# VOICE: Value-of-Information for Compute Continuum Ecosystems

Mattia Zaccarini‡, Benedetta Cantelli‡, Maria Fazio*, William Fornaciari†,
Filippo Poltronieri‡, Cesare Stefanelli‡, Mauro Tortonesi‡
* University of Messina, Messina, Italy
Email: maria.fazio@unime.it
† Politecnico di Milano, Milan, Italy
Email: william.fornaciari@polimi.it
‡ University of Ferrara, Ferrara, Italy
Email: benedetta.cantelli@edu.unife.it, {filippo.poltronieri, cesare.stefanelli, mauro.tortonesi, mattia.zaccarini}@unife.it

***Abstract*—The "Compute Continuum" (CC), which encompasses and harmonizes the Edge, Fog, and Cloud computing paradigms, emerges as an opportunity for an efficient deployment and adoption of innovative services ranging from eHealth to virtual reality. However, the effective management of services all along the CC presents the challenge of service deployment in relatively resource scarce or poorly connected locations. We argue that CC environments call for novel resource management strategies with a holistic perspective of the whole computing ecosystem, that also consider the QoS reconfiguration and tuning of services in the resource reallocation process. Tackling these challenges, we introduce VOICE, a cutting-edge platform designed for service management in CC ecosystems. Thanks to a Value of Information (VoI)-based technique that emphasizes the significance of data filtering, VOICE supports dynamic service components allocation throughout the CC. To evaluate the choice of VoI as enabling-technology for VOICE, we leverage computational intelligence (CI) approaches and compare it with more traditional strategies (e.g. maximization of satisfied requests or latency minimization). Final results show how VoI can be a valuable proposal to optimize the usage of resource scarcity that characterizes CC environments.**

***Index Terms*—Compute Continuum, Value of Information, Computational Intelligence, Service Management**

## I. Introduction

Over the past decade, Fog and Edge Computing have emerged as promising computing paradigm which can process the Internet-of-Things (IoT) data close to the sources, thus avoiding to transmit all data to distant cloud computing facilities [1]. The presence of multiple and heterogeneous computing layers which we can summarize in the edge, fog, and cloud layers. This aggregate can can provide computing capabilities at different performance and latency. This aggregate is generally refereed to as "Compute Continuum" (CC), or sometimes "Computing Continuum", [2] and presents compelling opportunities for the effective realization of next generation services, e.g., eHealth, gaming, Virtual and Augmented reality, the development of new-generation services that demands high computing capabilities.

The CC extends high-performance cloud services towards energy-efficient and low-latency devices close to the data sources. This brings significant opportunities in terms of service delivery, making them even more pervasive and efficient, but also new challenges at the service development, deployment, and management levels, that need to be specifically investigated and addressed [3] [4]. This is especially true in terms of how to effectively deploy and configure software components so that they deliver high Quality of Service (QoS) even in (relatively) resource scarce environments, as it is often the case at the edge of the network.

In fact, the proliferation of IoT sensors and devices is generating a deluge of data which is often processed in cloud computing facilities located far away from where the IoT data were generated. Despite new generation communication technologies, i.e., 5G and beyond, promise higher bandwidth and reliability, distributing processing along the CC represents a promising approach in terms of lower service latency and communication overhead [5]. This suggests the opportunity to experiment with information-centric approaches [6] and lossy and adaptive service models [7] that focus on the processing and dissemination of important data.

In this regards, Value of Information (VoI) represents a compelling approach to rank information and services according to the actual value they provide to the end users are particularly attractive [8], [9]. In fact, VoI-based approaches enable to maximize in a straightforward fashion resource utilization from the user perspective across the entire CC, also addressing the issue of resource sharing among competing services.

Orchestration tools so far have focused on providing a seamless deployment solution for containerized microservices. Despite remarkable progress in recent years, also in CC environments and applications, they have relatively neglected the opportunity to integrate with service components at the QoS tuning level [10]. Instead, we argue that CC environments call for new resource management strategies with a holistic perspective of the whole computing ecosystem, that consider opportunities at both the resource reallocation - either horizontally (across nodes on the same Cloud, Fog, or Edge layer) or vertically (across multiple layers) - and at the service QoS reconfiguration levels [11].

To enable resource allocation all along the CC while delivering the highest possible value to end users, we developed *Value-of-Information for Compute Continuum Ecosys-*

*tems (VOICE)*, an innovative platform that allows to run and manage services with dynamic topologies on the CC, with service components running on devices exhibiting highly heterogeneous capabilities, in terms of storage, connectivity and performance. VOICE aims to push forward the state of the art for service elasticity management in the CC by explicitly involving application components in resource allocation expansion or contraction operations, to ensure the best QoS levels according to conditions and application runtime requirements. VOICE builds on top of previous research on adaptive VoI-based service models and runtimes [9], CI-based optimization solutions [12], and innovative Digital Twin approaches [13], [14] to provide a full fledged orchestration solution designed for the CC.

We demonstrate the effectiveness of the VOICE approach by comparing its VoI-based service and resource management capabilities with a more traditional orchestration approach that does not consider and tries to process all the data. Experimental results demonstrates that the VOICE approach is much more efficient for the management of high priority services in a high-load scenario.

## II. Related Work

In literature, some works present solutions for orchestrating and managing resources at different computing levels, such as in the Cloud [15], Fog [16], and at the Edge [17]. However, these solutions cannot be directly applied to the CC since the high diversity in computation assets at different levels and resource management requirements.

The concurrent execution of an application on the entire CC and its dependency on the underlying infrastructure is a big challenge [18]. Even if in literature some works deal with these issues, they are mainly focused on specific uses case needs, such as Assisted Autonomous Vehicle systems [19] or Industry 4.0 customized production [20], where it is planned a priori the computing workflow to deploy and how Cloud, Fog and Edge resources can be exploited according to the computational complexity of the software components. However, such static and application-oriented strategies limit the feasibility of the proposed solutions, which can not be applied in domains with different workloads and scalability requirements. Most of all, in a holistic view of the computing ecosystem, specific actions should be undertaken to deal with concurrent executions of several applications characterized by a significant number of users with heterogeneous interests.

Existing orchestration platforms (such as Kubernetes, Docker Swarm, Amazon EC2 Container Service, and OpenShift Origin) are not able of handling conflicting QoS requirements while undertaking container-mapping decisions. Some optimization strategies in literature support scheduler configurations using different low-level QoS requirements (e.g., cost, throughput, latency, availability, etc.) [21] [22].

The need to optimize complex service fabrics in highly heterogeneous environments fostered the development of innovative concepts and tools, including Value-of-Information (VoI). Originally born from the seminal research by Howard in 1966 [23] as an extension to the Shannon's Information Theory for economics, and widely explored in decision sciences, the VoI concept has been recently investigated in different research areas related to the network and service management [9]. In [8] VoI was applied to optimize bandwidth consumption and information delivery in low-bandwidth and disrupted wireless networks. The work in [24] proposes a VoI-based mechanism for ranking IoT sensors data. A VoI-based computation scheduling for Cloud Computing applications, which considers both the costs and the benefits for processing information is proposed in [25]. Finally, the application of VoI was investigated in [26] to develop a prioritization tool for filtering the most important information messages to transmit in future vehicular networks. However, the integration of VoI management policies within resource management and orchestration is still a relatively unexplored research avenue.

## III. The case for VoI-based approaches in the CC

The CC is composed of three main layers: a Cloud layer, characterized by a Cloud data center that acts as a remote infrastructure shareable by different tenants with similar needs and provides abundant storage and computing resources. The second layer is the Fog layer, which enables the execution of microservice components with an interesting trade-off between latency and resource availability. It also can support temporary network disconnection of Edge nodes from the network, without propagating the issue to the Cloud data centers and reducing delays in recovering/restoring services at the Edge. Finally, the Edge layer provides specialized and/or on-site processing. It is the most limited in total resources but the most efficient in reducing latency and enabling faster computation and decision-making.

Therefore, task allocation/re-modulation processes, when a specific microservice requires more processing capabilities or responding to new QoS requirements, need to consider whether to expand workloads horizontally, i.e., by allocating more or different resources to the software component in the same computing level (e.g., in the Edge), or vertically, i.e., by migrating the software component to a resource-richer node or to upper layers (e.g., from the Fog to the Cloud).

At the same time, there is the need to address the gap between an ever increasing deluge of information and the (relative) resource scarcity at the edge, where this information are produced - and often consumed. In fact, the massive introduction of IoT devices and the increasing need for computation closer to the users, to avoid processing latencies caused by the transfer to and processing of data in the Cloud, are arguably the main reasons why CC nowadays has a significant and growing interest.

These challenges introduced by CC environments suggests the opportunity to push forward the state of the art and experiment with lossy and adaptive service models. In this context, Value-of-Information (VoI) methodologies and tools represent an innovative and interesting foundation for the realization of immersive and adaptive services. *VoI is a subjective measure that allows quantifying the value that an*

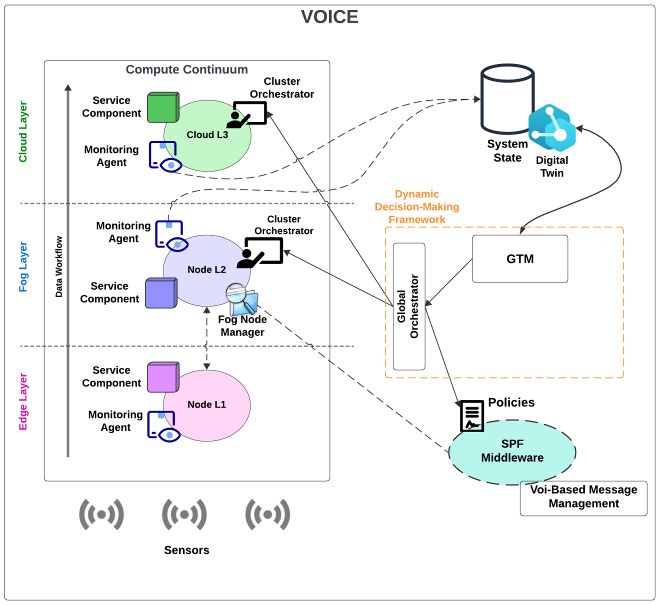

Fig. 1. Global Picture of the VOICE platform.

*information provides to its users*. VoI-based methods explicitly consider the different characteristics of Information Objects (IOs), tracking the value/utility of IOs during their processing and dissemination phases, thus enabling to rank these IOs according to their importance and relevance and to prioritize the processing and dissemination of the most important IOs, while discarding low value ones [7]. In turn, ranking services and microservice components according to the total amount of VoI they provide to end users represents a natural and effective approach to realize self-adaptive services for Fog computing applications. To this end, interested readers can refer to the work in [9], in which the mathematical framework for VoI calculation is presented.

By using the amount of VoI delivered to end users as a resource assignment criterion, a system will be able to naturally and seamlessly prioritize the assignment of resources to microservices that are providing the highest value to their end users - either because they are serving a considerable amount of users or because they are providing highly valuable information [9]. Factors such as the availability of resources or the changing demands of applications could be taken into account to estimate the relevance of requests and dynamically allocate resources in the CC environment. Thus, the modulation of VoI-based message processing policies allows decision-making components to naturally adapt to the current resource availability and the computing context. Therefore, in case of resource abundance and delivery of high VoI to end users, every message generated will ideally be processed by these components. However, in case of resource scarcity and/or an abundance of concurring services that try to compete for the same resources, decision-making components will prioritize the processing and delivery of messages with a higher VoI and discard those with a lower VoI.

## IV. VOICE

To realize this vision, there is the need for new platforms designed for CC and capable of defining, running, and managing VoI based IT services. VOICE aims to develop a distributed and adaptive software infrastructure to optimally support, monitor and orchestrate an ecosystem of concurrent IoT services for the CC.

### A. *Service Model and Runtime*

VOICE conceives the CC as an ensemble of heterogeneous computing clusters for enabling the development, dynamic deployment, and management of elastic IT services within and across multiple and heterogeneous clusters, each one containing different types of nodes (i.e., L1: edge, L2: fog, L3: cloud), as depicted in Fig. 1. Different types of nodes in the system give different opportunities for data processing and software execution. Specifically, VOICE services are realized through the composition of clearly separated but interconnected components (i.e.microservices) that can be independently deployed and executed. Each service has a description document, identifying the requirements that each component has and the connections between components. Our main purpose is to tackle the challenge of combining various execution platforms, from smart IoT devices, useful to collect data from the environment and trigger events, to geographically distributed edge nodes and from fixed or mobile fog nodes to the Cloud, into an ubiquitous and seamless execution environment. However, their interoperability represents a challenge due to the adoption of different technologies at different levels and the different perspectives in the management of resources.

Furthermore, VOICE is able to manage all the available resources in the CC implementing a dynamic and multi-layer orchestration solution with differentiated strategies for optimizing service deployments and scalability at different levels. This is the key approach for a seamless and pervasive computing system able to optimize performance of running applications and services.

VOICE adopts “Sieve, Process, and Forward” (SPF) as a reference service component execution runtime supporting VoI-based processing [7]. SPF provides a powerful set of concepts and tools for the development of VoI-based service fabrics. VOICE exploit SPF to build a multi-layer resource management framework that extends state-of-the-art VoI modeling to define specific VoI evaluation solutions tailored to the type of information to be processed.

### B. *Hierarchical Resource Management and Orchestration*

VOICE introduces a multi-layer orchestration solution to overcome specific resource management needs at different levels, with a dedicated resource management layer built on top of the Global Orchestrator, Cluster Orchestrator, and Fog Node Manager components.

The Global Orchestrator is in charge of configuring and coordinating the deployment of applications and services across computing clusters and nodes at different levels, elaborating information from the MAs and enabling the CC with specific mechanisms that allow software components to move vertically along the resource level stack. Each Fog node connected to the VOICE platform has a Fog Node Manager component, which is in charge of abstracting the properties of the specific physical/virtual node and dynamically allocating a given share of available resources to one or more microservice components according to the requests of the Global Orchestrator. The Node Manager will implement potential enhancements to the computing resource management at a local level working on the reservation of computing resources, the placement of the virtual elements contributing to a service, the performance efficiency, and the mutual isolation of virtualized workloads sharing the same hardware substrate.

To optimize service deployments and resource allocation along the CC, VOICE develops a dedicated dynamic decision-making framework. Global Topology Manager (GTM) is the software entity in charge of driving the allocation of the service components to the nodes at different levels such that their demand for computational resources is met. Such decisions are then passed to the Global Orchestrator, which is in charge of carrying out the actual deployment operations. To perform its duties, GTM has perfect knowledge of the computational capacity of each device as well as the demand of each service component and their mutual interactions, beside the service priority and expected quality-of-service (QoS). The aims are manifold: first, the demand of each service component should be fully satisfied, meaning that the number of service components allocated to the Cloud should be minimized while keeping latency sensitive service components at the edge. Second, service components with a high level of interaction should be allocated on the same node to minimize the overall latency. Third, VOICE would try to allocate service components to maximize the VoI delivered to their users. At the Edge level, the Cluster Orchestrator is a distributed software entity that automates the deployment of service topologies in a cluster of Edge nodes, allowing resource reallocations for horizontal scalability.

The VOICE GTM goes beyond the current state-of-the-art by introducing scheduling strategies that consider both heterogeneous resource availability in different nodes and the total VoI optimization criterion for resource assignment. Different example of them are provided in Section VI, where experiments show how VOICE is able to adapt the distribution of allocated resources depending on its final objective. Toward that goal, GTM adopts sophisticated CI solutions, such as Genetic Algorithms (GAs) and Quantum-inspired Particle Swarm Optimization (QPSO), that can quickly explore the very large and complex space of feasible IT service configuration to identify the optimal one - also leveraging a Digital Twin as detailed in the next subsection. Along with other CI-based solutions, both GA and QPSO have demonstrated remarkable flexibility and performance in managing resources in CC scenarios [12], especially due to their ease of implementation and ability to avoid stagnation in local minima thanks to their stochastic nature.

### C. Digital Twin-based Proactive Service Reconfiguration

VOICE features a distributed monitoring subsystem that not only allows to collect standard performance metrics such as response times, etc., but also implements the monitoring of VoI generation at both the single service component and at the entire service level. This information allows the VOICE resource management subsystem to make well-informed and cognitive service instantiation and adaptation decisions.

The distributed monitoring subsystem of the VOICE platform leverages a capillary infrastructure of dedicated Monitoring Agent (MA) components. Furthermore, differentiated versions of MAs along the CC provide monitoring capabilities for the global computing infrastructure and QoS of running applications. To do so, MAs will monitor both node resource availability and the current state of the service components running in their control domain.

As described in the previous subsection, CI solutions allow the VOICE orchestration components to execute the monitoring policies in a very reactive way, overcoming the problem typical of traditional monitoring strategies that can only provide the current information on node resources and services performance, limiting the vision of the logic behind the dynamic variation of system behavior.

However, we designed VOICE to realize proactive service reconfigurations by taking advantage of a Digital Twin approach, that allows to create a virtual replica of a real system, i.e., the CC infrastructure and services, for what-if scenario analysis and system optimization purposes [27] [28]. The VOICE GTM can leverage the Digital Twin of the system to evaluate the performance of a service in an alternative deployment and configuration, thus avoiding harm to the configuration of the real-world system. The Digital Twin will also provide an effective platform for experimentation with a wide range of VoI policies, with the objective of identifying the most promising ones.

The VOICE Digital Twin can be configured to use two different subsystems for reenacting a complex IT service in the CC. The default option is Phileas, a discrete event simulator that reenacts the execution of middleware solutions in CC environments [29]. Phileas features a relatively sophisticated service model that accurately evaluates the VoI produced by each single service component across the entire service fabric. As a result, it represents a very good fit for VOICE. Alternatively, we are currently developing KubeTwin, a state-of-the-art Digital Twin solution for the reenactment of Kubernetes applications [13]. KubeTwin has already proved capable of reenacting the behavior of Kubernetes clusters and applications in a remarkably accurate fashion [14]. Since Kubernetes is being increasingly adopted as the de facto standard orchestration platform across the CC, as KubeTwin matures we expect it to become the Digital Twin platform of choice for VOICE.

## V. Use Case

As a reference scenario for the adoption of the VOICE platform, we propose a smart city scenario characterized by several service components that must be allocated on the CC in a way that maximizes the total VoI delivered to end-users. We suppose that the smart city provides a multitude of cameras, which collect video-feeds from the surroundings to feed a plethora of smart city services such as traffic-monitoring, object-tracking, and object-detection applications.

We defined a total of six different services: pollution, traffic, audio, video, healthcare, and safety. The pollution service estimates the air quality in several locations of the smart city and notifies citizens via mobile devices if they have a high exposure to polluted air. The traffic service collects information about the traffic flows in the urban environment to let citizens know what routes to take, e.g., to avoid congestion and heavily polluted areas. On the other hand, the audio service provides analysis based on acoustic data, e.g. noise pollution caused by traffic, construction sites, industries, and so on. The video service instead analyzes camera feeds from nearby CCTVs and other video content, e.g. social media, to run machine learning processing. Healthcare is a compelling topic often associated with smart city environments, mobile telecommunications, and IoT. Therefore we describe a generic healthcare service for this use case. Finally, the safety service is to provide a variety of services including social safety, anti-terrorism, and crime prevention efforts. As for the video service, the latter two are defined to be time-sensitive and more critical than traffic, pollution, and audio services. In fact, there are many cases where an immediate response or action is required, such as locating a potential threat or finding a missing person.

Several instances of these services would be deployed on the top of 13 computing servers distributed in the cloud, fog, and edge levels. Specifically, for these experiments, we assume that 10 servers would be available at the edge level, and 2 at the fog level, and we model a single cloud entity with unlimited resources. This can be a representative example of a CC scenario where the majority of service components would exploit the computing capabilities in the close proximity of users and sensors. We also imagined that there would be several citizens interacting with these services. Therefore, we specify 10 different user groups that generate services' requests. Both devices and users are located in 13 different locations, all characterized by latitude and longitude coordinates according to the Global Positioning System (GPS) and with starting VoI values. In addition, we describe for each service type two different decay functions, a linear decay used for pollution, traffic, and audio services and an exponential one which is used for time-sensitive services. Data sources associated with time-sensitive services would generate requests with a higher starting VoI value, as we assume these services as more significant and urgent from users perspective. For the following experiments we model the VoI values in a $[0, 1]$ interval, where 0 corresponds to the lowest VoI possible and 1 to the highest. Let us also note that the VoI associated with time-sensitive services would have a stronger time decay when compared to standalone / batch services. It is therefore important to process time-sensitive requests as quickly as possible to minimize the decay information objects are subject to from their origination to their processing and delivery.

With regard to the communication modeling, to approximate the communication latency between the different levels of the CC, we rely on the measurements available at the CloudPing website (`https://www.cloudping.co/`), from which we select three pairs of locations whose communication latency could be a reasonable approximation of a computing scenario composed of connected edge, fog, and Cloud resources. For each pair of locations, we compute the parameters for modeling a random variable with a truncated Gaussian distribution. The same type of variable is adopted to model the starting VoI value for each type of service in the scenario. During the simulation, each time a new request of a specific type is generated, the starting VoI is computed according to its specific mean and standard deviation configuration, as presented in Table I. Specifically, we decided to define two main service classes: low VoI services, which include Audio, Pollution, and Traffic, and high VoI services, which include Healthcare, Safety, and Video. Each of these classes has its own characteristics, such as time decay type (Linear for low VoI services and Exponential for high VoI service), number of respective data sources, and amount of requests per second depending on the load scenario. All these metrics are visible in Table I. Furthermore, the longer each request waits in the relative queue, the more its VoI decays and the less likely the available resources will be allocated for its processing. Ideally, if a request remains stuck in the waiting queue for too long, its VoI decreases remarkably, and the users may no longer be strongly interested in receiving the computation results. Finally, considering the computing modeling, we assume that each service instance processes the incoming requests sequentially in a queue fashion. More specifically, we model the execution time of each service component with the adoption of a random variable with exponential distribution characterized by a specified parameter.

## VI. Experimental Evaluation

We reenacted the use case described in the previous Section in the VOICE Digital Twin component, leveraging the Phileas simulator [29], to compare three different optimization criteria for the service orchestration in the CC: VoI maximization, latency minimization, and requests completion ratio maximization, i.e., the number of requests processed within the simulation time window. We tested these methodologies in two different scenarios: the first one is a moderate load scenario, characterized by nearly 10,000 requests. Instead, the second is a heavy load scenario for a total of more than 20,000 total requests, with a remarkable increase of low VoI requests compared to the first outline. In a second experimental phase, we took a further step in the exploitation of the VoI. By manipulating a VoI defined for each service, CI approaches demonstrated their capability to find the configuration that better exploits the resource scarcity to compute high-value requests, aiming to overcome the limitations observed in the heavy load scenario during the first evaluation.

TABLE I
SERVICE METRICS USED IN EXPERIMENTAL EVALUATION

| Service Type | Time Decay | Data Sources | Avg. Message Generation Moderate Load (Msg/Sec) | Avg. Message Generation Heavy Load (Msg/Sec) | Starting VoI | |
|---|---|---|---|---|---|---|
| | | | | | **Mean** | **Standard Deviation** |
| Audio | Linear | 1 | 29 | 100 | 0.4 | 0.1 |
| Healthcare | Exponential | 2 | 9 | 9 | 0.8 | 0.05 |
| Pollution | Linear | 1 | 40 | 125 | 0.4 | 0.1 |
| Safety | Exponential | 2 | 8 | 8 | 0.8 | 0.05 |
| Traffic | Linear | 1 | 40 | 100 | 0.4 | 0.1 |
| Video | Exponential | 3 | 10 | 10 | 0.8 | 0.05 |

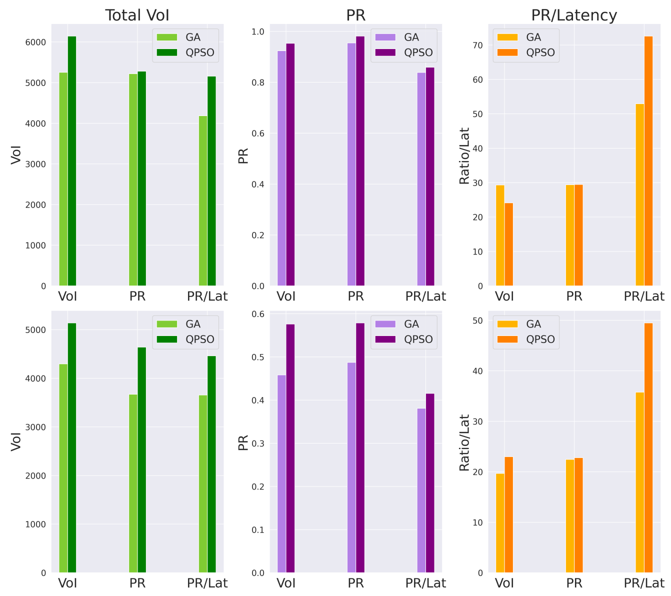

Fig. 2. Results of the three different optimization methodologies using GA and QPSO as optimization algorithms.

## A. VoI Approach Validation

To compare the various criteria we implemented a Genetic Algorithm (GA) and a Quantum-inspired Particle Swarm Optimization (QPSO) based optimizers and we defined three fitness functions corresponding to the optimization criteria. For all experiments, both GA and QPSO optimizers ran for a total of 250 generations. Each generation corresponds to 128 and 40 evaluations, i.e. a simulation run, of the fitness function respectively. Each simulation has a one-minute duration (including 10 seconds of warm-up), corresponding to the generation of roughly 10,000 service requests in the moderate load scenario and more than 20,000 in the heavy load one. This is possible by manipulating the exponential random variables that model the time between the generation of subsequent service requests. The orchestrator determines where to allocate instances for the defined service components among the CC devices. Let us note that the service orchestrator can activate multiple instances of the same service component on different devices to distribute the application load and improve the performance when necessary. Let us also specify that a computing device can run multiple instances of different service components, according to its computing capabilities. For these initial experimentation runs, we excluded the VoI thresholds to make a comparison between strategies as fair as possible for both scenarios.

For each optimization criterion, we select the best service component configuration for both load scenarios, in terms of the fitness function. We report the results in Fig. 2, which shows the total VoI, the requests completion ratio, and the ratio between this ratio and the averaged latency (measured as the summation of the requests' transfer time) for all three configurations of both GA and QPSO approaches. As expected, in both scenarios the VoI approach reports the highest VoI value. At the same time, it achieves a very competitive request completion rate. Regarding its PR/Latency outcome, it remains consistent with the results obtained by the "Max Ratio" approach. This is expected, as both methodologies aim to process as many requests as possible without considering other optimization aspects. This can easily lead to a higher average latency as a consequence, reducing the PR/latency value. Moreover, to evaluate the validity of our VoI approach, it is important to examine its component allocation efficiency at every CC layer compared to other strategies. Table II illustrates how the different methodologies place the multiple service components instances. The provided results prove once again the effectiveness of our VoI-based approach. Indeed, it is capable of obtaining very similar performance compared to the "Max Ratio" strategy for every load, but with resource savings up to 10% in the GA case and 30% for the QPSO. In the last configuration, it is evident that it achieves the lowest value of total requests processed and generated VoI. As mentioned before, this is an understandable behavior since the system cannot rely on the allocation of numerous service components (especially on the Fog and the Cloud layer) to minimize the latency. In fact, the more services components are allocated more the cumulative latency increases, causing a deterioration of the mean latency and, consequentially, of the final PR/Latency rate. This is even more evident in both Tables, which show that the Ratio/Latency configuration is the one that allocates the fewest service components in every occurrence.

In general, passing from a moderate to a heavy load scenario

TABLE II
SERVICE COMPONENTS ALLOCATED

| Moderate Load | | | | | |
|---|---|---|---|---|---|
| **Configuration** | **Metaheuristic** | **Edge** | **Fog** | **Cloud** | **Total** |
| Max VoI | GA | 37 | 6 | 4 | 47 |
| Max Ratio | GA | 41 | 8 | 5 | 54 |
| Ratio/Latency | GA | 34 | 3 | 0 | 37 |
| Max VoI | QPSO | 40 | 2 | 6 | 48 |
| Max Ratio | QPSO | 55 | 10 | 6 | 71 |
| Ratio/Latency | QPSO | 43 | 0 | 0 | 43 |

| Heavy Load | | | | | |
|---|---|---|---|---|---|
| **Configuration** | **Metaheuristic** | **Edge** | **Fog** | **Cloud** | **Total** |
| Max VoI | GA | 37 | 3 | 5 | 45 |
| Max Ratio | GA | 38 | 11 | 2 | 51 |
| Ratio/Latency | GA | 37 | 3 | 1 | 41 |
| Max VoI | QPSO | 53 | 6 | 6 | 65 |
| Max Ratio | QPSO | 59 | 12 | 6 | 77 |
| Ratio/Latency | QPSO | 39 | 0 | 0 | 39 |

results in a significant performance decline. Indeed, the huge amount of low VoI requests generated deteriorates the efficient usage of the available resources. To address this, our proposed approach enables the exploitation of VoI thresholds mechanism which act as a filter to cut requests with low VoI values and prioritizes the ones carrying higher values to the end-users. As a result, our proposed approach will guide the orchestrator towards service components configurations capable of delivering higher amount of VoI.

## *B. VoI Threshold Exploitation*

In this second experimental phase, we introduced a new parameter that both optimization approaches must handle for each type of service component: a VoI-based threshold to filter requests still considered as valuable from those for which it is not worth allocating resources. By manipulating these thresholds, we leveraged both GA and QPSO to demonstrate their ability to identify the configuration that best exploits the scarcity of available resources. This prioritizes the processing of high-value requests for the end user. To employ the filtering capabilities and overcome the shortcomings exposed in the previous evaluation, we applied the service thresholds on the scenario defined as heavy load. Specifically, we decided to run both orchestrators for a total of 500 generations each with the same number of evaluations of the previous evaluation. This enables them to better explore and find a more effective solution in this very dynamic proposed scenario. *During the process, the orchestrators could choose between 4 possible threshold values: 0.0, 0.125, 0.250, and 0.5.* Once set, these values remained constant until the next simulation. Fig. 3 and 4 illustrate the evolution of the total VoI obtained in relation to the varying thresholds throughout the optimization process conducted respectively by GA and QPSO. Specifically, the GA-based optimizer reaches a very similar VoI value than the previous one in the heavy load scenario (visible in the bottom-left plot in Fig. 2). However, in this case, it achieves this while processing over 3,000 fewer requests than before. This indicates that the thresholds effectively filtered out most low VoI requests, allowing system resources to be reserved for the processing of requests with high VoI value. In contrast,

TABLE III
SERVICE COMPONENTS ALLOCATED WITH THRESHOLD FILTERING

| Configuration | Metaheuristic | Edge | Fog | Cloud | Total |
|---|---|---|---|---|---|
| VoI Thresholds | GA | 40 | 5 | 5 | 50 |
| VoI Thresholds | QPSO | 44 | 6 | 6 | 56 |

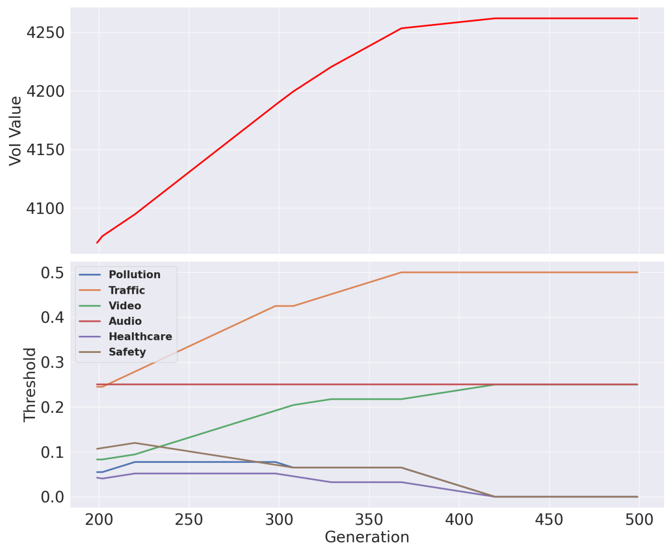

Fig. 3. Total VoI evolution in relation to the service thresholds manipulation by GA.

QPSO performed significantly better. Indeed, thanks to its thresholds manipulation, it identified a solution that achieves a better VoI with several thousand fewer processed requests along with a notable resource-saving (56 total components allocated as visible in Table III compared to 65 in Table II). In both cases, due to the remarkable increase of low VoI service requests in the heavy load scenario, their relative thresholds are predominant. Specifically, the QPSO optimizer achieves the best VoI values when it sets all low VoI service thresholds above the others (0.25 regarding the Traffic service and 0.125 for both Audio and Pollution). On the other hand, while GA can improve its efficiency, it is not as effective as QPSO. In fact, GA tends to lift up the threshold for the video service component instead of the Pollution one. Due to the fact that Pollution is not considered as critical as Video, this leads to a service component configuration that delivered a lower amount of VoI than the one generated with QPSO.

## VII. CONCLUSIONS

The preliminary experimental evaluation that we presented validates the concepts and design of the VOICE platform, showing that VoI-based service development and management has the potential to enable the creation of services that can seamlessly adapt to different contexts through the CC - with high dynamics, using the available resources efficiently, and providing very high QoS to the end users.

Future works will investigate how the core concepts and mechanisms at the heart of VOICE can be applied in federated CC domains. In this context, we will consider both centralized

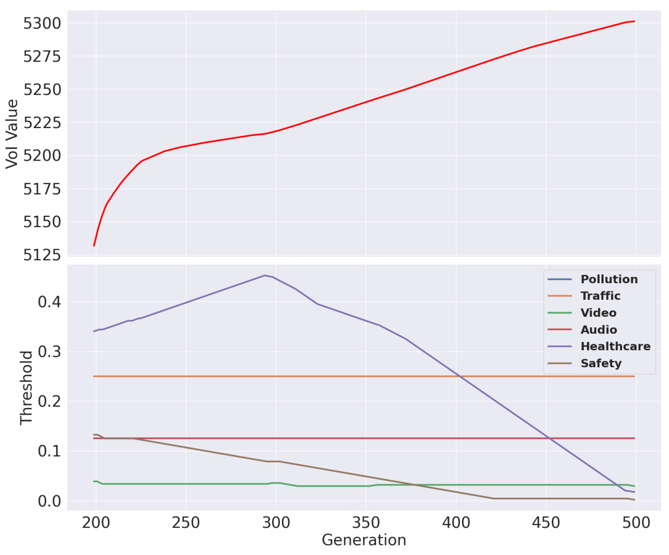

Fig. 4. Total VoI evolution in relation to the service thresholds manipulation by QPSO.

and distributed decision making solutions and comparatively evaluate their performance.

In addition, we are planning to investigate Reinforcement Learning (RL) as a potentially very promising alternative to computational intelligence solutions, i.e., GA and PSO, for proactive resource management within VOICE. In particular, offline RL solutions trained on a Digital Twin by leveraging the accurate modeling capabilities of KubeTwin, seem to hold the promise of addressing the main issue of RL - sample inefficiency - and thus represent very interesting research avenue.

## Acknowledgments

This work has been partially supported by the Spoke 1 “FutureHPC & BigData” of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Missione 4 - Next Generation EU (NGEU).

## References

[1] V. Sindhu et al. Fog based Edge Learning using IoT: Concepts, Challenges and Applications. In *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 18–21, 2022.

[2] S. Moreschini et al. Cloud Continuum: The Definition. *IEEE Access*, 10:131876 – 131886, 2022.

[3] M. Zanella et al. BarMan: A run-time management framework in the resource continuum. *Sustainable Computing: Informatics and Systems*, 35:100663, 2022.

[4] M. Zanella. *Post-cloud Computing: Addressing Resource Management in the Resource Continuum*, pp. 105–115. Springer International Publishing, Cham, 2023.

[5] M. Giordani et al. Toward 6G Networks: Use Cases and Technologies. *IEEE Communications Magazine*, 58(3):55–61, 2020.

[6] J. J. L. Escobar et al. Decentralized Serverless IoT Dataflow Architecture for the Cloud-to-Edge Continuum. In *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pp. 42–49, 2023.

[7] M. Tortonesi et al. Taming the IoT data deluge: An innovative information-centric service model for fog computing applications. *Future Generation Computer Systems*, 93:888–902, 2019.

[8] N. Suri et al. Exploring value-of-information-based approaches to support effective communications in tactical networks. *IEEE Communications Magazine*, 53(10):39–45, October 2015.

[9] F. Poltronieri et al. A Value-of-Information-based management framework for fog services. *International Journal of Network Management*, 32(1):e2156, 2022.

[10] K. Fu et al. Adaptive Resource Efficient Microservice Deployment in Cloud-Edge Continuum. *IEEE Transactions on Parallel and Distributed Systems*, 33(8):1825–1840, 2022.

[11] M. Villari et al. Osmotic Computing: A New Paradigm for Edge/Cloud Integration. *IEEE Cloud Computing*, 3(6):76–83, 2016.

[12] F. Poltronieri et al. Reinforcement Learning vs. Computational Intelligence: Comparing Service Management Approaches for the Cloud Continuum. *Future Internet*, 15(11), 2023.

[13] D. Borsatti et al. Modeling Digital Twins of Kubernetes-Based Applications. In *IEEE Symposium on Computers and Communications, ISCC 2023, Gammarth, Tunisia, July 9-12, 2023*, pp. 219–224. IEEE, 2023.

[14] L. Manca et al. Characterization of Microservice Response Time in Kubernetes: A Mixture Density Network Approach. In *2023 19th International Conference on Network and Service Management (CNSM)*, 2023.

[15] O. Tomarchio et al. Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks. *Journal of Cloud Computing*, 9:49, 09 2020.

[16] A. Bocci et al. Secure FaaS orchestration in the fog: how far are we? *Computing*, 103, 05 2021.

[17] P. D. Diamantoulakis et al. Optimal Design and Orchestration of Mobile Edge Computing With Energy Awareness. *IEEE Transactions on Sustainable Computing*, 7(2):456–470, 2022.

[18] C. R. de Mendoza et al. Near Optimal VNF Placement in Edge-Enabled 6G Networks. In *2022 25th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pp. 136–140, 2022.

[19] J. Arulraj et al. eCloud: A Vision for the Evolution of the Edge-Cloud Continuum. *Computer*, 54(5):24–33, 2021.

[20] C. Jiang and J. Wan. A Thing-Edge-Cloud Collaborative Computing Decision-Making Method for Personalized Customization Production. *IEEE Access*, 9:10962–10973, 2021.

[21] J. Santos et al. Resource Provisioning in Fog Computing: From Theory to Practice. *Sensors*, 19(10), 2019.

[22] L. J. M. León et al. EFCC: a flexible Emulation Framework to evaluate network, computing and application deployments in the Cloud Continuum. In *2023 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, 2023.

[23] R. A. Howard. Information Value Theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966.

[24] S. Bharti et al. Value of Information Based Sensor Ranking for Efficient Sensor Service Allocation in Service Oriented Wireless Sensor Networks. *IEEE Transactions on Emerging Topics in Computing*, 9(2):823–838, 2021.

[25] L. Bölöni and D. Turgut. Value of information based scheduling of cloud computing resources. *Future Generation Computer Systems*, 71:212–220, 2017.

[26] M. Giordani et al. Investigating Value of Information in Future Vehicular Communications. In *2019 IEEE 2nd Connected and Automated Vehicles Symposium (CAVS)*, pp. 1–5, 2019.

[27] S. Laso et al. Deploying Digital Twins Over the Cloud-to-Thing Continuum. In *2023 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, 2023.

[28] R. Minerva et al. Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models. *Proceedings of the IEEE*, 108(10):1785–1824, 2020.

[29] F. Poltronieri et al. Phileas: A Simulation-based Approach for the Evaluation of Value-based Fog Services. In *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pp. 1–6, Sep. 2018.