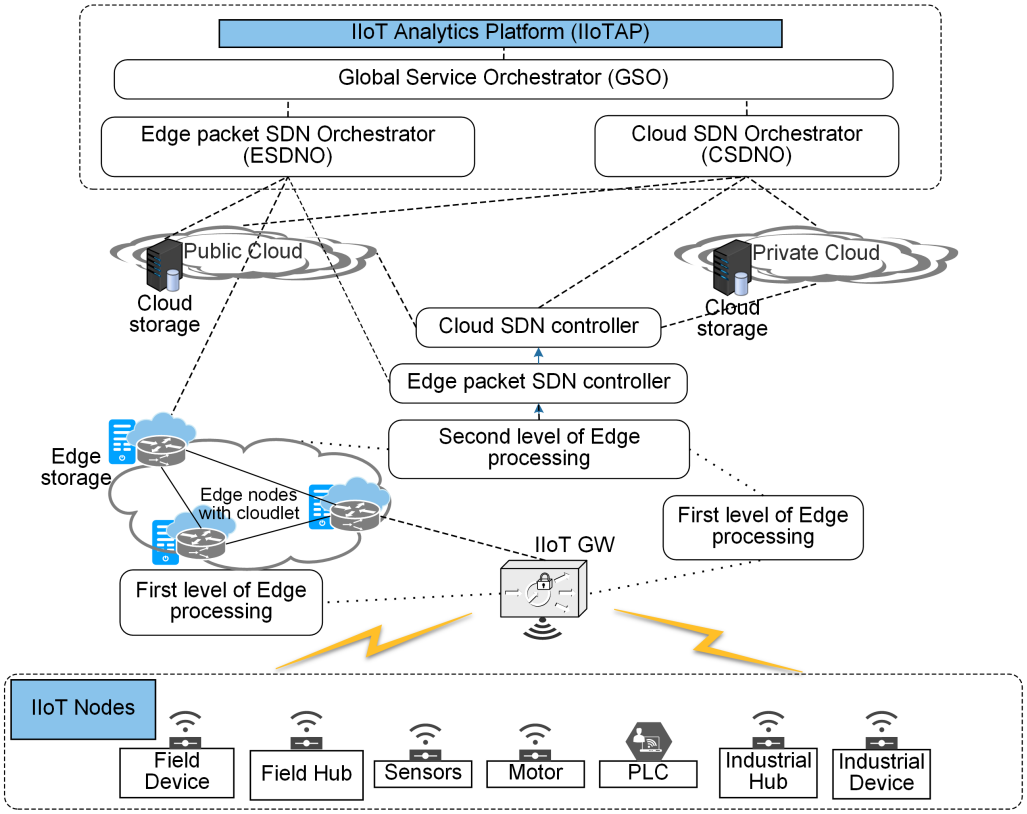


FIGURE 1. An SDN-enabled IIoT ecosystem with collaborative edge-cloud processing.

where the optimization problem is solved using linear programming. They assume that offloading to distant cloud only occurs when there is no cloudlet coverage. Reinforcement learning solution is proposed in [26] for selecting appropriate collaborative edge servers and allocating corresponding portion of the computing task to individual edge severs as well as the radio bandwidth resource. A simple scheme is adopted that offloads task to the cloud only when edge servers are occupied. Two virtual machine allocation methods based on semi-Markov decision process are proposed in [25] to balance the tradeoff between the high cost of providing services by the remote cloud and the limited computing capacity of the local fog. Model-based planning method and model-free reinforcement learning (RL) method are used to allocate virtual machines. The authors distinguish the high and low-priority services. Only high-priority services can access the cloud while low priority services can be accepted if there is free space in the fog. The problem of edge-cloud offloading is addressed in these works by setting predefined policies. In our work, this decision is dynamic, and depends on the instantaneous network conditions, demand and service priority. In [25] priority is used to construct the fixed offloading policy. In [28] deep q-learning algorithm is used to minimize the weighted sum of energy and delay in dynamic allocation. All these works assume general computational tasks offloading schemes, while our work focuses on the specifics of IIoT use case, utilizing inherent priority level classification of services in the scenario. In addition our work considers storage consumption requirement critical for processing high volumes of generated data in IIoT. Aforementioned work focuses only on computational resources allocation.

B. EDGE-CLOUD COMPUTING ARCHITECTURE

In traditional cloud systems, the data-processing and associated decision-making logic are handled at centralized cloud data centers. However, novel IoT applications, such as latency-critical industrial process control systems or highly data-intensive healthcare imaging systems, among many others, require real-time communications and generate high volumes of data, which are both problematic to run on centralized cloud systems. The latency of accessing traditional centralized cloud computational resources is typically too high for real-time operation due to high logical and geographical distance between the end-user and the cloud [29]. Furthermore, exchanging high volumes of data between the local nodes and data centers is sub-optimal from the viewpoint of resource-efficiency [30].

To overcome these challenges, the concept of edge computing has emerged to bring parts of the cloud computing capabilities closer to the end-user devices and data sources [29]. Edge computing can provide several desired features for IoT scenarios, such as local pre-processing and filtering of raw

TABLE 1. Comparison of related works on resource orchestration.

Reference	Environment	Design objective	Resource allocation scheme	Performance metrics
[21]	Satellite communication systems	Resource management based on requests, priority levels, and assured bandwidths	Heuristic algorithms together with some innovative scaling schemes	Satisfaction measure based on priority level, assured level, and QoS
[20]	Industrial IoT	Supports centralized management of resources	Transparent Computing (TC)	Satisfaction measure based on efficiency, cross-OS IoT service, cost, reliability
[39]	Cloud, Edge, Fog an IoT layers	Providing computing paradigm based on deployment and migration strategies related to the infrastructures and application requirements	Smart Orchestration using MicroELement (MEL)	Satisfying the end users' Quality of Service (QoS)
[22]	Adaptive Fog	Optimize service hosting and task admission decisions using minimal system information while guaranteeing close-to-optimal performance	Mixed-integer nonlinear stochastic optimization with long-term constraints	Delay-Optimal Fog Configuration (D-optimal), Noncooperative Fog Configuration (NCOP), Single-Slot Constraint (SSC)
[40]	Edge clouds	Establishing cross-service communication between network slices of different tenants	Blockchain-based service orchestrator that leverages the automation capabilities of smart contracts	Reduced orchestration overhead
Our work	IIoT system level, Edge cloud	Optimize resource allocation based on predefined constraints, minimize edge resource utilization	Constraint Satisfaction Problem (CSP) where optimality is based on the solution of a predefined SAT problem, SDN.	Execution time, total cost, satisfaction measure based on priority level, assured level, QoS, tasks success rates

data to reduce network burden and avoid unnecessary propagation of sensitive raw data, or analysis and decision-making in proximity for improved latency [31].

Therefore, by moving processing from data centers to the edge, cloud systems can better serve applications requiring low latency while saving computational and networking resources at core networks and data centers. In this cloud-edge continuum, different system parts can be deployed in an optimal computational tier (data center or edge server) based on the application or service requirements related to e.g. performance, efficiency, security privacy, and the available computational and network capacity provided by the underlying architecture. For example, data pre-processing and latency-critical functions can be deployed on the edge tier to reduce the data volumes to be delivered on cloud servers and to enable real-time operation, whereas functions that require high computational performance or high interaction with other data sources could be offloaded to a data center.

C. SDN-ENABLED COLLABORATIVE CLOUD-EDGE NETWORKS FOR IIoT

In SDNs, the network can be programmed at run-time from a central vintage point, thus, enabling live service or network function migration without perceivable session breakups through proactive flow path setups. The basic management challenges in IoT are related to security, load balancing and resource management. Each of these can be effectively solved with the logically centralized management, network programmability through APIs, and dynamic approach to handling updates in SDN. Therefore, a number of efforts have been carried out to utilize SDN for improving the security [32], load balancing [33], and management [34] in future networks to facilitate IoT [35]. The run-time packet redirecting capability of SDN [36] can be used to improve the collaboration of cloud and edge computing frameworks for IoT [37].

Leveraging edge-based SDN for IoT has been proposed and evaluated in [38]. The main aim of the work is to mitigate the challenges of heterogeneity of IoTs, packet loss and latency. An SDN-based edge controller improves the ineteroperability of different types of IoT and facilitates the connection of a local IoT network to the global network. The local and global network communications are also facilitated by OpenFlow switches which form the backhaul network. The local edge and centralized cloud resources are synchronized through a specific protocol, called IPv6 over low-power wireless personal area network (6LoWPAN)-SDN protocol (6LE-SDNP) to provide need-based resources for different functions.

In collaborative edge-cloud networks, SDN can help bridge the gap when combining edge computing and traditional clouds. For instance, in our implementation model, SDN serves as the decision-maker on whether tasks or data should be uploaded and processed in the cloud or at the edge. With basic AI integration, the SDN controller can determine periods of high network resource utilization on specific links or sub-systems. The controller can then request more processing to be completed at the edge to eliminate network bottlenecks that could occur if, for some reason, the processing is done in the cloud. In the following section, we discuss how to enable automatic resource provisioning for IIoT in a collaborative edge and cloud frameworks.

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. PROBLEM DESCRIPTION

This paper is predicate on how to automatically compute the optimal resource allocation scheme and configuration for various IIoT use cases and to dynamically

initiate a reallocation according to real-time process demands and SLA requirements. To accomplish this we propose an SDN-enabled Resource Management scheme-SDRM. SDRM automatically computes the optimal resource allocation for various IIoT use cases and dynamically adjusts assigned resources based on predefined constraints to ensure SLA and optimality.

According to our approach, optimality means: 1) minimizing the number of active nodes, sensors, and actuators based on the network configuration and predefined service constraints, hence ensuring that the amount of resources allocated to a given sub-system, matches its active requirements, 2) throttling excess resources and either convert to other limited resources or for the case of subscription-based resources, renegotiate subscriptions according to the demands of the production cycle, and 3) minimizing the amount of real-time data required by nodes and processes to ensure that the need for edge resources such as edge caching is minimized while maximizing the use of cloud storage.

The first goal is accomplished by leveraging the network awareness feature of SDN to power off unused hosts, nodes, and switches in a bid to minimize the number of active nodes and save energy. The second goal is directly tied to cost optimization or reducing environmental impacts as the case may be. However, achieving any of these two objectives will depend on how the required resources are sourced or generated, i.e. whether they are locally generated in the production premise or sourced from third-party service providers or both. Regardless of how resources are sourced or generated, the proposed resource allocation scheme would still ensure optimality with regards allocated resources.

For the sake of simplicity, and to set a model ready for future IIoT systems, we are assuming a fully-fledged IIoT solution where Anything as a Service (XaaS) [41] is fully implementable. Although different sub-systems are still represented in the model, however actual resources are treated as dynamic and provisioned through third parties. As such, converting from one form to another would mean recouping the excess cost associated with one resource and putting the funds back into another resource in a dynamic and optimal fashion.

B. IIoT USE CASE AND SYSTEM MODEL

Our system model is shown in Figure 2. Here we modelled a fully integrated manufacturing facility which consists of four sub-systems: 1) Building Energy Management Sub-system (BEMS), 2) Building Management Sub-system (BMS), 3) Smart Connected Assembly Line and Industrial Plant Control Sub-system (SCAS), and 4) Inventory Monitoring and Management Sub-system (INV). More specific elements of each sub-system is presented in Table 2. Here it is important to note that the values on this table are arbitrary values for illustration purposes. These values are used to show how our approach would perform in such an IIoT system model. With this model, various other IIoT system models can be evaluated on our approach by simply plugging in the

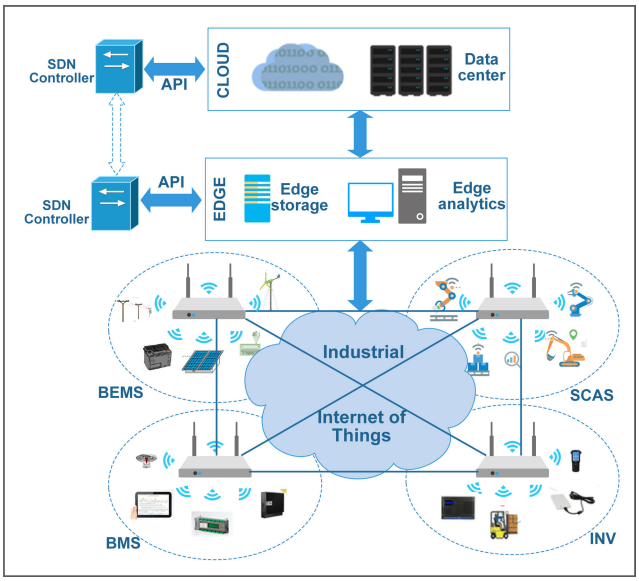


FIGURE 2. System model.

values and parameters for various subsystems, applications and devices. Included in this table are the priority levels of different elements of each sub-system, the use case, corresponding sensors and actuators as well as the ranges of resources required.

1) BUILDING ENERGY MANAGEMENT SUB-SYSTEM (BEMS)

BEMS is arguably the most essential component of both IoT and IIoT solutions. Considering the fact that every other element of such systems require some amount of power to function. BEMS provides monitoring and information specifically focused on systems involving energy use and demand that facilities managers can then act upon to create savings. The BEMS in our use case consists of other sub-systems mainly the solar PV, batteries, aggregators and the inverter. Most functions related to power resource optimization would depend on the BEMS for execution. For example, if the controller calls a demand response event, a BEMS can receive that external signal from the controller and send control instructions to building systems in response. For a scenario where reducing the overall load is required, the BEMS may direct lights to dim in certain areas, increase the temperature set point, and/or shift from utility generation to inverter system. BEMS are also capable of monitoring, aggregating, and processing data at a basic level to inform logic-controlled responses.

2) BUILDING MANAGEMENT SUB-SYSTEM (BMS)

Also known as a Building Automation System (BAS). This offers a central control system for monitoring and controlling all building operations from temperature control to door control, fire alarm, lighting, security systems and elevators, as well as the Heating, ventilation, and air conditioning (HVAC). The BMS also makes the automation of system

TABLE 2. Industrial IoT system model.

Industrial sub-system	Priority level	IIoT application area/Use case	Sensors/Actuators	Memory	Power usage	Bandwidth	Edge storage
Building energy management Sub-system (BEMS)	8	Solar PV	Solar trackers Temperature sensors Radiation sensors	3GB - 5GB	450W - 550W	300 - 350Kbps U 100 - 150Kbps D	5MB - 7MB
	10	Batteries	Acid level sensors Temperature sensors	5GB - 7GB	1.7W - 2.3W	200 - 250Kbps U 100 - 200Kbps D	3MB - 4MB
	7	Aggregators	RPM counter Fuel level sensors Oil level sensors	2GB - 4GB	0.4W - 0.6W	200 - 250Kbps U 100 - 150Kbps D	1MB - 2MB
	8	Inverter	Temperature sensors Voltage regulator	3GB - 4GB	0.45W - 0.6W	150 - 200Kbps U 100 - 150Kbps D	4MB - 6MB
Building management Sub-system (BMS)	6	Temperature control	Temperature sensors Actuators	5GB - 7GB	130W - 180W	200 - 250Kbps U 100 - 200Kbps D	3MB - 4MB
	9	Security alarm	Motion sensors Actuators	7GB - 10GB	150W - 200W	200 - 250Kbps U 100 - 200Kbps D	6MB - 8MB
	9	Door control	Motion sensor Surveillance cameras Actuators	12GB - 15GB	300W - 500W	200 - 250Kbps U 100 - 200Kbps D	50MB - 100GB
	10	Fire alarm	Smoke sensors Fire alarm actuators	10MB - 12GB	120W - 150W	200 - 250Kbps U 100 - 200Kbps D	2GB - 3MB
Smart connected assembly line/Industrial plant control Sub-system (SCAS)	6	Radiation detector	Radiation sensors Actuators	5GB - 7GB	150W - 200W	200 - 250Kbps U 100 - 200Kbps D	2MB - 3MB
	7	Pressure probe	Pressure sensors Actuators	15GB - 17GB	450W - 600W	200 - 250Kbps U 100 - 200Kbps D	3MB - 5MB
	8	Flow rate	Liquid flow meter Liquid control actuators	5GB - 7GB	300W - 500W	200 - 250Kbps U 100 - 200Kbps D	2MB - 3MB
	8	Noise level	Sound level meter Actuators	4GB - 6GB	300W - 400W	200 - 250Kbps U 100 - 200Kbps D	5MB - 7MB
Inventory monitoring and management sub-system (INV)	5	Liquid level	Liquid level sensors Faucet actuators	5GB - 7GB	400W - 600W	200 - 250Kbps U 100 - 200Kbps D	3MB - 4MB
	5	Smart crates	Photoelectric sensors	6GB - 8GB	3W - 5W	200 - 250Kbps U 100 - 150Kbps D	2MB - 3MB
	6	Stock counter	Photoelectric sensors	3GB - 5GB	1.5W - 2W	100 - 150Kbps U 100 - 150Kbps D	1MB - 2MB
	5	Bar code scanner	CMOS linear image sensor	3GB - 5GB	1.8W - 2.5W	100 - 150Kbps U 100 - 150Kbps D	1MB - 2MB

operations possible. For instance, the scheduling of lights or heating systems to turn on and off at certain times is handled by the BMS. Generally, using a BMS by itself even without further optimization would likely produce significant energy savings compared to not using it. This is mostly because it provides easy access to the control elements required to implement cost-efficient operations.

3) SMART CONNECTED ASSEMBLY LINE AND INDUSTRIAL PLANT CONTROL SUB-SYSTEM (SCAS)

This sub-system constitutes the manufacturing hub of the IIoT system. SCAS is a networked system where all assembly related tools and processes are networked to each other, and integrated into the production network. Data generated from SCAS is used for active control of the manufacturing process. It also provides real-time information on processes that need improvement. It is important to note that most of the data generated in an IIoT system comes from the SCAS. The SCAS is also one of the most sensitive and least tolerant to delays or service breaks. The margin for error is very low. One half inch too early or too late and the entire system could come to a complete halt, or worse still, major manufacturing faults could occur.

4) INVENTORY MONITORING AND MANAGEMENT SUB-SYSTEM (INV)

Also called as inventory system, INV mainly tracks goods and raw materials throughout the entire supply chain, from purchasing to production to end sales. Inventory management is an essential part of IIoT if full automation is to be achieved. Each company will manage stock in their own unique way, depending on the nature and size of their business. From basic Excel spreadsheets and smart crates to more sophisticated stock counters and bar code scanners that run on CMOS linear image sensor technology. We model this system on SAVILE ROW using ESSENSE PRIME modeling language.

C. RESOURCE MANAGEMENT AND ALLOCATION

We consider resource allocation for an IIoT system where resources and services are provisioned by service providers. We adopt the concept of Service Level Agreement (SLA), which is a set of parameters that defines the capabilities and the paid rights of every resources in the IIoT system becomes essential. Furthermore, the concept of Committed Resource Rate (CRR), which is the rate that is guaranteed by the system, is also essential. Typically it is low enough so that the system can guarantee the IIoT application has this specific rate. Then we have the Peak Resource Rate (PRR),

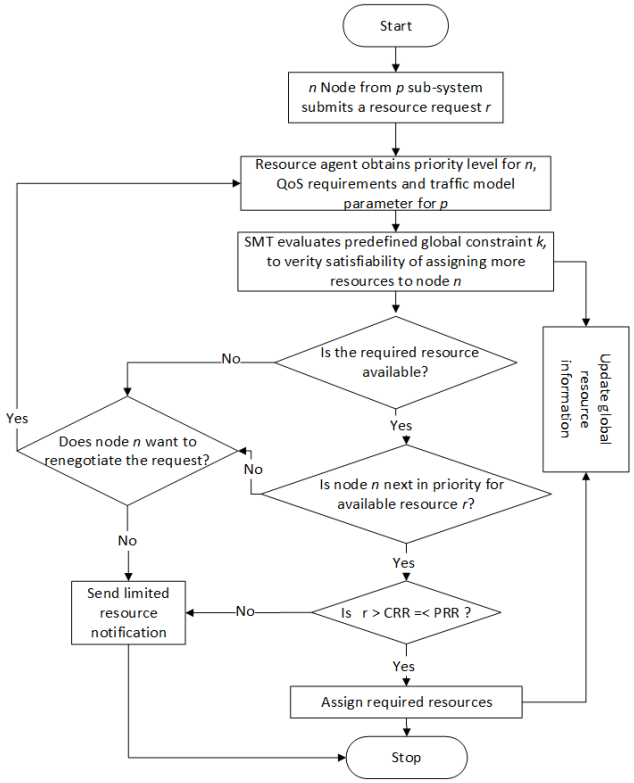


FIGURE 3. Flowchart for system resource allocation technique.

which is the maximum rate of resources that may be allocated to a particular application or process. When a process or application demands for a rate higher than CRR and lower or equal to PRR, the system will allocate this rate if available. However, when a process demands for a rate higher than PRR, the system will consider it as if the request was equal to PRR. We use the term assured resource as a quantity that the system can provide most of the time but not always.

We also have the Subsystem Priority (SSP), which is a decision parameter that assigns values to each subsystem to represent the priority level of the subsystem. In this research, we assumed that the priority of the subsystems are predefined during the initial set-up. These priority levels determine the significance of the subsystem according to their value cost paid to the service provider. In a situation of low resources, requests from high priority subsystem may cause the system to decrease the rate of low priority processes to CRR. The flowchart for our system resource allocation technique is shown in Figure 3.

D. CONSTRAINT BASED RESOURCE ALLOCATION

Constraint Programming (CP) offers an efficient method of solving complex constrained optimization or combinatorial problems [42]. For high-level system resource allocation expression, we use the ESSENCE PRIME constraint specification language. It is a constraint modelling language, which allows the user to solve constraint satisfaction problems (CSPs). It is mainly designed for describing $\mathcal{NP}$-hard

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$          CONSTRAINTS FOR BUILDING MANAGEMENT SUB-SYSTEM (BMS)            $
$                            PRIORITY LEVEL 6                              $
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

$Constraints for memory allocation.

forAll r : bms_mem_iter . (
    bms_mem_balance[r] >= 5 $all nodes must have at least 5MB
    /\ bms_mem_balance[r] <= 15 $and no node should have more than 15MB
    /\
    bms_mem_balance[1] >= 5
    /\ bms_mem_balance[1] <= 7
    /\
    bms_mem_balance[2] >= 7 $alarm system node memory must be at least 7MB
    /\ bms_mem_balance[2] <= 10 $but not more than 10MB
    /\
    bms_mem_balance[3] = 12 $door control node must always have 12MB
    /\
    bms_mem_balance[4] >= 10
    /\ bms_mem_balance[4] <= 12
),

(sum i : int(1..4) . bms_mem_balance[i]) <= bmsMem
```

FIGURE 4. Sample constraints for BMS sub-system in our demo system model.

decision problems [43], [44]. An ESSENCE specification identifies the input parameters of the problem class which is initiated by the keyword *given*, whose values define an instance. The combinatorial objects to be found is initiated by the keyword *find*. The constraints the objects must satisfy is initiated by the key phrase *such that*, while the keyword *letting* is used to declare the identifiers. We also need an optional objective function *(minimizing/maximising)* to enable the solver decide what the objective of the constraints are. For our case, we needed the *minimizing* function. Here, the combinatorial object to be found is represented by a single abstract variable whose type is set of sequence of int. Figure 4 shows a snippet of some of the constraints we defined for the BMS-subsystem in our implementation model.

IV. IMPLEMENTATION AND VALIDATION

A. SYSTEM MODELING TOOLS

1) ESSENSE PRIME AND SAVILE ROW

SMT solvers for SDN resource allocation can be considered as a tool for solving decision problems over questions that can be proposed with predicate logic. While the SMT solver applications at large could be thought more as for model checking purposes for software verification, e.g., properties of liveness and safety (e.g., Z3 and TLA+), here, for this study, we are mainly interested in model optimizations rather than satisfiability properties. In fact, we consider our class of models always satisfiable, i.e., that resources can always be allocated. Instead of focusing on satisfiability, we use SMT for finding the most optimal results.

The Essence Prime language and the Savile Row tool complement our objective by having an easy way to represent objective functions (min/maximising) in SMT models.[2] Effectively, the objective functions, as provided by Essence Prime, allow us to trivially perform a complete search to

[2]Language representations in different SMT languages can be seen on the SONET problem: https://www.csplib.org/Problems/prob056/models/

the underlying model representation, which returns us an optimized model.

The SDN resource allocation problem resembles the knapsack problem: given a set of items, each with a mass and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. With SDN, the items are nodes, mass and value are resources like RAM and wireless spectrum, and then the knapsack size is the sum of edge applications we want to allocate.

2) CLOUDSIMSDN

Although Mininet [45] is widely considered as the conventional tool for emulating the network topology of OpenFlow switches and Software-Defined Networking systems in general. It is however, limited mainly to the testing of different SDN-based traffic management policies in controller. As such, it does not provide an adequate network environment for emulating and testing other cloud resource management techniques such as Virtual Machine (VM) along with network resource consolidation.

To address these limitations, Son *et al.* [46] introduced CloudSimSDN[3]; an SDN extension of CloudSim project version 2.0.[4] CloudSimSDN enables more advanced features like the simulation of policies for the joint allocation of compute and network resources, which is a key element of our approach. In addition, CloudSimSDN simulates utilization of hosts and networks, and response time of requests in SDN-enabled cloud data centers. Furthermore, it supports resource provisioning for NFV in the edge computing environment and also provides a simulation framework for NFV in edge and cloud computing for inter cloud data centers. Other features include support for policy-based network link selection, VM allocation, Virtual Network Function(VNF) placement, and SFC auto-scaling algorithms. It also provides performance evaluation of the framework for dynamic use case scenarios.

CoudSimSDN offers scheduling policies such as VM placement algorithm and network policy management. This is achieved by programming the brokers to simulate actual end-user behaviors or data centers through a series of predefined policies [46]. Such policies are user-defined, hence they could be designed as an extension of the abstract class or from built-in policies. It further provides a resource allocation layer which contains modules that allocate resources such as CPU power, memory, and storage size according to the user-defined parameters.

In creating our IIoT system model, we leveraged a switch class in CloudSimSDN to perform SDN-enabled switching functions managed by the controller. The physical topology consists of IIoT hosts, switches, and links. Each host is configured with a number of VMs, each representing an individual element or a set of joint-functional elements of the IoT device such as sensors and actuators. Here each host represents an IIoT device on the network. The hosts are then assigned resources according to the solution of the solver from the SAT constrained solution. With that each node is then specified with their computational power, memory, and storage size. Links connect a host to a switch or connect switches with specified bandwidth.

In defining out link parameters, we adopted the model presented in [46], where the bandwidth of each link is determined by the priority level of the end hosts it connects to. Here SDN is configured to allocate a specific amount of bandwidth to each channel according to the given priority level. The priority levels for our IIoT use case is presented in Table 2. After allocating these required bandwidths with maximum satisfiability, the remaining bandwidths are then reserved for on-demand allocations or some system-level network configuration changes. A cost-based SAT optimization cycle could also use the information generated after such full bandwidth allocation to renegotiate the global bandwidth allocation of the entire IIoT system. With this bandwidth allocation approach, the allocated bandwidth $BW_{c,l}$ for a channel c in the link l is defined by the equation:

$$BW_{c,l} = \frac{BW_l}{N_l} \tag{1}$$

where the link (l) has available bandwidth (BW_l) shared by the number of channels (N_l). To simplify the link model of our implementation, we adopted the *weakest link approach* for scenarios where a channel consists of multiple links with varying bandwidths. In such a case, we take the channel bandwidth to equal the bandwidth of the link with the least bandwidth [46]. Hence for a given time period Δt during which the number of channels remain unchanged, the amount of transferred data D_c from a given sender to a receiver on a channel c can be calculated using the equation:

$$D_c = \Delta t \times Min(BW_{c,l}) \tag{2}$$

A channel update message is sent across to all existing channels through the Network Operating System (NOS) which serves the role of the controller to inform all links of a newly added or removed channel. Consequently, the new bandwidth value is then factored in according to the weakest link approach. The new value for the channel bandwidth is then sent across to all the links in the channel.

B. EFFECTS OF CLOUD-EDGE COLLABORATION (PUREEDGESIM)

To study the effect of cloud-edge collaboration on our system model, we implemented the same network configuration using PureEdgeSim[5]; another extension of the CloudSim designed to efficiently handle various resource management strategies in cloud, edge, and mist computing environments [47]. PureEdgeSim allows for a highly scalable design capable of integration thousands of IIoT devices into the

[3]https://github.com/Cloudslab/cloudsimsdn

[4]https://github.com/Cloudslab/cloudsim

[5]https://github.com/CharafeddineMechalikh/PureEdgeSim

```
<?xml version="1.0"?>
<edge_devices>
   <device arch="x86" os="linux" vmm="xen">
      <mobility>true</mobility>
      <speed>1.4</speed>
      <minPauseDuration>100</minPauseDuration>
      <maxPauseDuration>400</maxPauseDuration>
      <minMobilityDuration>10</minMobilityDuration>
      <maxMobilityDuration>60</maxMobilityDuration>
      <battery>true</battery>
      <percentage>30</percentage>
      <batteryCapacity>18.75</batteryCapacity>
      <idleConsumption>0.078</idleConsumption>
      <maxConsumption>3.3</maxConsumption>
      <isOrchestrator>false</isOrchestrator>
      <generateTasks>true</generateTasks>
      <hosts>
          <host>
              <core>8</core>
              <mips>25000</mips>
              <ram>4000</ram>
              <storage>128000</storage>
              <VMs>
                  <VM>
                      <core>8</core>
                      <mips>25000</mips>
                      <ram>4000</ram>
                      <storage>128000</storage>
                  </VM>
              </VMs>
          </host>
      </hosts>
   </device>
```

FIGURE 5. The edge device parameters in XML file.

setup. Another great feature of PureEdgeSim is the support for heterogenous devices by taking to considerations all major granularity of such IIoT device. This includes features like mobility, power source, i.e. battery-powered or not, different applications requirements, i.e. tasks file size, tasks CPU utilization, latency requirement and so on. Such device parameter granularity also extends to other components like the cloud and edge data centers. The XML snippets of our configuration parameters for edge devices, cloud, and edge data centers are presented in Figures 5, 6, and 7 respectively.

V. MEASUREMENT RESULTS AND ANALYSIS

In this section, we present performance evaluation and analysis of the results from our demo system. First we analyze the performance of the SAT constraint modeling approach we used for resource allocation based on the execution time of the solver and the number of solver nodes required for different network instances. We then analyze the SDN based implementation on CloudSimSDN based on power consumption and other analytics of the SDN broker. Finally, we present the effects of cloud-edge collaboration on CPU and network utilization as well as the task success rates as the number of IIoT devices increase.

A. SAT CONSTRAINT BASED RESOURCE ALLOCATION

Table 3 presents the performance parameters of the SAT constraint allocation scheme for our 5 tested network instances.

```
<?xml version="1.0"?>
<cloud_data_centers>
 <datacenter arch="x86" os="Linux" vmm="Xen">
     <idleConsumption>0.00015</idleConsumption>
     <maxConsumption>0.016</maxConsumption>
     <isOrchestrator>true</isOrchestrator>
     <hosts>
         <host>
             <core>8</core>
             <mips>2000000</mips>
             <ram>16000</ram>
             <storage>1000000</storage>
             <VMs>
                 <VM>
                     <core>1</core>
                     <mips>250000</mips>
                     <ram>2000</ram>
                     <storage>20000</storage>
                 </VM>
```

FIGURE 6. Cloud data center parameters in XML file.

```
<?xml version="1.0"?>
<edge_datacenters>
  <datacenter arch="x86" os="Linux" >
     <idleConsumption>0.00015</idleConsumption>
     <maxConsumption>0.016</maxConsumption>
     <isOrchestrator>false</isOrchestrator>
     <location>
         <x_pos>100</x_pos>
         <y_pos>100</y_pos>
     </location>
     <hosts>
         <host>
             <core>8</core>
             <mips>800000</mips>
             <ram>16000</ram>
             <storage>200000</storage>
             <VMs>
                 <VM>
                     <core>4</core>
                     <mips>200000</mips>
                     <ram>4000</ram>
                     <storage>20000</storage>
                 </VM>
```

FIGURE 7. Edge data center parameters in XML file.

TABLE 3. Time and node parameters for minion solver solution with different network instances.

Network instance	Number of nodes	Minion solver nodes	Minion solver total time (s)	Seville row total time (s)
Net. Inst. 1	100	74	0.001959	0.471
Net. Inst. 2	200	72	0.002960	0.490
Net. Inst. 3	300	72	0.002202	0.499
Net. Inst. 4	400	74	0.002333	0.518
Net. Inst. 5	500	74	0.002933	0.501

The number of nodes for each network instance is also showed on the Table. Here the variation in the number of Minion solver nodes required for each network instance doesn't depend directly on the number of IIoT node for the given

instance. For instance, the number of solver nodes required for the 100 nodes network instance is 74, while that for the 300 nodes network is 72. Although the average total time of the Minion solver is vanishingly small, we observed a minimal progression in value as the number of network nodes increased. The same observation was true for the Saville row total time. Such extremely low magnitude of Solver time becomes very essential for realizing the real-time dynamism required for optimal resource allocation in an IIoT system.

B. SDN ENABLED IMPLEMENTATION ON CLOUDSIMSDN

The node, host, and link parameters for both the physical and virtual networks are shown in Tables 4 to 8. Tables 9 and 10 show the outcome of the SDN integration into our system model. Table 9 presents the switch utilization and power consumption parameters of our system model with two separate workloads and two different VM placement models, namely; the Best Fit (MFF, Most Full First) and Worst Fit (LFF, Least Full First). Our goal is to test the effect of VM consolidation in an SDN-enabled cloud data center. To implement such VM consolidation, the system leverages the network awareness feature of SDN to power off unused hosts and switches to save energy. With workload 1, the LFF placement model powers off the edge-2 and second-level edge-2 switches, hence saving about 200 watts of power. The MFF model turns off edge-1 and second-level edge-1 switches also saving about 200 Watts of power. With workload 2, the same switches are turned off for each VM placement models, however with more power savings of about 286 Watts.

The SDN broker also provides further results showing the CPU, network and server times for the two workloads as shown in Table 10. With the average server time per task being 1.02s for workload 1, 1.17s for workload 2, and 1.09s for workloads 1 and 2. The average CPU time for workload 1 is 0.012s, while for workload 2 is 0.013s and for workloads 1 and 2 is also 0.013s. This goes to show the little to no effect an increase in workload has on the SDN broker time parameter, hence making it a more scalable solution for discrete resource planing in IIoT.

TABLE 4. Node parameters for physical network topology.

Name	Type	IOPS	UpPort	DnPort	Bandwidth
GW(x)	gateway	2000000000	1	2	25000000
Inter(x)	intercloud	2000000000	0	2	25000000
Core	core	2000000000	0	2	25000000
Edge(x)	edge	2000000000	1	4	25000000

TABLE 5. Host parameters for physical network topology.

Name	Type	PES	MIPS	RAM	Storage	Bandwidth
BEMS	host	1	4000000	10240	10000000	25000000
BMS	host	1	4000000	10240	10000000	25000000
SCAS	host	1	3000000	5120	5000000	12500000
INV	host	1	2000000	5120	5000000	12500000

TABLE 6. Link parameters for physical network topology.

Source	Destination	Latency
Core	Edge1	1.0
Core	Edge2	1.0
Edge1	BEMS	0.5
Edge1	BMS	0.5
Edge2	SCAS	0.5
Edge2	INV	0.5

TABLE 7. Node parameters for virtual network topology.

Name	Type	Size	PES	MIPS	RAM
VM01	vm	2000	1	3000	512
VM02	vm	2000	1	3000	512
VM03	vm	2000	1	3000	512

TABLE 8. Link parameters for virtual network topology.

Name	Source	Destination	Bandwidth
L32	VM03	VM02	500000
L21	VM02	VM01	500000
Default	VM01	VM02	-
Default	VM02	VM01	-
Default	VM02	VM03	-
Default	VM03	VM02	-

C. EFFECTS OF CLOUD-EDGE COLLABORATION

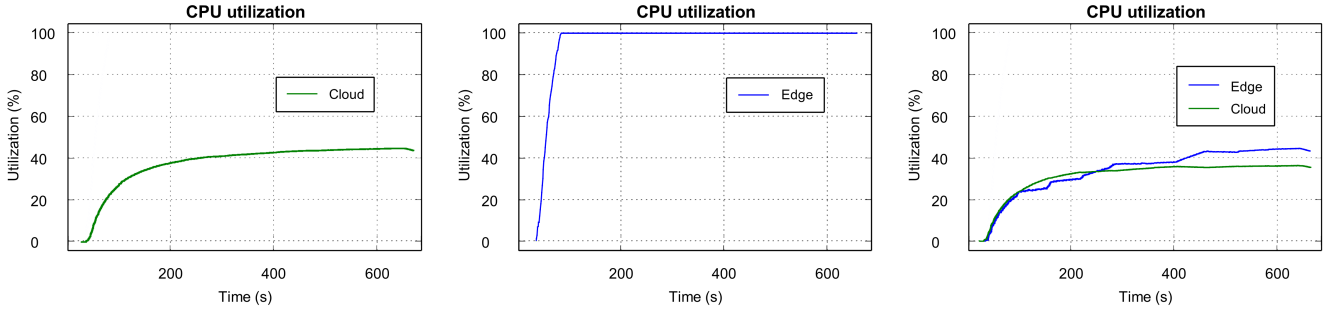
Here we configured PureEdgeSim with the simulation parameters presented in Table 9. The goal is to analyze the effect such cloud-edge collaboration would have on the network performance with regards to CPU utilization, network utilization, as well as the task success rates. As such, we tested for three main scenarios, the cloud-only, the edge-only and the cloud-edge scenarios. In addition, we also wanted to observe what effect the number of IIoT nodes would have on this performance metrics, as such we tested with an increasing number of nodes from 200 to 400 nodes.

1) CPU UTILIZATION

Figure 8 shows the effect each scenario has on CPU utilization for 200 edge devices. Here, the CPU utilization for the cloud-only scenario peaks at about 45% of the available capacity. The edge-only scenario tends to max out the CPU capacity within the first 2 minutes of activity. This is mainly due to limited resources a the network edge. With the edge-cloud scenario, the CPU utilization is fairly shared between the edge and cloud resources, hence the cloud utilization drops to 34% which is about 11% decrease from the cloud alone scenario, while the edge utilization drops to about 47%, which is less than half of the utilization for the edge-only scenario. With the increase in the number of edge devices from 100 to 200, the CPU utilization for the cloud-only scenario caps at about 65%, while for edge-only scenario, this led to a faster maxing out of the total CPU capacity within the first 60 seconds of activity as shown in Figure 9. The edge-cloud scenario, there tends to be a more consistent and uniform distribution of tasks between the edge and the cloud, hence we observe smoother curves that with the 200 edge device use cases. Here the edge utilization peaks at about

TABLE 9. Switch utilization and power consumption of SDRM based on different VM allocation policies.

Switches	Workload 1			Workload 2			Workloads 1&2		
	Switch status		Power	Switch status		Power	Switch status		Power
	LFF VM	MFF VM	Cons. (W)	LFF VM	MFF VM	Cons. (W)	LFF VM	MFF VM	Cons. (W)
Gateway 1	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
Gateway 2	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
InterCloud 1	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
InterCloud 2	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
Edge 1	✓	-	200.7	✓	-	286.6	✓	-	286.6
Edge 2	-	✓	200.7	-	✓	573.2	-	✓	573.2
Core 1	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
Core 2	✓	✓	401.5	✓	✓	573.2	✓	✓	573.2
2 Edge 1	✓	-	200.7	✓	-	286.6	✓	-	289.5
2 Edge 2	-	✓	200.7	-	✓	573.2	-	✓	573.2



(a) Cloud only (b) Edge only (c) Edge-cloud

FIGURE 8. Comparing CPU utilization for cloud-only, edge-only, and cloud-edge IIoT network with 200 edge devices.

TABLE 10. SDN broker analytics for SDRM with different workloads.

Parameter	Workload 1	Workload 2	Workloads 1&2
System Workloads	57	57	114
CPU Workloads	114	114	228
Network workloads	57	57	114
Workloads time out	0	0	0
Total server time (s)	58.19	66.52	124.52
Total CPU time (s)	1.50	1.52	2.99
Total Net. time (s)	56.69	65.00	121.54
Avg. server time (s)	1.02	1.17	1.09
Avg. CPU time (s)	0.012	0.013	0.013
Avg. Net. time (s)	0.99	1.14	1.06

TABLE 11. Edge-cloud simulation parameters.

Parameter	Value
Simulation area (m^2)	[L=300m] [W = 300m]
Edge devices range (m)	25m
Edge data centers coverage(m^2)	$250m^2$
Registry mode	CLOUD
Applications CPU allocation policy	SPACE_SHARED
Min number of edge devices	200
Max number of edge devices	400
Edge device counter size	100
WLAN bandwidth (Mbps)	80 Mbps
WAN bandwidth (Mbps)	60 Mbps
WAN propagation delay (s)	0.2 s
Realistic network model	True
Consumed energy per bit (J/bit)	0.00000004 J/bit

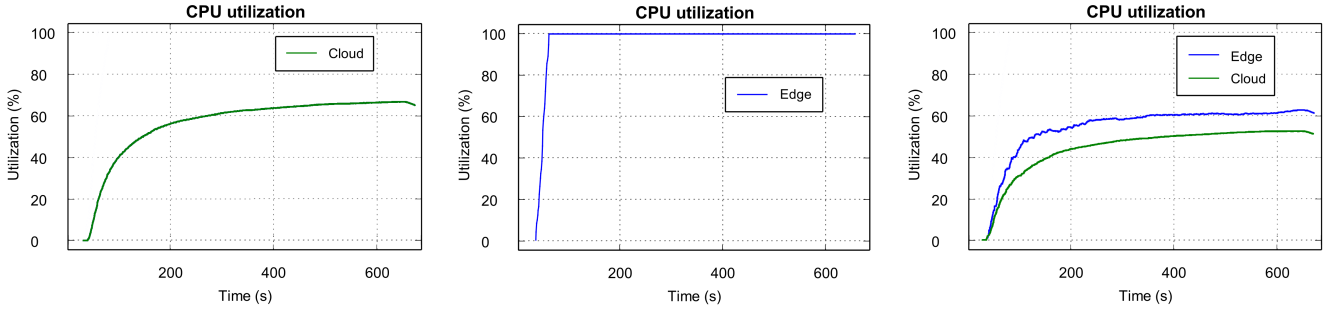
67%, while the cloud utilization peaks at about 50%. Further increasing the number of edge devices to 400 began to max out the CPU resources at both the edge and the cloud as shown in Figure 10. Here the CPU utilization for the cloud-only scenario begins to go above the 90% mark, while for the edge-only scenario, the utilization tends to max out immediately after transmission begins. The edge-cloud scenario still offered more leverage to higher scalability, with cloud resource utilization at about 68%, while the edge utilization peaks at about 82%. This demonstrates the effect of such edge-cloud collaboration in ensuring more scalable, optimal, and dynamic resource utilization.

2) NETWORK UTILIZATION AND STABILITY

Figure 11 compares the effect of the three network models with regards to network utilization. Here it is clear that with the edge-only model, the network utilization pattern is non-bursty and relies on only the LAN network. As such, the WAN utilization stands at about 0Mbps, while the cloud-only and the edge-cloud scenarios have about the same network utilization averaging at about 20Mbps. However, the cloud-only scenario tends to be more bursty than the edge-cloud scenario. This is shown by the more steep edges of the curve. Hence it goes to show a more steady and stable connectivity with the edge-cloud scenario compared to the cloud-only scenario.

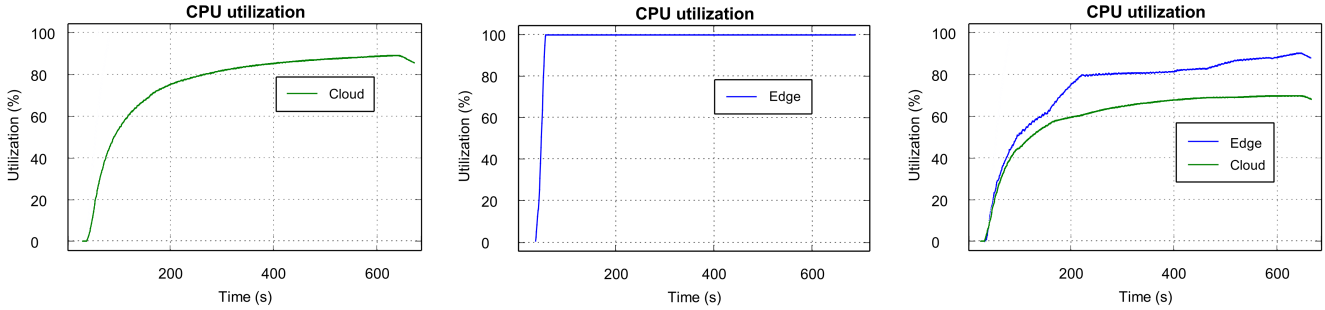
3) TASKS SUCCESS RATES

The tasks success rate is a key indicator of the effectiveness of the three implementation models. This represents the actual fraction or percentage of success among a number of



(a) Cloud only (b) Edge only (c) Edge-cloud

FIGURE 9. Comparing CPU utilization for cloud-only, edge-only, and collaborative cloud-edge IIoT network with 300 edge devices.



(a) Cloud only (b) Edge only (c) Edge-cloud

FIGURE 10. Comparing CPU utilization for cloud-only, edge-only, and collaborative cloud-edge IIoT network with 400 edge devices.

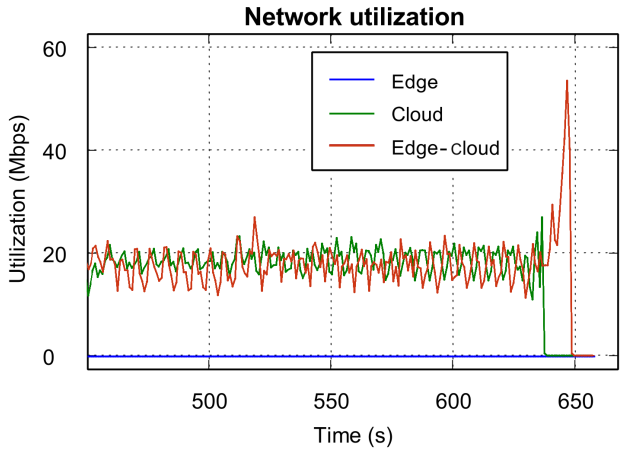
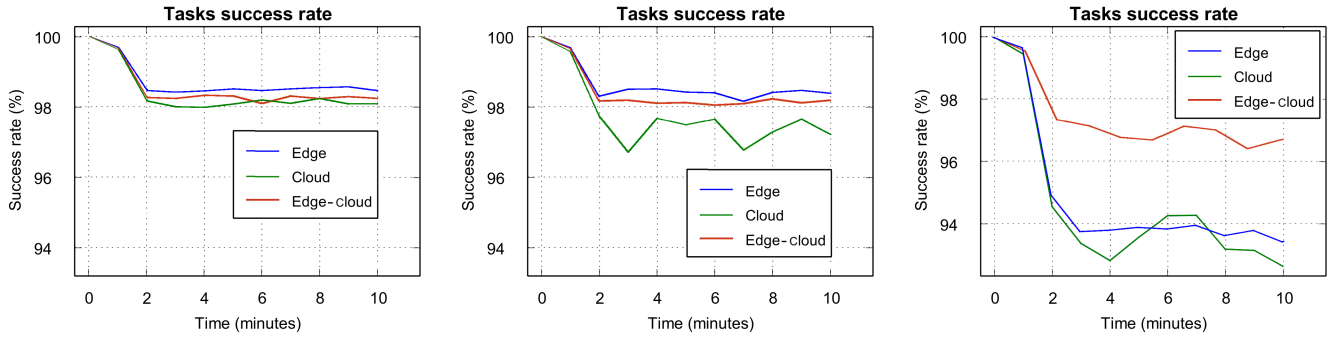


FIGURE 11. Comparing network utilization for edge-only, cloud-only, and collaborative edge-cloud IIoT network with 400 edge devices.

attempts for different IIoT applications and devices. Here we compared the three implementation models to each other and also checked to see what effect increasing the number of edge devices would have on the overall task success rates of each model. We tested this with 200, 300, and 400 edge devices, and the outcome is shown in Figure 12. With 200 edge devices, the cloud-only scenario offers about 98.2% success rate for active tasks, while the edge-only scenario offers about 98.7%, and the edge-cloud scenario offers about 98.4% success rate. Here we conclude that with fewer edge devices, there is little to no variations in the tasks success rates with all three implementation models. However, as we increased the number of edge devices to 300, we began to see a noticeable drop in the success rates of the cloud-only scenario. We deduce that this drop in the success rate is mostly due to an increase in the number of latency-critical tasks, for which the cloud-only model is not fully suited for. For this scenario, the edge-only and the edge-cloud models still maintained fairly the same success rates as with the 100 edge device scenario. Increasing the number of edge devices to 400 showed a rather substantial variations in the task success rates of the three models, as shown in Figure 12c,with cloud-only scenario offering about 93% success rate. Edge-only scenario offers a little more than 93% success rate, while the edge-cloud scenario offers about 97% success rate. This goes to show the added benefit of such edge-cloud collaboration for IIoT use cases. This 4% improvement in task success rate was achieved with edge-cloud collaboration is of substantial significance, considering the sensitivity of IIoT scenarios and use cases.

VI. DISCUSSION AND FUTURE WORK

Our main goal for this work is to analyze the performance improvements of leveraging SDN and constraint programming for resource orchestration in IIoT. We used the SAT constraint programming paradigm and minion solvers for



(a) 200 edge devices (b) 300 edge devices (c) 400 edge devices

FIGURE 12. Comparing tasks success rates for edge-only, cloud-only, and collaborative edge-cloud IIoT network.

resource allocation. Here we defined resource allocation as a combinatorial problem. SAT constraint programming draws on a wide range of techniques from artificial intelligence, computer science, and operations research [48]. In applying constraint programming to our resource allocation problem, we declaratively state the constraints on the feasible solutions for a set of decision variables. In our case, these variables are the amount of resources for the different subsystems and IIoT applications in our system model.

On the network segment, we considered the collaborative combination of edge and cloud computing for optimal resource allocation and improved performance. Here the SDN controller serves as the decision-maker on whether a task or resulting data should be processed on the edge or uploaded and processed in the cloud. In addition, the SDN controller can also determine periods of high network resource utilization on specific links or sub-systems and reroute traffic accordingly or request more resources to ensure SLA is met. The performance of the proposed model is compared for scenario where all network resources are provisioned on the edge versus the cloud, and then both scenarios are compared to the collaborative edge-cloud approach. The outcome underscored the benefits of such collaborative approach, especially on the success rates of tasks. It was also interesting to realize that the number of edge devices significantly influence the marginal benefit of network models. With fewer edge devices, all three models tend to offer about the same tasks success rates. However, as number of edge devices continue increasing, the cloud and edge scenarios displayed substantial drop in tasks leading to lower success rates. The edge-cloud approach maintained a relatively high success rates. As such, it would benefit the system designers to first estimate the number of edge devices and the kind of applications before choosing the most optimal network model for the specific implementation.

Notwithstanding the proven benefits of combining these technologies in industrial automation, adopting and fully integrating these technologies into current IIoT environments poses several technical challenges. Addressing these challenges will be required to elicit a successful wide-scale deployment of the proposed solution. Such challenges include cascade failures where the failure of one or two components causes a ripple effect across other connected devices. Furthermore, we have the challenge of security and privacy, considering that IIoT devices exist as nodes on the network, hence just like other network nodes, can be subject to adversarial attacks. We also have the challenge of coexistence and interoperability given that a typical IIoT system consists of many coexisting devices deployed in close proximity in the limited spectrum. As such, a future work would include a deeper analysis on how to prevent as well as mitigate the effect of such challenges on our proposed solution. Another potential future work would be to develop a unified standard for seamlessly combining these technologies for such IIoT applications. This would require a close collaboration with all stakeholders.

VII. CONCLUSION

In this paper we proposed an efficient and dynamic resource management approach for IIoT leveraging SAT constraint modeling. We further proposed the integration with SDN along with cloud and edge computing for true realization of the dynamic network environment required for Industry 4.0 and beyond. We called this approach SDRM - A Software Defined Resource Management model for industrial IoT solutions. We implemented our model on Savile Row using ESSENCE PRIME programming language. We highlighted the importance of this approach not only in ensuring optimal resource allocation for IIoT but also ensuring that these solutions are provided in a dynamic and timely fashion.

Based on the outcome of our implementations, we conclude that the SAT constraint modelling approach is well poised to handle resource allocation for the dynamic and increasingly heterogeneous IIoT use cases in Industry 4.0. The outcome in terms of the number of solver nodes, the Savile Row total time, as well as the solver total times underscores the efficiency of this approach. SDN integration also demonstrated significant performance improvements with regards to power consumption and processing times for different VM allocation schemes. Our results also showed the benefits of edge-cloud integration with regards to CPU utilization, network utilization as well as task success rates. With

edge-cloud implementations showing substantial improvements for these performance metrics, especially the tasks success rates.

REFERENCES

[1] T. Lins, R. A. R. Oliveira, L. H. Correia, and J. S. Silva, "Industry 4.0 retrofitting," in *Proc. 8th Brazilian Symp. Comput. Syst. Eng. (SBESC)*, Nov. 2018, pp. 8–15.

[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018.

[3] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial Internet of Things (IIoT): An analysis framework," *Comput. Ind.*, vol. 101, pp. 1–12, Oct. 2018.

[4] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. Vasilakos, "Software-defined industrial Internet of Things in the context of industry 4.0," *IEEE Sensors J.*, vol. 16, no. 20, pp. 7373–7380, Oct. 2016.

[5] C. Arnold, D. Kiel, and K.-I. Voigt, "How the industrial Internet of Things changes business models in different manufacturing industries," *Int. J. Innov. Manage.*, vol. 20, no. 8, Dec. 2016, Art. no. 1640015.

[6] I. Ahmad, S. Shahabuddin, H. Malik, E. Harjula, T. Leppanen, L. Loven, A. Anttonen, A. H. Sodhro, M. M. Alam, M. Juntti, A. Yla-Jaaski, T. Sauter, A. Gurtov, M. Ylianttila, and J. Riekki, "Machine learning meets communication networks: Current trends and future challenges," *IEEE Access*, vol. 8, pp. 223418–223460, 2020.

[7] L. R. Saragih, M. Dachyar, T. Y. M. Zagloel, and M. Satar, "The industrial IoT for nusantara," in *Proc. IEEE Int. Conf. Internet Things Intell. Syst. (IOTAIS)*, Nov. 2018, pp. 73–79.

[8] *IoT Signals Report*, Microsoft Hypothesis, Microsoft Azure, vol. 2, 2020, pp. 8–10. [Online]. Available: https://azure.microsoft.com/en-us/resources/iot-signals/

[9] M. Bayern. (2019). *Industrial IoT Market Will Hit $922B by 2025, Driven by Cost Savings and Availability*. [Online]. Available: https://www.techrepublic.com/article/industrial-iot-market-will-hit-922%b-by-2025-driven-by-cost-savings-and-availability/

[10] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 27–51, Jun. 2015.

[11] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Commun. Mag.*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

[12] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2961–2991, 4th Quart., 2018.

[13] A. T. Velte, T. J. Velte, R. C. Elsenpeter, and R. C. Elsenpeter, *Cloud Computing: A Practical Approach*. New York, NY, USA: McGraw-Hill, 2010.

[14] Y. Hu, M. Song, and T. Li, "Towards, 'full containerization' in containerized network function virtualization," in *Proc. 22nd Int. Conf. Architectural Support Program. Lang. Operating Syst.*, 2017, pp. 467–481.

[15] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.

[16] D. Bringhenti, G. Marchetto, R. Sisto, F. Valenza, and J. Yusupov, "Automated optimal firewall orchestration and configuration in virtualized networks," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2020, pp. 1–7.

[17] R. M. Shukla, S. Sengupta, and M. Chatterjee, "Software-defined network and cloud-edge collaboration for smart and connected vehicles," in *Proc. Workshop Program 19th Int. Conf. Distrib. Comput. Netw.*, Jan. 2018, pp. 1–6.

[18] Z. Zhang, C. Wang, C. Gan, S. Sun, and M. Wang, "Automatic modulation classification using convolutional neural network with features fusion of SPWVD and BJD," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 3, pp. 469–478, Sep. 2019.

[19] X. Lai, Q. Hu, W. Wang, L. Fei, and Y. Huang, "Adaptive resource allocation method based on deep q network for industrial Internet of Things," *IEEE Access*, vol. 8, pp. 27426–27434, 2020.

[20] W. Li, B. Wang, J. Sheng, K. Dong, Z. Li, and Y. Hu, "A resource service model in the industrial IoT system based on transparent computing," *Sensors*, vol. 18, no. 4, p. 981, Mar. 2018.

[21] S. I. Wayer and A. Reichman, "Resource management in satellite communication systems: Heuristic schemes and algorithms," *J. Electr. Comput. Eng.*, vol. 2012, pp. 1–10, Jan. 2012.

[22] L. Chen, P. Zhou, L. Gao, and J. Xu, "Adaptive fog configuration for the industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4656–4664, Oct. 2018.

[23] M. Ghobaei-Arani, S. Jabbehdari, and M. A. Pourmina, "An autonomic approach for resource provisioning of cloud services," *Cluster Comput.*, vol. 19, no. 3, pp. 1017–1036, 2016.

[24] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Trans. Mobile Comput.*, vol. 15, no. 10, pp. 2398–2410, Oct. 2016.

[25] Q. Li, L. Zhao, J. Gao, H. Liang, L. Zhao, and X. Tang, "SMDP-based coordinated virtual machine allocations in cloud-fog computing systems," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1977–1988, Jun. 2018.

[26] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu, "IRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 7011–7024, Aug. 2019.

[27] T. Zhao, S. Zhou, L. Song, Z. Jiang, X. Guo, and Z. Niu, "Energy-optimal and delay-bounded computation offloading in mobile edge computing with heterogeneous clouds," *China Commun.*, vol. 17, no. 5, pp. 191–210, May 2020.

[28] Y. Wang, H. Ge, A. Feng, W. Li, L. Liu, and H. Jiang, "Computation offloading strategy based on deep reinforcement learning in cloud-assisted mobile edge computing," in *Proc. IEEE 5th Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Apr. 2020, pp. 108–113.

[29] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[30] X. Chen, S. Tang, Z. Lu, J. Wu, Y. Duan, S.-C. Huang, and Q. Tang, "iDiSC: A new approach to IoT-data-intensive service components deployment in edge-cloud-hybrid system," *IEEE Access*, vol. 7, pp. 59172–59184, 2019.

[31] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[32] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, 4th Quart., 2015.

[33] S. Namal, I. Ahmad, A. Gurtov, and M. Ylianttila, "SDN based inter-technology load balancing leveraged by flow admission control," in *Proc. IEEE SDN Future Netw. Services (SDNFNS)*, Nov. 2013, pp. 1–5.

[34] I. Ahmad, M. Liyanage, S. Namal, M. Ylianttila, A. Gurtov, M. Eckert, T. Bauschert, Z. Faigl, L. Bokor, E. Saygun, H. A. Akyıldız, O. L. Perez, M. U. Itzazelaia, B. Ozbek, and A. Ulas, "New concepts for traffic, resource and mobility management in software-defined mobile networks," in *Proc. 12th Annu. Conf. Wireless On-Demand Netw. Syst. Services (WONS)*, Jan. 2016, pp. 1–8.

[35] J. Okwuibe, J. Haavisto, E. Harjula, I. Ahmad, and M. Ylianttila, "SDN enhanced resource orchestration of containerized edge applications for industrial IoT," *IEEE Access*, vol. 8, pp. 229117–229131, 2020.

[36] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Towards software defined cognitive networking," in *Proc. 7th Int. Conf. Technol., Mobility Secur. (NTMS)*, Jul. 2015, pp. 1–5.

[37] Z. Lv and W. Xiu, "Interaction of edge-cloud computing based on SDN and NFV for next generation IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5706–5712, Jul. 2020.

[38] R. K. Das, N. Ahmed, F. H. Pohrmen, A. K. Maji, and G. Saha, "6LE-SDN: An edge-based software-defined network for Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7725–7733, Aug. 2020.

[39] L. Carnevale, A. Celesti, A. Galletta, S. Dustdar, and M. Villari, "From the cloud to edge and IoT: A smart orchestration architecture for enabling osmotic computing," in *Proc. 32nd Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, May 2018, pp. 419–424.

[40] K. Papadakis-Vlachopapadopoulos, I. Dimolitsas, D. Dechouniotis, E. E. Tsiropoulou, I. Roussaki, and S. Papavassiliou, "On blockchain-based cross-service communication and resource orchestration on edge clouds," *Informatics*, vol. 8, no. 1, 2021, p. 13.

[41] Y. Duan, G. Fu, N. Zhou, X. Sun, N. C. Narendra, and B. Hu, "Everything as a service (XaaS) on the cloud: Origins, current and future trends," in *Proc. IEEE 8th Int. Conf. Cloud Comput.*, Jun. 2015, pp. 621–628.

[42] A. M. Frisch, W. Harvey, C. Jefferson, B. Martínez-Hernández, and I. Miguel, "Essence: A constraint language for specifying combinatorial problems," *Constraints*, vol. 13, no. 3, pp. 268–306, Sep. 2008.

[43] P. Nightingale, Ö. Akgün, I. P. Gent, C. Jefferson, and I. Miguel, "Automatically improving constraint models in savile row through associative-commutative common subexpression elimination," in *Proc. Int. Conf. Princ. Pract. Constraint Program.* Cham, Switzerland: Springer, 2014, pp. 590–605.
[44] P. Nightingale, P. Spracklen, and I. Miguel, "Automatically improving SAT encoding of constraint problems through common subexpression elimination in Savile row," in *Proc. Int. Conf. Princ. Pract. Constraint Program.* Cham, Switzerland: Springer, 2015, pp. 330–340.
[45] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw. (Hotnets)*, 2010, pp. 1–6.
[46] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, "CloudSimSDN: Modeling and simulation of software-defined cloud data centers," in *Proc. 15th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2015, pp. 475–484.
[47] J. Wei, S. Cao, S. Pan, J. Han, L. Yan, and L. Zhang, "SatEdgeSim: A toolkit for modeling and simulation of performance evaluation in satellite edge computing environments," in *Proc. 12th Int. Conf. Commun. Softw. Netw. (ICCSN)*, Jun. 2020, pp. 307–313.
[48] N. Eén and N. Sörensson, "Translating pseudo-Boolean constraints into SAT," *J. Satisfiability, Boolean Model. Comput.*, vol. 2, nos. 1–4, pp. 1–26, 2006.

JUDE OKWUIBE received the B.Sc. degree in telecommunications and wireless technologies from American University of Nigeria, Yola, in 2011, and the master's degree in wireless communications engineering from the University of Oulu, Finland. He is currently pursuing the Ph.D. degree in communications engineering with the University of Oulu Graduate School (UniOGS), Finland. His current research interests include software defined networking, 5G and future networks, edge computing, the Internet of Things, the Industrial Internet of Things, container orchestration, SDN, network security, and biometric verifications.

JUUSO HAAVISTO received the joint M.Sc. degree (expected) from Université de Lorraine, France (formal reasoning), and the University of St Andrews, Scotland (software engineering), in July 2021. He is currently pursuing the Erasmus Mundus Joint Master Degree (EMJMD) in advanced systems dependability with the University of St Andrews, under the full scholarship. He will continue to doctoral studies, in fall 2021, to study total functional programming languages on single instruction and multiple data (SIMD) parallel computers. He worked for startups in Silicon Valley, published two first-authored articles with practical artifacts on 5G edge computing, and operated a limited liability company since the age of 17, specializing in software contracting using Go and proof-of-concept development in the IoT. Y Combinator, a startup incubator in Silicon Valley, invited him to on-site interviews (acceptance rate 4%) over Periferia, a project about developer tools for reducing 5G network end-to-end latency, in 2018.

IVANA KOVACEVIC (Member, IEEE) received the bachelor's degree in electronics and telecommunication engineering from the University of Belgrade, Serbia, in 2012, and the M.Sc. degree in communications engineering from the University of Oulu, Finland, in 2015, where she is currently pursuing the Ph.D. degree in wireless networks. She joined the Centre for Wireless Communications, University of Oulu, in 2015. Her research interests include network slicing, low-latency communications, radio resource management, edge computing, network optimization theory, game theory, and machine learning.

ERKKI HARJULA (Member, IEEE) received the M.Sc. and D.Sc. degrees from the University of Oulu, Finland, in 2007 and 2016, respectively. He works as an Assistant Professor (tenure track) with the Centre for Wireless Communications—Networks and Systems (CWC-NS) Research Group, University of Oulu. He focuses on wireless system level architectures for future digital healthcare, where his key research topics are wrapped around intelligent trustworthy distributed IoT and edge computing. He has background in the interface between computer science and wireless communications, such as mobile and the IoT networks, distributed networks, cloud and edge computing, and green computing. He has also long experience as a research project manager.

IJAZ AHMAD received the M.Sc. and Ph.D. degrees in wireless communications from the University of Oulu, Finland, in 2012 and 2018, respectively. He has been a Postdoctoral Fellow with the Centre for Wireless Communications, Oulu, from 2018 to 2019. He was a Visiting Scientist with Aalto University, Finland, in 2018, and TU Vienna, Austria, in 2019. Since 2019, he has been working as a Research Scientist with the VTT Technical Research Centre of Finland. His research interests include 5G and 6G, 5G/6G Security, the IoT, and the application of machine learning in wireless networks. He was a recipient of several awards, including the Nokia Foundation, Tauno Tönning, and Jorma Ollila grant awards, and two IEEE best paper awards.

JOHIRUL ISLAM (Graduate Student Member, IEEE) received the bachelor's degree in information and communication technology from Mawlana Bhashani Science and Technology University, Bangladesh, in 2014, and the master's degree in wireless communications engineering from the University of Oulu, Finland, in 2019. He is currently doing his doctoral research under the supervision of Asst. Prof. Erkki Harjula at the Centre for Wireless Communications—Networks and Systems (CWC-NS) Research Group, University of Oulu. His research interests include the Internet of Things (IoT), cloud and edge computing, and virtualization technologies for intelligent environment.

MIKA YLIANTTILA (Senior Member, IEEE) received the M.Sc., Dr.Sc., and eMBA degrees, and the Ph.D. degree in communications engineering from the University of Oulu, Finland, in 2005. He is currently a full-time Associate Professor (tenure track) with the Centre for Wireless Communications (CWC), Faculty of Information Technology and Electrical Engineering (ITEE), University of Oulu. He also leads NSOFT (Network security and softwarization) Research Group, CWC Networks and Systems Research Unit, which studies and develops secure, scalable, and resource-efficient techniques for 5G and beyond 5G systems. He is also the Director of communications engineering doctoral degree program. Previously, he was the director of the Center for Internet Excellence, from 2012 to 2015; the Vice Director of the MediaTeam Oulu research group, from 2009 to 2011; and a Professor (pro tem) in computer science and engineering, from 2005 to 2010. He has coauthored more than 180 international peer-reviewed articles. His research interests include network security, edge computing, network virtualization, and software-defined networking. He is an Editor of *Wireless Networks* and an Associate Editor of IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.