Noname manuscript No.
(will be inserted by the editor)

Building a Blockchain-based Decentralized Ecosystem for Cloud and Edge Computing: an ALLSTAR approach and empirical study

Huan Zhou · Zeshun Shi · Xue Ouyang · Zhiming Zhao

Received: date / Accepted: date

Abstract Cloud computing has been one of the disruptive technologies to change the traditional application operation for the last decades. The success of Cloud boosts ever more newly-built data centers. Although these data centers are distributed all around the world, the computing resources are managed in a relatively centralized manner within one big data center. For a specific small area, the centralized Cloud lacks the dispersion to satisfy the requirements of collaborative applications, e.g., the nearest data center might still be too far to satisfy the network latency. Through spreading the computing resources at the edge of the network, the emerging Edge computing can complete the data processing before uploading to Cloud. However, Edge computing still stays at the conceptual and experimental stage. Trust and incentive model are missing to motivate the Edge node and micro Cloud owners to share the computing infrastructure resources for building a more generalized and decentralized ecosystem. Traditional method of building trust through authority is not applicable in current edge environment, which is more like peer-to-peer relationship between the customer and provider. To tackle this issue, ALLSTAR

The first two authors contribute equally to this paper.
The last two authors are corresponding authors.

Huan Zhou
School of Computer Science, National University of Defense Technology, Changsha, China
E-mail: huanzhou@nudt.edu.cn

Zeshun Shi
Multiscale Networked System group, University of Amsterdam, Amsterdam, The Netherlands E-mail: z.shi2@uva.nl

Xue Ouyang
School of Electronic Science, National University of Defense Technology, Changsha, China
E-mail: ouyangxue08@nudt.edu.cn

Zhiming Zhao
Multiscale Networked System group, University of Amsterdam, Amsterdam, The Netherlands E-mail: z.zhao@uva.nl

is proposed, which is a blockchain-based approach to enhance the trust for equally combining all the Cloud and Edge resources to be seamlessly leveraged by the application. The ALLSTAR approach is a systematic solution to realize decentralized resource management, including Cloud and Edge resource sharing and trading, and target at building the trustworthy ALLSTAR ecosystem. In this paper, we first analyze the challenges of utilizing distributed Cloud and Edge resources, and describe the overall architecture of ALLSTAR, including the related key techniques, detailed application development and operations processes as well as the new business model. Moreover, an empirical study on the permissioned blockchain evaluation is conducted. The study not only demonstrates the ALLSTAR approach is feasible but also provides insights of which blockchain to choose when constructing such an ecosystem.

Keywords Blockchain · DevOps · Edge Computing · Resource Management

1 Introduction

As Web 3.0 rears into action, decentralization of core services across every facet of online activities is in the need for a powerful, open, data-driven, user-centric, interoperable platform or ecosystem to operate emerging applications Jeferry et al. (2015). In this context, with the wide adoption of novel paradigms such as Internet of Things (IoT), robotics, and crowdsourcing, ever more applications require operation at a larger scale in both involved users (e.g., cooperative crowd storytelling) and required infrastructure (e.g., mixing networked high-end servers and low-end client devices). Such Next Generation Internet (NGI)[1] collaborative applications exhibiting high business values (e.g., precise service recommendation) and essential societal impacts (e.g., robotics in health care) require critical runtime time constraints (e.g., sensitive latency in immersive virtual realities (VR)), intensive artificial intelligence (AI) (e.g., processing natural languages during social interactions), and strict trust (e.g., using self-made media via social networks).

Cloud computing has been a major disruptive technology in the last decade providing resources-as-a-service for diverse Internet applications Buyya et al. (2009). According to Gartner's report, it is poised to grow by around 27.5% and expected to reach $1,250 billion by 2025. While Cloud offers elastic capacity and customizable connectivity over a large-scale network, the resilience, sustainability and human-centric collaborative requirements of NGI applications demand an interoperable end-to-end ecosystem design that pushes the infrastructure services, traditionally bounded within big data centers, towards remote nodes closer to the data sources. These concerns are partially addressed by federating Cloud services with emerging Edge and Fog computing paradigms with reduced overheads of transferring distributed data into remote data centers Shi et al. (2016). Since the data can be processed in the Edge or Fog nodes, the pressure of Cloud can be reduced and the application can also

[1] https://www.ngi.eu/about/

retrieve response faster. However, most of the Edge devices are maintained by a specific organization, or application developers and operators construct their own Edge nodes and devices. For example, the Edge resources are provided by base stations and are owned by the Shanghai Telecom company in Guo et al. (2020). By contrast, application operators utilize their own local cluster to work as Edge resources in Qian and Andresen (2016). Teerapittayanon et al. (2017) propose to deploy the neural network across multiple Edge devices, such as sensors and cameras, in a distributed manner. The issue is that these devices must be physically owned by the application developer to control and operate. Hence, the lack of an effective mechanism to motivate the Edge owner to share the resource hinders the Edge application developer to further utilize resources distributed around the world.

This paper introduces the ALLSTAR approach, a blockchain-based ecosystem to provide a solution for achieving collaborations among Cloud/Edge resource providers and application developers/operators. The goal of this solution is to leverage the blockchain as the underlying support to enhance the trust among the resource providers and application developers when further motivating the Edge device owners to share their resources. We further conduct an empirical study on the evaluation of scalability, stability, and resource consumption of different blockchain platforms, which demonstrates the feasibility of the ALLSTAR approach.

In the rest of this paper, we demonstrate the challenges of utilizing current Cloud and Edge resources to orchestrate NGI collaborative applications as well as related works in Section 2. To address the challenges, we propose the ALLSTAR approach and introduce the detailed architecture components in Section 3. We also presents the business model and the application DevOps lifecycle between resource providers and application developers/operators when adopting ALLSTAR approach in the same section. Furthermore, Section 4 conducts an empirical study on blockchain to demonstrate the feasibility of utilizing blockchain as the fundamental layer. Finally, Section 6 concludes the paper.

2 Challenges and Related Works

Traditional Clouds are maintained by several well-known providers, such as Amazon, Google. Although their data centers are distributed around the world, the management method within one data center is still centralized. Especially, the distribution of the data centers is still limited. Zooming in to a small region, the resources within one data center are still too centralized to satisfy the application requirements, such as low latency, fast response, etc. Therefore, we illustrate the challenges of developing and operating applications within such a complex computing environment as the underlying infrastructure in Figure 1. The green part of the figure represents the state of the art of orchestrating applications with the current Cloud computing infrastructure. When further considering to add the more decentralized Edge resources as the underlying infrastructure, we identified five critical barriers in the current Cloud envi-

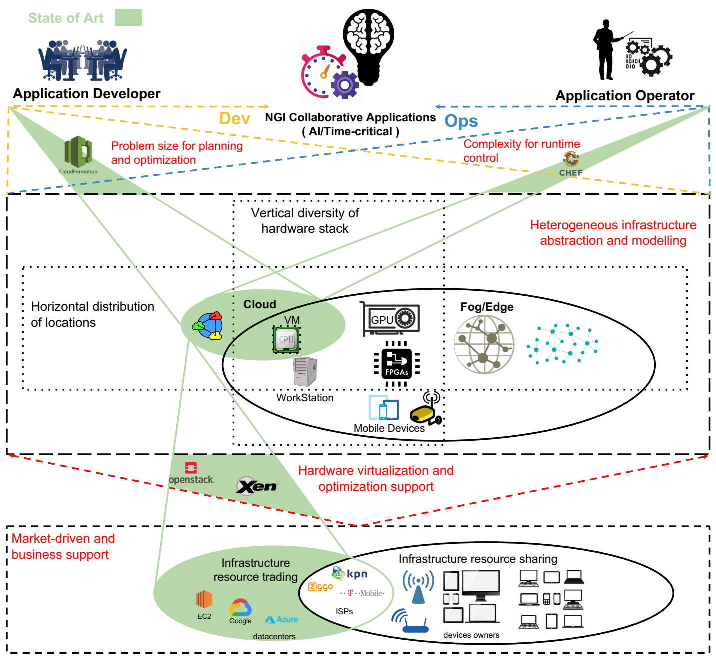

State of Art
Application Developer
NGI Collaborative Applications (AI/Time-critical)
Application Operator
Dev
Ops
Problem size for planning and optimization
Complexity for runtime control
CHEF
Vertical diversity of hardware stack
Heterogeneous infrastructure abstraction and modelling
Horizontal distribution of locations
Cloud
VM
GPU
FPGAs
WorkStation
Fog/Edge
Mobile Devices
openstack.
Xen
Hardware virtualization and optimization support
Market-driven and business support
Infrastructure resource trading
Infrastructure resource sharing
kpn
Ziggo
T-Mobile
ISPs
EC2
Google
Azure
datacenters
devices owners

Fig. 1: The state of the art of the application DevOps in current Cloud computing environment and challenges when further considering Edge resources

ronments that hinder the development, management and operation of NGI collaborative applications. We demonstrate these barriers by considering a crowd engaged live video and real-time storytelling use-case, the goal of which is for elastic collection and distribution of collaborative, trusted, high-quality live content and story opinions from a massive number of dynamically engaged crowd participants across hostile network channels. This use case is provided by an information broadcast and video encoding company called MOG Technologies SA from Portugal.

Barrier 1: Difficulty to develop and operate (DevOps) applications over heterogeneous infrastructures across distributed Cloud data centers and Edge devices. The MOG development team needs to keep their live event and storytelling service continuously online when incorporating dynamic customer requirements and feedback. Unfortunately, current Cloud DevOps solutions fail in automating continuous testing, integration, deployment and monitoring of distributed applications over heterogeneous resources, such as Clouds (e.g., for video processing and switching), Edge (e.g., for video and user contribution pre-checking), and mobile devices (e.g., media rewarding), due to diverse resource management models and interfaces.

Barrier 2: Difficulty to guarantee application and system level performance over heterogeneous infrastructures. For business-critical customers, MOG needs to ensure the performance of the application and deliver a guaranteed service quality (QoS). However, the diverse abstraction models on resource performance and data access constraints, such as security and privacy, result in inconsistent views from different providers. MOG cannot effectively plan its infrastructure capacity across provider boundaries to schedule and balance task loads among distributed Edge nodes and Cloud data centers, for effectively runtime handling of the dynamic data, energy, cost and security constraints.

Barrier 3: High complexity in controlling infrastructures consisting of heterogeneous distributed Cloud data centers and Edge resources. When system performance degrades (due to imbalanced load, device defects or security attacks), the high complexity of the heterogeneous infrastructure hampers MOG from making real-time decisions to adapt the infrastructure, especially when considering performance, energy, security, and cost, together with the lack of effective formal control verification mechanisms.

Barrier 4: Difficulty to utilize advanced hardware accelerators in virtualized infrastructures. MOG developers heavily rely on natural language processing and deep neural networks to process the collected stories. Advanced specialized hardware like GPU and FPGA can largely accelerate the computation of such machine learning (ML) tasks but is only available in data centers of few public providers. MOG lacks a flexible and optimized commodity solution for accelerating its data processing, ML, and other resource-intensive tasks, capable of considering different hardware characteristics and virtual infrastructure constraints.

Barrier 5: Insufficient business support for providers in building on-demand trustworthy resource federations. MOG developers often need to consider not only services from different Cloud providers, but also special remote Edge devices deployed close to the event site where the live video platform operates. The current centralized quality control mechanisms available in the Cloud, built upon several layers of abstraction, do not suffice, hampering MOG from effectively federating heterogeneous resources from different providers. MOG strives for a trustworthy solution supported by a proper business model underneath that dynamically federates high quality and reliable resources on-demand in response to its real-time application demands.

In fact, there are plenty of tools and academic research working on how to utilize the Cloud resources. For example, OpenNebula[2] or OpenStack[3], provide different virtual infrastructure functions, known as APIs (Applications Programming Interfaces), to access physical resources through the hardware virtualization, while they are not targeting at managing Edge resources. On the other hand, academic works mainly focus on Cloud resource allocation and scheduling. Venkateswaran and Sarkar (2018) optimize the application deployment process on hybrid Clouds and achieve the best-fit hosting combination. Wang et al. (2011) enhance the makespan and the reliability of a workflow application through evaluating the Cloud resource reputation. Zi-

[2] https://opennebula.org/
[3] https://www.openstack.org/

afat and Babamir (2018) develop the algorithm selecting a proper Cloud to run the task according to the data center geolocation. All these works lack an efficient solution for the application to program and control the computing infrastructure in the DevOps lifecycle. Although our previous work CloudsStorm framework tackling the resource management issue from the application DevOps perspective Zhou et al. (2018, 2019a), it still mainly focuses on working with Cloud resources, which is insufficient for a more complex Edge computing environment.

At runtime, to ensure the service quality, SLA (Service Level Agreement) is leveraged by Cloud computing for compensating the customer when the service quality is not met Faniyi and Bahsoon (2015). However, this agreement is based on the trust that the Cloud provider would behave honestly. However, Edge providers or micro Cloud providers in the new computing environment have the peering relationship with the customer Wang et al. (2020); Higuchi et al. (2018). Hence, the providers do not have the sufficient capability to endorse their services.

Blockchain is the technology proposed to solve the trust concern Nofer et al. (2017). It is an innovative technology to make every participant having trust in a decentralized ledger through a consensus algorithm, such as Proof of Work (PoW) Nakamoto (2008), and Practical Byzantine Fault Tolerance (PBFT) Castro and Liskov (1999). The blockchain using the PoW type of consensus algorithm is classified as permissionless blockchain, which is open to allow any participant to join. The blockchain using the PBFT type of consensus algorithm is classified as permissioned blockchain, which only allows a fixed set of participants to join. Moreover, the blockchain-based smart contract technique is developed to enable the application execution on a blockchain platform Buterin (2014). Its execution process can benefit from the blockchain features and, therefore, can be trusted.

To enhance the trust between the provider and the customer, there are also some initial efforts paid in the Cloud environment to apply these blockchain related techniques. For instance, Nakashima and Aoyama (2017) automate the Cloud SLA enforcement using blockchain. We also previously proposed the witness model in Zhou et al. (2019b); Uriarte et al. (2020) to improve the trustworthiness of demonstrating whether a Cloud service violation really happens. However, a comprehensive and systematic consideration for establishing collaborations among different roles of the ecosystem is still missing, especially when Edge resources are involved in the ecosystem.

3 The ALLSTAR Approach

3.1 Design Requirements

By analyzing the current barriers, we firstly highlight a number of requirements when designing an approach to address these issues:

- Resilient and seamless management across data center VMs (e.g., running heavy machine learning tasks), Edge computers (e.g., handling communication among mobile devices collecting live videos from users), and mobile devices (e.g., processing user inputs or lightweight learning processing), required to facilitate the application operation;
- Effective infrastructure adaptation during the crowd journalism application operation for manipulating the network topology and resources, and relocating tasks (e.g., when dealing with security attacks), alongside Cloud services scaling and load balancing;
- Data privacy and security ensured when processing data (e.g., personal information), especially when coming from distributed sources (e.g., from crowd users or robots);
- Critical time constraints required by crowd journalism applications involving user interactions (e.g., in the immersive VR) or decision making (e.g., in the business recommendation);
- SLAs required by all applications and guaranteed as QoS metrics by resource providers (even by multiple providers jointly hosting a single distributed crowd journalism application);
- Trustworthy service trading ecosystem based on the blockchain technology for sharing digital assets (e.g., individual story contributions) or developer media with use tracking;
- Hardware characteristics, in particular accelerators, required in the virtualized environment to optimize the training of the machine learning tasks such as neural networks.

These requirements cover the key aspects in the development and operation lifecycle of the MOG's crowd journalism use case, including their virtual hosting infrastructure and their service management model. To solve these requirements, the current Internet, the Cloud, Fog, and Edge computing paradigms, blockchain, AI, ML, and the Cloud DevOps software engineering concept are important starting points. However, these technologies currently focus on different aspects and do not seamlessly work in a unified ecosystem. Therefore, we propose ALLSTAR architecture to unify them as a single integrated approach and to guarantee a flexible approach for citizen-based journalism use case in MOG. The objective of "ALLSTAR" is to flatten the computing architecture of the infrastructure and make Edge resources equal to Cloud resources from the business perspective. Hence, more Edge resource will be motivated to adopt our approach and freely join the ecosystem. With their contributions in providing a more decentralized computing infrastructure, the requirements of the MOG use case can be satisfied. Since the computing resources in the environment are organized in a decentralized manner, the traditional centralized solutions cannot operate so many heterogeneous resources from different providers, and also cannot provide the trust for all the providers and customers. Because traditional public cloud providers are endorsed by their companies. However, the relationship between micro cloud/edge providers and customers is more like peer-to-peer. The customer cannot easily believe the provider would offer the guaranteed service as the customer requests. To tackle this issue, traditional method is through the endorsement by a trusted authority. This method is not practical, since there is usually no trusted authority in our edge environment. On the other hand, blockchain is an emerging technology to build trust in such a distributed environment, which naturally fits in this scenario Zheng et al. (2018). Therefore, we consider leveraging blockchain to construct the trust layer for the edge environment.

3.2 Architecture Overview

As mentioned above, a more decentralized "Cloud+Edge" ecosystem requires new business models and application DevOps procedures, considering that the resources are complex and heterogeneous. Therefore, the proposed architecture targets at solving following four challenging questions:

1) How to describe diverse resources and services in the ecosystem?

2) How to use a descriptive model to build a seamless DevOps solution?

3) How to consider hardware features to optimize application function containerization and deployment?

4) How to provide a decentralized environment for sharing digital assets (e.g., infrastructure resources, software services, data)?

Hence, ALLSTAR designs four subsystems in response to above questions, as shown in Figure 2.

1. Heterogeneity-AS-code toolKIT (**HASKIT**) provides an open modeling language for describing heterogeneous resources and services in the ALL-

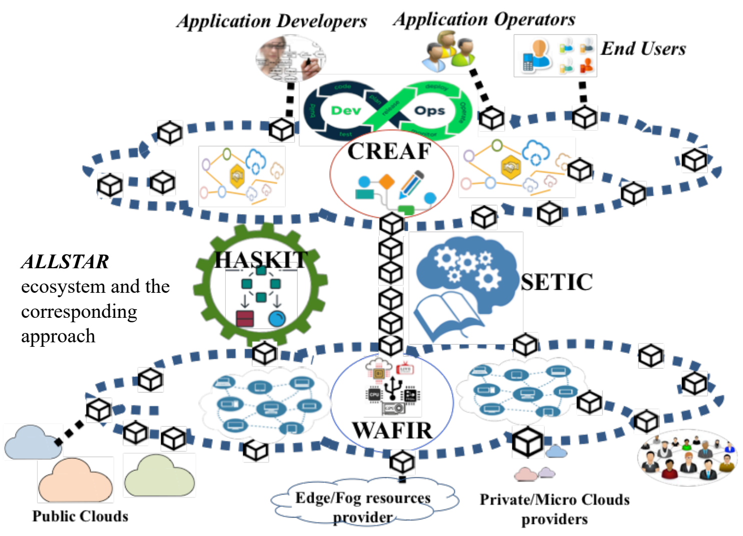


Fig. 2: The architecture overview of the ALLSTAR approach and the related roles of the corresponding ecosystem

STAR ecosystem, including both functional (e.g., application components and network topology) and non-functional properties (e.g., QoS, QoD, and data accessibility constraints) of an NGI collaborative application at a different stage of its lifecycle. Moreover, it provides tools to other subsystems to validate the specification composed by application developers, make control decisions, and analyze the service reputations during the operation of the application.
2. CRoss-Edge seamless infrAstructure orchestration Framework (**CREAF**) provides tools and APIs (Application Programming Interface) for application developers to plan the necessary capacity for an application, automate the provisioning of the underlying virtual infrastructure and the deployment of application components. The CREAF subsystem also provides tools to monitor, diagnose, and control the application at runtime.
3. HardWare chAracterized Function vIRtualization framework (**WAFIR**) provides a framework that focuses on hardware accelerators (e.g., GPU and FPGA) and OS-level (Operating System) virtualization (e.g., through containers) for meeting application QoS concerns. The framework also provides special care on security isolation and energy optimization.
4. Service fEderation defined Trustworthy Inter-Chain platform (**SETIC**) provides a blockchain-based decentralized fabric for the ALLSTAR ecosystem to record the service transactions among providers and consumers, enforces the SLAs using smart contracts, and provides secure-by-design services for naming, lookup, resource discovery, and querying acquired knowledge.

3.3 Technical Details

The ALLSTAR software architecture consists of a novel combination of state-of-the-art technologies in DevOps, Cloud, Fog, and Edge computing, Blockchain. For each component mentioned in the previous section, we present the detailed description of sub-components and related technologies. The details are shown in Figure 3 using different colored boxes, where each colored box represents the main component.

3.3.1 Heterogeneity-AS-code toolKIT (HASKIT)

The Heterogeneity-AS-code toolKIT (HASKIT) provides: 1) an open description language called ALLSTAR Modelling Language (ALLSTAR-ML); 2) a number of tools for validating the model; 3) decision making mechanisms for applications runtime control; 4) reputation auditing for the participants in the ecosystem.

– *ALLSTAR Modeling Language* (ALLSTAR-ML) provides an open description language for modeling not only the key elements in both NGI applications (e.g., services, components, functions, and dependencies among them)

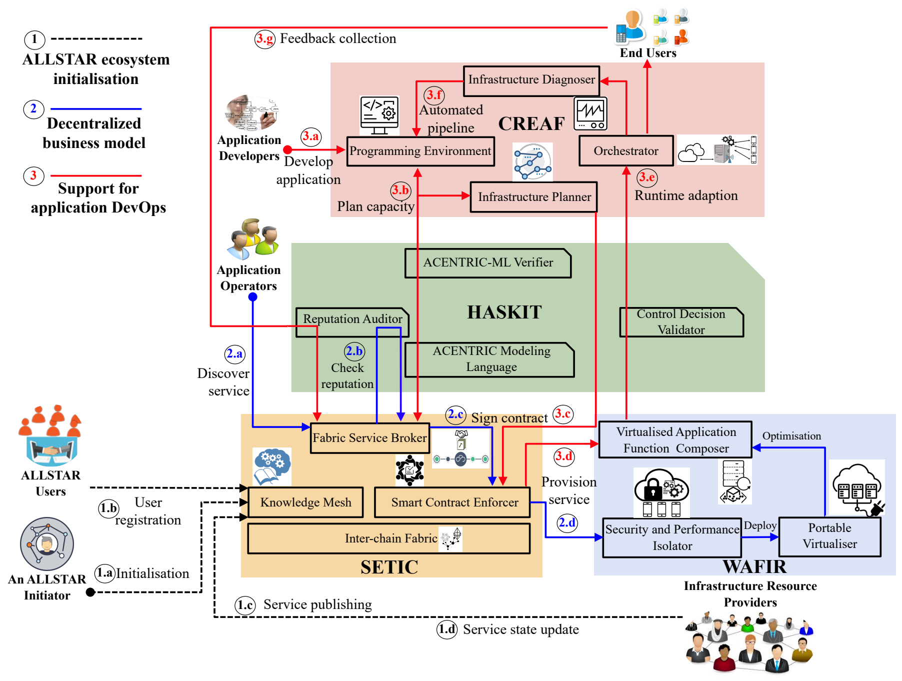


Fig. 3: Detailed sub-components description and key interaction lifecycles (business model and DevOps) of ALLSTAR approach

and infrastructures (e.g., resource types, devices and network topologies), but also the properties of these elements at different security and access constraints levels (e.g., QoS, QoE, and QoD). The language examines the industrial modeling standards (e.g., TOSCA) and Cloud ontologies (e.g., INDL, mOSAIC) for describing Cloud-Edge computing infrastructures and extends them with flexible semantic linking to effectively capture and specify properties, which are derived from new characteristics of NGI collaborative applications.

- *Interactive ALLSTAR-ML verifier* provides verification support for both functional and non-functional requirements of applications, as specified in the NGI collaborative application model. It verifies the application logic, time, quality constraints, as well as security and privacy requirements in the context of requirement modeling, application composition, provisioning, and deployment.
- *Complex NGI-Cloud control decision validator* applies complex systems theories, such as (probabilistic) Boolean networks and dynamical systems, to validate the consequences a control decision may cause on the graph representation of the runtime Cloud infrastructure status. Efficient control algorithms detect the factors driving the infrastructure into a critical state,

which may affect the overall performance and stability of the application. Furthermore, it proposes solutions in case of identified problems.
– *Reputation auditor* for ecosystem participants provides a reputation model focused on QoS experiences and history to audit the behavior of the ecosystem participants. The results are achieved not only based on end users feedback, but also on violation detection reports during the crowd-based SLA enforcement. The SETIC blockchain-based fabric explained later provides this information.

3.3.2 CRoss-Edge seamless infrAstructure orchestration Framework (CREAF)

The CRoss-Edge seamless infrAstructure orchestration Framework (CREAF) provides a unified and robust interface for agile and programmatic seamless application-infrastructure orchestration considering heterogeneity across Cloud and Edge resources. CREAF also empowers ALLSTAR with smart, automated infrastructure capacity planning, and resource management. It employs predictive analysis of application-infrastructure driven requirements to simplify scaling and provisioning with improved operational efficiency, and also minimize management costs.

– *Interactive NGI application-infrastructure programming environment* creates the description, with privacy-by-design features in accordance to GDPR guidelines, using ALLSTAR-ML and the interactive verification tool provided by HASKIT for user-centric support approaches. CREAF further leverages real-time analytics based on intelligent optimization and deep learning heuristics, to measure against infrastructure and application-driven metrics. It provides proactive deployment insights of the application and the infrastructure management is based on ALLSTAR-ML constraints and specifications. It also considers performance issues, data privacy, security, application needs, unexpected traffic spikes or even to control costs between providers across Cloud and Edge boundaries.
– *Infrastructure planner* with automated integration capability enhances the continuous provisioning, deployment, testing, and intra-service orchestration DevOps processes across the software development lifecycle. Firstly, it is able to plan the infrastructure capacity according to the application requirements. Meanwhile, it simplifies and accelerates the transition from manual to automated continuous service delivery, ensuring full capability across the federation of heterogeneous physical and virtual infrastructures, services and applications. Precisely, CREAF utilizes the existing industrial DevOps tools (e.g., Kubernetes , Puppet , Chef) and provides AI-driven services to automate the data analysis and accelerate routine operations (e.g., continuous integration, provisioning, deployment, testing, and delivery) with effective infrastructure usage and collaboration.
– *Systematic NGI application-infrastructure diagnoser*, which is fed with covariate performance metrics and measurements across heterogeneous Cloud

and Edge infrastructures, monitors and exploits the application and infrastructure orchestrated resource history as a baseline. It exploits the baseline performance against small performance deviations (e.g., network congestions, delays, and bandwidth allocations) and flags a possible heterogeneous resource anomalous condition in need of further proactive actions.
– *Seamless cross-Edge infrastructure orchestrator* considers the geographically dispersed and fragile networked infrastructures orchestration within and across Cloud data centers. It exploits open source orchestration tools (e.g., Kubernetes) and automates the operations not limited to application-driven development, production and deployment, but also incorporates infrastructure adjustments. Such adjustment owes to complex and heterogeneous architecture design integration, including microservices, hybrid Cloud, Edge computing and IoT systems.

3.3.3 hardWare chAracterized Function vIRtualization framework (WAFIR)

The hardWare chAracterized Function vIRtualization framework (WAFIR) provides hardware accelerator aware virtualization and function containerization for quality-critical NGI applications. By adopting proper level of virtualization techniques (e.g., VM, container, and unikernel), WAFIR creates self-contain portable components for application functions according to the requirements and available hardware acceleration capacity (e.g., Intel, ARM, Nvidia GPU, FPGA), and publishes them to the knowledge mesh within the fabric of SETIC (see next section) based on the ALLSTAR-ML schema. WAFIR also provides CREAF with underlying hardware acceleration support for optimizing application function deployment.

– *Hardware portable function virtualizer* supports different types of hardware, able to expose to the correct acceleration depending on the application SLA. In the particular case of FPGAs, specific hardware extensions enable dynamic resource allocation, reconfiguration and runtime migration.
– *Quality-critical and energy-aware function composer* allows application developers iteratively select the ingredients of application functions and optimize them based on the energy and performance constraints, and the availability of the hardware accelerators choosing the suitable level of virtualization (e.g., container, VM, or unikernel) for different performance and security requirements.
– *Security and performance isolator*, for mixed criticality tasks in NGI applications, isolates functional safety critical workloads from untrusted connected applications to ensure security and performance. A secure enclave will leverage CPU processor extensions, such as ARM TrustZone and Intel SGX, to ensure a secure environment.

3.3.4 Service fEderation defined Trustworthy Inter-Chain platform (SETIC)

The Service fEderation defined Trustworthy Inter-Chain platform (SETIC) provides the underlying blockchain and smart contract support for constructing the decentralized Cloud ecosystem, in which participants collaborate without relying on a centralized authority. Infrastructure providers can dynamically join and leave the ecosystem, and offer their resources as a service to the community of application developers and operators.

- *Service federation defined inter-chain fabric* mainly provides two aspects of functionalities: application DevOps management and the application execution environment. For the management perspective, the inter-chain fabric provides: 1) the underlying blockchain for transactions and interactions among the different assets providers in the Cloud ecosystem, including infrastructure resources, services and models; 2) an interoperable fabric for interconnecting different blockchains used in the ecosystem. For the execution perspective, the inter-chain fabric provides customizable blockchain-as-a-service for service federation defined applications, which can be executed as smart contracts on a new blockchain on demand. The content on the blockchain includes data, software, media content, etc. Through combining different chains, the fabric exploits the trade-off between the system performance and consensus mechanisms according to the transaction types: permissionless blockchain with more trust and less efficiency, and permissioned blockchain with conversely less trust and more efficiency instead.
- *Evolutionary knowledge mesh* manages the information and the knowledge in the decentralized ecosystem using the underlying inter-chain fabric, including 1) infrastructure provider information (e.g., prices, capacity, and special hardware feature), 2) a catalogue of virtualized application function repository for agile deployment, 3) application naming information for delivering services to end users, and 4) reputation from crowds and user feedback.
- *Crowd-based trustworthy smart contract enforcer* provides a trustworthy mechanism for service providers and consumers to automate specific service transactions, required by the applications in the ecosystem (e.g., negotiating prices, payment conditions, and violation conditions), with incentivized witnesses model to credibly feedback about the off-chain events. The results of this module also play an important role in generating the providers' reputation.
- *Context-aware fabric service broker* enables users to interact with the inter-chain fabric and knowledge mesh, for example for publishing service on the mesh and semantically discovering resources or other types of assets from the ecosystem. It can also interact with executable smart contract code for negotiating SLA, generating new smart contracts, or detecting violation during enforcement.

3.4 The Business Model and DevOps Lifecycle

ALLSTAR combines the HASKIT, CREAF, WAFIR, and SETIC subsystems into a single ecosystem with high development modularity, with each subsystem encapsulating features and modules as microservices. The integration interfaces among subsystems and microservices define a high-level abstract and generic application-programming interface (API) to ensure portability and sustainability, so that new implementations are able to interoperate with the existing ones as the technology evolves. The ALLSTAR ecosystem integration has three main scenarios: 1) decentralized ALLSTAR ecosystem instantiation; 2) decentralized business model of resource providers; 3) application development and operation in the ecosystem. We briefly explain each scenario in the following key steps.

Firstly, the decentralized ALLSTAR ecosystem instantiation sets up a working ecosystem in four steps: a) *Initialization.* The first component needs to be initialized in the ALLSTAR ecosystem is the knowledge mesh of SETIC, which can initialized by creating specialized ALLSTAR smart contract on a blockchain; b) *User registration.* The knowledge mesh provides interfaces for a blockchain participant to register his/her account (e.g., wallet address) in the ecosystem as a specific role, e.g., application developer, an application operator, resource provider, or end user; c) *Service publishing.* A provider can announce his/her services by publishing service description (using the ALLSTAR-ML) into a catalogue in the knowledge mesh; d) *Service state update.* A provider is also able to modify the statue (e.g., availability) of his/her service by setting availability state of the corresponding smart contract of the catalogue in the knowledge mesh.

Secondly, users buy services from providers by creating smart contracts of the SLA, using SETIC. The decentralized business model for buying and provisioning resources in the ecosystem has four steps: a) *Discover service.* A user (e.g., application developer or operator) can discover services using the context-aware fabric service broker provided by SETIC based on the ALLSTAR-ML schema; b) *Check reputation.* The user may invoke the HASKIT reputation auditor to check the resource provider reputation, performed without a centralized third party and with trustworthy results; c) *Sign contract.* The user (e.g., application developer) can directly buy services from a provider invoking SETIC to negotiate and sign a smart contract with the provider based on quality constraints, and to initialize crowd-based trustworthy smart contract enforcer for detecting service violation; d) *Provision service.* A service provider can utilizes the hardware portable function virtualizer of WAFIR to provision virtual resources on top of specific hardware. The resource provisioning and application function deployment is often embedded as part of the DevOps lifecycle. We will discuss them in more detail in the next scenario.

Thirdly, the DevOps application development lifecycle, embedding the resource provisioning and application function deployment, operates the application in the ecosystem in seven steps: a) *Develop application.* An application developer utilizes the interactive application-infrastructure programming environment from CREAF to customize the virtual infrastructure by invoking infrastructure information from the knowledge mesh. The developer also queries providers on their resources availability, prices, reputation, and so on; b) *Plan capacity.* CREAF performs dynamic capacity planning to optimize the application and the required virtual infrastructure, and invokes the interactive ALLSTAR-ML verifier for dynamically verifying whether the composed application and virtual infrastructure match the performance and data security constraints; c) *Sign contract.* The application developer uses SETIC to negotiate and sign smart contracts with the selected resource providers, after the application and infrastructure design; d) *Provision service.* CREAF provisions the resources and invokes the quality-critical and energy-aware function composer of WAFIR to create a suitable application function for deployment based on application requirements and available components. CREAF deploys a blockchain-as-a-service and registers it to the knowledge mesh of SETIC, if the application needs its own blockchain services; e) *Runtime adaptation.* CREAF enables the runtime application operators to diagnose the infrastructure failures according to the monitoring information and perform control decisions with support from the complex NGI-Cloud control decision validator (provided by HASKIT); f) *Automated pipeline.* CREAF continuously considers changes in the application or its virtual infrastructure and continuously automates the pipeline during the application lifecycle; g) *Feedback collection.* SETIC provides decentralized naming services to deliver the application service to the end user. Furthermore, the end user is able to feedback with the QoS experiences.

4 Empirical Study of ALLSTAR Blockchain Infrastructure

According to the design requirements, the core function of ALLSTAR is to leverage the blockchain as the underlying infrastructure to enhance the trust among the decentralized resource providers when further motivating the Edge device owners to share their resources. The HASKIT, CREAF, and WAFIR are designed and developed mainly for programming and controlling the heterogeneous Cloud and Edge resources, optimizing the application DevOps on them. To tackle this issue, we have proposed different solutions in our previous work, such as scheduling algorithms in Hu et al. (2020) Hu et al. (2018) and infrastructure resource programming frameworks in Koulouzis et al. (2020) Zhou et al. (2019a). Therefore, the blockchain as the fundamental layer of the entire architecture is the key part to support all the upper transactions. In order to demonstrate the feasibility of the ALLSTAR approach, we conduct an empirical study on different blockchain platforms to analyze their performance and provide insights for choosing the proper blockchain platform when constructing the ALLSTAR ecosystem.

4.1 Preliminaries

In the previous section, we have already discussed that the blockchain is the main component of the SETIC to provide a decentralized Cloud/Edge environment. The performance of the blockchain will directly affect the efficiency of the decentralized ALLSTAR framework when motivating resource providers to adopt our approach and join the ecosystem easily. It is, therefore, important to discuss the performance of ALLSTAR blockchain infrastructure.

In general, blockchains can be permissionless or permissioned. Platforms like Bitcoin[4] and Ethereum[5] are permissionless blockchains, which means anyone can choose to join the network. Due to the performance limitations of current permissionless blockchains, and the huge energy and money consumption when deploying and executing smart contracts, it is not desirable to implement all the functions of ALLSTAR ecosystem on the permissionless blockchain platform. By contrast, the permissioned blockchains, which have exhibited better performance and demonstrated great potential to provide trustworthy and security services in various industrial scenarios, are more suitable in our ecosystem to work as decentralized infrastructure solutions. We designed the Inter-chain fabric in the SETIC module, hoping to combine the advantages of different permissioned blockchains to achieve our goal of creating a trusted trading environment for different Cloud and Edge resource providers. To evaluate these functionalities of SETIC, we designed three experiments which will be discussed in this section.

First of all, we would like to clarify the basic assumptions and settings of our experiments.

- We would not discuss the performance of permissionless blockchain platforms because these data can be easily obtained from major monitoring protocols (e.g. Etherscan[6]). Instead, our evaluation mainly focuses on the discussion of performance comparison of different permissioned blockchain platforms under the same scenario, which is lacking in most current studies.
- Although different platforms implement their own consensus algorithms, the evaluation in this paper focuses on the comparison of different blockchain platforms and does not specifically discuss the comparison of consensus algorithms. In our previous research, we have demonstrated that there are some differences in the performance of different consensus algorithms under the same platform Shi et al. (2019). However, we believe that the impact of consensus algorithms on performance is limited by the framework itself.
- A basic smart contract was leveraged to model the decentralized Cloud/Edge market and to benchmark different blockchain platforms. The functions of this smart contract include general operations in a decentralized Cloud/Edge market, e.g., generating new transactions (write) and querying existing transactions (read). We leave the complex functional design of smart contracts as our future work.
- Regarding the benchmark metric, the performance of the blockchain is measured by the commonly adopted metric “throughput” in permissioned blockchain community. It is defined as the rate at which write/read operations are committed to the blockchain (as shown in equation 1) and

[4] https://bitcoin.org/en/
[5] https://www.ethereum.org/
[6] https://etherscan.io/

can reflect whether the underlying infrastructure (blockchain) of the decentralized Cloud/Edge solution can meet the requirements of industrial applications.

$$Write/Read\ Throughput = \frac{Total\ Write/Read\ Operations}{Total\ Time\ in\ Seconds} \tag{1}$$

– Five promising blockchain platforms were selected as our decentralized infrastructure and evaluation objects, namely Hyperledger Sawtooth, Hyperledger Iroha, Hyperledger Fabric, Hyperledger Besu, and Ethereum[7]. These platforms have been involved in many successful commercial projects. It should be noted that Ethereum is not used as a permissionless blockchain but as a private deployment platform. A more detailed comparison of those 5 platforms is illustrated in Table 1
– We used a MacBook Pro laptop as the running machine, with 2.9 GHz Intel Core i5 CPU and 16 GB 1867 MHz DDR3 memory. We used docker container to deploy our blockchain network. All of our code and experimental details are open source on Github[8].

4.2 Results

Based on the above assumptions and settings, we designed experiments with three dimensions to discuss scalability, stability, the resource consumption (CPU, memory) of various blockchain platforms. More specifically, the experiments related to scalability is to explore the blockchain performance under different transaction input patterns. Performance stability experiments leverage the same transaction input rate to test the distribution and stability of throughput results. Finally, the resource consumption experiment discussed the CPU and memory consumption of different blockchain platforms under different transaction models.

4.2.1 Scalability Evaluation

In this experiment, we used the transaction input rates that increased linearly from 10 tps to 100 tps to observe the performance scalability of the blockchain platform when facing different levels of transaction requests. In practical scenarios, it often happens that blockchain transaction requests increase rapidly in a short period of time. Therefore, such a rate control strategy can simulate the performance change of the blockchain platform when processing transaction requests with different densities.

Figure 4a shows the changes in write throughput of different blockchain platforms at different transaction input rates. When the rate increases, the

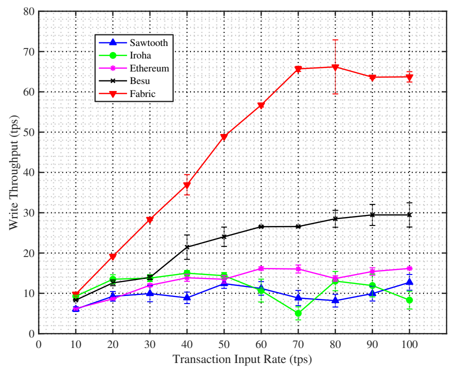
[7] For simplicity, Sawtooth, Iroha, Fabric, Besu, and Ethereum are used in the following text.
[8] https://github.com/ZeshunShi/ALLSTAR

Table 1: Comparison of Five Permissioned Blockchain Platforms

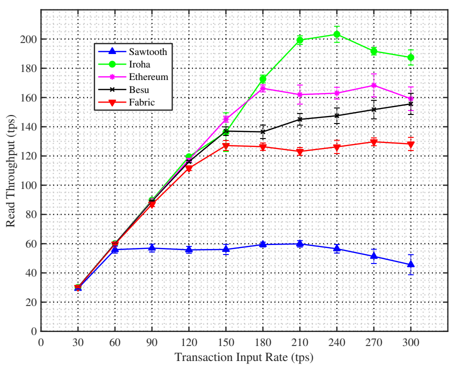
Type	**Hyperledger Sawtooth**	**Hyperledger Iroha**	**Hyperledger Fabric**	**Hyperledger Besu**	**Ethereum (Private)**
General Description	A framework for building enterprise-grade permissioned blockchain with a focus on security, scalability, and modularity.	A framework designed to be simple and easy to incorporate into infrastructure or IoT projects that require blockchain.	An open source enterprise-grade permissioned blockchain with a highly modular and configurable architecture.	An Ethereum client designed to be enterprise-friendly for both public and private permissioned network use cases.	Ethereum is a decentralized, open-source blockchain with smart contract functionality.
Consensus Protocols	Pluggable consensus algorithm e.g., Raft, PBFT, and PoET (hardware based)	A Byzantine fault tolerant consensus algorithm called YAC	Pluggable ordering service e.g., Raft and Kafka (deprecated in v2.x)	Several consensus algorithms including PoW, PoA (IBFT, IBFT 2.0, Etherhash, and Clique)	Use PoW in most cases, other options include PoS and PoA
Smart Contract	- Transaction Processor - Written in Python or Javascript	- Built-in smart contracts called "commands"	- Chaincode - Written in Golang or JavaScript - Run in Docker containers	- Smart contract - Written in Solidity or Vyper	- Smart contract - Written in Solidity or Vyper
Token	- None - Currency can build via transaction processors	- No Iroha token - Sora is a Iroha based project support XOR token	- None - Currency can build via Chaincode	- None - Currency can build via smart contract	- Ether (ETH) is the native cryptocurrency of the platform
Features	Supports EVM bytecode smart contracts and parallel transaction execution.	A robust permission system using accounts and roles mechanism.	Private channel technique, which are essentially separate ledgers.	The first Hyperledger blockchain project that can operate on a permissionless blockchain.	Provide programmability on blockchain. The most successful blockchain platforms after Bitcoin.

performance trend is to increase first and then stabilize at the bottleneck value. It can be seen that the performance of Fabric is significantly better than other blockchain platforms. Besu takes second place, and the remaining three are at the same level. For Fabric, the performance bottleneck is reached when the input rate is 70 tps. Whereas for Iroha, Sawtooth, and Ethereum, when the input rate reaches 20 tps, the performance already reached their bottleneck. In general, the performance of Iroha, Sawtooth, and Ethereum shows some fluctuations, but the performance of Sawtooth tends to be the worst. It can also be seen that Ethereum outperforms Iroha at high transaction input rates (greater than 50 tps).

Similarly, Figure 4b shows the changes in read throughput of different blockchain platforms. In order to show trends and bottlenecks more clearly, we increased the maximum input rate to 300 tps. It can be seen that compared with write performance, the read performance of major blockchain platforms has been greatly improved. Overall, Iroha performed the best, with a bottleneck of around 200 tps, followed by Ethereum, Besu, and Fabric. Not surprisingly, Sawtooth has the worst read performance, with a bottleneck of about 60 tps.



(a) Transaction input rate vs. Blockchain write throughput



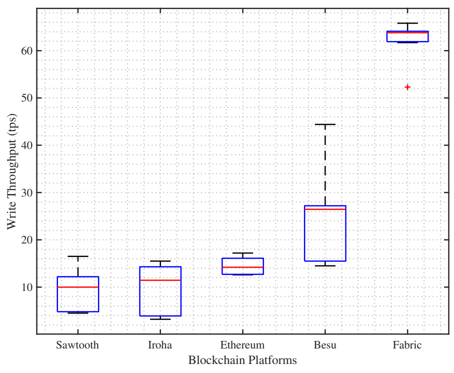
(b) Transaction input rate vs. Blockchain read throughput

Fig. 4: Impact of transaction input rate on blockchain performance

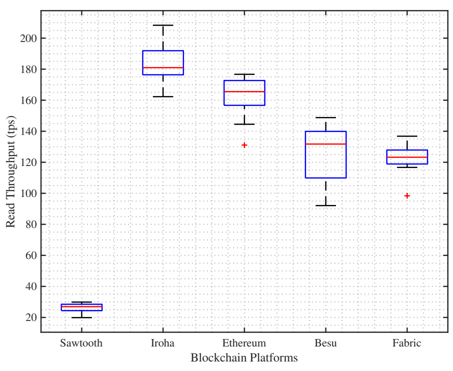
4.2.2 Stability Evaluation

In this experiment, we used a fixed transaction input rate with 100 tps and repeat it 10 times to observe the performance stability of the blockchain platform when facing the same transaction request multiple times. In fact, such an experimental setup can reveal the performance of the blockchain platform in the continuous processing of the same or similar transaction requests.

As can be seen from Figure 5a, in terms of write throughput, the performance of Fabric is the most stable one, and its throughput is much higher than other platforms. Although Ethereum's performance is relatively stable as well, the average throughput is not high, only about 14 tps. Iroha, Sawtooth, and Ethereum are at the same level and their stability is relatively poor. In terms of read throughput, although Sawtooth has the most stable performance and small fluctuations, its throughput is far less than the other four. In contrast, Fabric and Iroha have better stability whereas Besu has the worst one, which can be seen from Figure 5b .



(a) Variation in write throughput



(b) Variation in read throughput

Fig. 5: Variation in throughput of different blockchain platforms

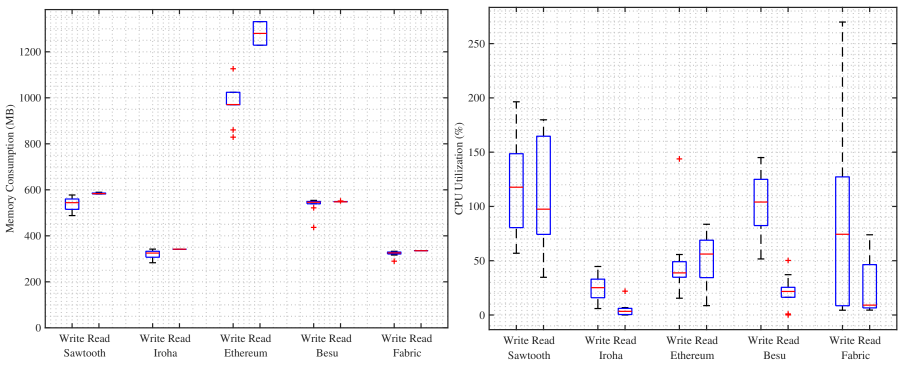
4.2.3 Resource Consumption Evaluation

In this experiment, we use a linear transaction input rate to observe the CPU and memory resource consumption of the blockchain platform when facing transaction requests under dynamic transactions patterns.

As can be seen from Figure 6a, in general, the memory consumption of several blockchain platforms is relatively stable. More specifically, Ethereum is more sensitive to memory consumption in our transaction mode. Its average memory consumption is about 970 MB for write and 1280 MB for read. Sawtooth and Besu have similar memory consumption for write/read, both between 500-600 MB. Iroha and Fabric have similar memory consumption, both of which are around 330 MB. In addition, the memory consumption of read operation is more stable than write operation on most platforms. Ethereum is a special case. Although its overall memory consumption for write is more stable than read , there are three outliers in the write part.

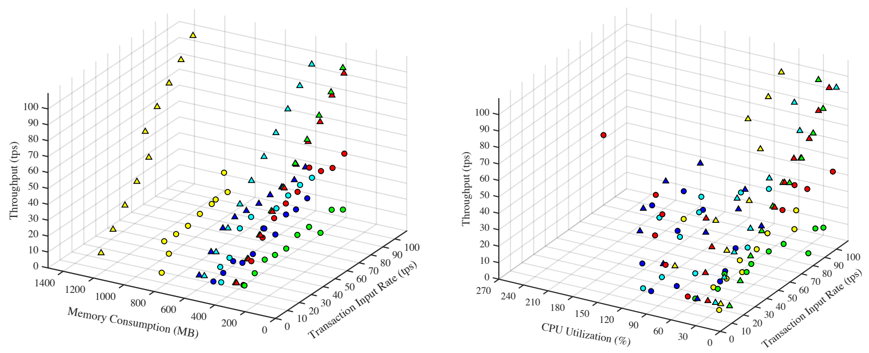
Compared with the stable memory consumption, the CPU utilization of the different platform changes significantly and tends to be unstable, as shown in Figure 6b. Most platforms have higher CPU utilization when doing write operations, except for Ethereum. Among them, Sawtooth has the most CPU usage, and its average CPU utilization for write and read reached 119% and 107%, respectively. This proves that the current transaction input volume has reached the performance bottleneck. Iroha has the lowest CPU utilization, with write/read operations for only 25% and 5%.

There are also some findings from the scatter plot shown in Figure 7a and Figure 7b. With the linear increase of transaction input rate from 0 to 100 tps, the improvement of the read performance of the platform is more obvious than the write performance. At the same time, the increase in transaction input rate is usually accompanied by an increase in memory consumption. Finally, when the input rate is increased, the changing trend of CPU utilization shows dynamic fluctuations and worse stability.



(a) Variation in memory consumption (b) Variation in CPU utilization

Fig. 6: Variation in resource consumption of different blockchain platforms



(a) Memory consumption vs. Throughput (b) CPU utilization vs. Throughput

Fig. 7: Scatter plot of blockchain resource consumption impact on performance. Blue, green, yellow, cyan, and red dots represent Sawtooth, Iroha, Ethereum, Besu, and Fabric, respectively. The triangle represents read operations, and the circles represent write operations.

4.3 Discussion

Based on the observations above, we draw the conclusions as follows:

1. In terms of performance scalability, with the increase of input transaction volume, the write/read performance of different blockchain platforms will first increase and then reach its bottleneck. Specifically, Fabric has the highest write performance of 66 tps, and Iroha has the highest read performance bottleneck at around 203 tps. By contrast, Sawtooth has the worst overall performance, with write and read bottlenecks of about 13 tps and 60 tps, respectively.

2. In terms of platform performance stability, Fabric has the most stable write transaction performance. Although Sawtooth has the most stable read performance, its average throughput is very low. In contrast, Iroha has both stable and the highest average read performance.
3. In terms of platform resource consumption, Ethereum needs more memory consumption under our transaction patterns. Its average memory consumption is about 970 MB for write and 1280 MB for read. Sawtooth and Besu have comparable memory consumption for write/read between 500-600 MB. The same situation comes to Iroha and Fabric, both of which are around 330 MB. In addition, when the transaction rate increases, it is usually accompanied by an increase in memory consumption. However, the CPU utilization is in a state of random dynamic fluctuations.

The above experimental results give us a lot of insights when building an ALLSTAR decentralized Cloud/Edge framework. By analyzing the demand of ALLSTAR on the underlying blockchain infrastructure, we believe that write throughput is the most important metric for evaluating ALLSTAR in response to new service transactions submitted by large-scale Cloud/Edge users. At the same time, the resource consumption of the ALLSTAR blockchain should be maintained at a low level to cope with the limited computing power of lightweight Edge devices.

In our experiments, Fabric has the optimal write throughput in terms of scalability and stability, while the read throughput remains at an acceptable level. It also has the lowest and most stable memory consumption. Therefore, we conclude that Fabric is the best choice to build ALLSTAR at this moment. This is in line with our expectation that Fabric is currently the most successful permissioned blockchain available in the market. Our next plan is to choose Fabric as the main infrastructure to build the ALLSTAR framework.

At the same time, we suggest that Sawtooth and Ethereum (private) are not good choices for ALLSTAR because they either show too low throughput or consume too many CPU/memory resources. Nevertheless, each platform has its own features and there exist trade-offs between different platforms. For example, while Iroha is less impressive in terms of write throughput, it has the highest read throughput and the most stable and lowest CPU consumption, which is a perfect choice for lightweight Edge devices. In practice, the inter-chain protocol of the SETIC subsystem will suggest users according to their specific needs to select and customize the right blockchain to build the ALLSTAR ecosystem.

5 Conclusion

This paper describes a typical scenario in the “Cloud+Edge” environment when involving Edge resources for orchestrating collaborative applications. We analyze five barriers in such a dynamic and complex environment, when considering various small Edge resource providers. To motivate resource providers to join the ecosystem and optimize the application DevOps lifecycle being faced with the complex environment, we propose the ALLSTAR approach, which targets at the vision that Edge infrastructure resources can equally contribute to the computing environment for usage, just as the Cloud resources. In this scenario, all the related roles, including Cloud/Edge resource providers, application developers/operators and end users, construct the ALLSTAR ecosystem. Different from the existing work, this paper considers the problem of the "Cloud+Edge" resource sharing and trading for the application orchestration, and put forward the solution in a systematic view, including components of HASKIT, CREAF, WAFIR, and SETIC. We present technique details of each component, and specifically detail three lifecycles to show how different roles interact with each other and support the business model. As the blockchain technique is adopted as the underlying trust layer, we further conduct an empirical study on evaluating different blockchain platforms, including their scalability, stability, and resource consumption. The results demonstrate that the permissioned blockchain is feasible to be leveraged in the inter-chain fabric as design. Furthermore, the systematic comparison provides the insights for choosing a proper blockchain when constructing the ecosystem.

Acknowledgements This research is partially funded by the EU Horizon 2020 research and innovation program under grant agreements No. 825134 (ARTICONF), No. 824068 (ENVRI-FAIR), and No. 862409 (BLUEClOUD). This work is also supported by the National Key Research and Development Program of China under grant 2018YFB0204301 and the Natural Science Foundation of Hunan Province under grant No. 2020JJ3042. We thank *MOG Technologies* for providing the crowd journalism usecase. The author, Zeshun Shi, was also sponsored by China Scholarship Council.

Conflict of interest

The authors declare that they have no conflict of interest.

References

Buterin V (2014) Ethereum: A next-generation smart contract and decentralized application platform

Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. Futur Gener Comp Syst 25(6):599–616

Castro M, Liskov B (1999) Practical byzantine fault tolerance. In: 1999 USENIX Symposium on Operating Systems Design and Implementation (OSDI), pp 173–186

Faniyi F, Bahsoon R (2015) A systematic review of service level management in the cloud. ACM Comput Surv 48(3):1–27

Guo Y, Wang S, Zhou A, Xu J, Yuan J, Hsu CH (2020) User allocation-aware edge cloud placement in mobile edge computing. Softw-Pract Exp 50(5):489–502

Higuchi T, Dressler F, Altintas O (2018) How to keep a vehicular micro cloud intact. In: 2018 IEEE Vehicular Technology Conference (VTC Spring), pp 1–5

Hu Y, Zhou H, de Laat C, Zhao Z (2018) Ecsched: Efficient container scheduling on heterogeneous clusters. In: 2018 European Conference on Parallel Processing (Euro-Par), pp 365–377

Hu Y, Zhou H, de Laat C, Zhao Z (2020) Concurrent container scheduling on heterogeneous clusters with multi-resource constraints. Futur Gener Comp Syst 102:562–573

Jeferry K, Kousiouris G, Kyriazis D, Altmann J, Ciuffoletti A, Maglogiannis I, Nesi P, Suzic B, Zhao Z (2015) Challenges emerging from future cloud application scenarios. Procedia Comput Sci 68:227–237

Koulouzis S, Martin P, Zhou H, Hu Y, Wang J, Carval T, Grenier B, Heikkinen J, de Laat C, Zhao Z (2020) Time-critical data management in clouds: Challenges and a dynamic real-time infrastructure planner (drip) solution. Concurr Comput-Pract Exp 32(16):e5269

Nakamoto S (2008) Bitcoin: A peer-to-peer electronic cash system

Nakashima H, Aoyama M (2017) An automation method of sla contract of web apis and its platform based on blockchain concept. In: 2017 IEEE International Conference on Cognitive Computing (ICCC), pp 32–39

Nofer M, Gomber P, Hinz O, Schiereck D (2017) Blockchain. Bus Inf Syst Eng 59(3):183–187

Qian H, Andresen D (2016) Automate scientific workflow execution between local cluster and cloud. Int J Networked Distrib Comput 4(1):45–54

Shi W, Cao J, Zhang Q, Li Y, Xu L (2016) Edge computing: Vision and challenges. IEEE Internet Things J 3(5):637–646

Shi Z, Zhou H, Surbiryala J, Hu Y, de Laat C, Zhao Z (2019) An automated customization and performance profiling framework for permissioned blockchains in a virtualized environment. In: 2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp 404–410

Teerapittayanon S, McDanel B, Kung HT (2017) Distributed deep neural networks over the cloud, the edge and end devices. In: 2017 IEEE International Conference on Distributed Computing Systems (ICDCS), pp 328–339

Uriarte RB, Zhou H, Kritikos K, Shi Z, Zhao Z, De Nicola R (2020) Distributed service-level agreement management with smart contracts and blockchain. Concurr Comput-Pract Exp p e5800

Venkateswaran S, Sarkar S (2018) Architectural partitioning and deployment modeling on hybrid clouds. Softw-Pract Exp 48(2):345–365

Wang N, Varghese B, Matthaiou M, Nikolopoulos DS (2020) Enorm: A framework for edge node resource management. IEEE Trans Serv Comput 13(6):1086–1099

Wang X, Yeo CS, Buyya R, Su J (2011) Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Futur Gener Comp Syst 27(8):1124–1134

Zheng Z, Xie S, Dai HN, Chen X, Wang H (2018) Blockchain challenges and opportunities: A survey. Int J Web Grid Serv 14(4):352–375

Zhou H, Hu Y, Su J, de Laat C, Zhao Z (2018) Cloudsstorm: An application-driven framework to enhance the programmability and controllability of cloud virtual infrastructures. In: 2018 IEEE International Conference on Cloud Computing (CLOUD), pp 265–280

Zhou H, Hu Y, Ouyang X, Su J, Koulouzis S, de Laat C, Zhao Z (2019a) Cloudsstorm: A framework for seamlessly programming and controlling virtual infrastructure functions during the devops lifecycle of cloud applications. Softw-Pract Exp 49(10):1421–1447

Zhou H, Ouyang X, Ren Z, Su J, de Laat C, Zhao Z (2019b) A blockchain based witness model for trustworthy cloud service level agreement enforcement. In: 2019 IEEE International Conference on Computer Communications (INFOCOM), pp 1567–1575

Ziafat H, Babamir SM (2018) Optimal selection of vms for resource task scheduling in geographically distributed clouds using fuzzy c-mean and molp. Softw-Pract Exp 48(10):1820–1846