

# Value of Information based Optimal Service Fabric Management for Fog Computing

Filippo Poltronieri\*, Mauro Tortonesi\*, Alessandro Morelli<sup>†</sup>, Cesare Stefanelli\*, Niranjan Suri<sup>‡</sup>

\* Distributed Systems Research Group, University of Ferrara, Ferrara, Italy

{filippo.poltronieri,cesare.stefanelli,mauro.tortonesi}@unife.it

<sup>†</sup> Florida Institute for Human and Machine Cognition (IHMC), Pensacola, FL, USA

{amorelli,nsuri}@ihmc.us

<sup>‡</sup> US Army Research Laboratory (ARL), Adelphi, MD, USA

niranjan.suri.civ@mail.mil

**Abstract**—Service fabric management in Fog Computing is a challenging task, which has to deal with a complex and resource scarce environment. We argue that approaches leveraging Value-of-Information (VoI) concepts and tools are particularly interesting to support the realization of that objective. This paper describes innovative methodologies and reference models for the service fabric management for Fog Computing applications. First, we formalize the VoI concept and discuss its adoption in Fog Computing environments. Then, we propose a formal model that aims at maximizing the allocation of Fog services from a value-based perspective. To overcome the complexity of this model, we present two possible approaches (simulation-based optimization and a model approximation) and we compare them by adopting Evolutionary Algorithms (EAs) as optimization techniques. Experimental results prove the validity of both models in finding resource allocation solutions capable of minimizing network latency and maximizing the utility for the end-users of Fog Computing services. Finally, we show how the results of the approximated model can be adopted as a first approximated approach for resource management of Fog Computing services.

**Index Terms**—Fog Computing, Service Management, Optimization Models.

## I. INTRODUCTION

Fog Computing is a relatively recent paradigm that advocates the allocation of information-processing services at the edge of the network. Fog Computing is particularly well suited for Internet-of-Things (IoT) applications, as it leverages proximity with either raw data sources, information consumers, or both, in order to reduce latency and burden with network infrastructure with respect to Cloud Computing approaches [1], [2], [3].

Fog service fabric management, i.e., the dynamic assignment and management of resources in Fog Computing scenarios to fulfill service level objectives, represents a particularly challenging task. In fact, Fog Computing solutions are responsible to deliver low-latency applications at the edge of the network with improvements due to the proximity of users and devices. However, managing such applications means dealing with a significant number of users with heterogeneous interests, a large number of devices, and applications with different Quality-of-Service (QoS) and Quality-of-Experience (QoE) requirements.

Scientific literature often addressed Fog Computing resource management as an optimization problem, considering many different optimization criteria, including: deadline matching [4], energy consumption (green computing) [5], job duration [6], or tail latency minimization [7]. In this context, *Value of Information (VoI)* maximization, i.e., the maximization of the utility that Information Objects (IOs) deliver to end recipients, represents a particularly interesting criterion. In fact, VoI-based optimization would allow to naturally and seamlessly prioritize the assignment of resources to services 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 IOs.

Research on VoI has been developing a particularly promising set of methodologies and tools that could represent a foundation for a next generation service fabric management solutions. Recently, a number of VoI based solutions were proposed in several application areas [8], [9], [10], [2]. However, while interesting results have been achieved so far, they mostly followed *objective approaches* that calculate VoI of a message according to its content, and the distance between time and location of information generation and receipt. Very few works considered *subjective approaches* that take into account different utility models per each user when estimating VoI. Among those, an approximation of utility per message and user profile type is proposed in [11] and an interesting empirical approach measuring the user utility is followed in [12].

The challenging objective of this paper is to further develop existing methodologies and tools to exploit VoI concepts in next generation service management. To this end, the manuscript develops a theoretical framework for the optimization of service fabric in Fog Computing applications. The paper then investigates generally applicable models for the assessment of total VoI generated by a specific service fabric configuration, designed to be fitted in the above mentioned optimization framework. More specifically, we propose both a *user-specific VoI model* that assumes perfect information on the entire system and requires a simulation approach for its evaluation and a simpler *approximated model* which we specifically designed with practical adoption in mind. Finally, the

paper presents an experimental evaluation that compares the proposed models, adopting Evolutionary Algorithms (EAs) as optimization techniques. This experimental evaluation proves the validity of both models in finding services allocations capable of maximizing the end-users' utility of Fog services.

The considerations in this manuscript emerged from our experiences in the SPF [13] and Phileas [14] research projects. SPF is a Fog-as-a-Service middleware solution for the management and orchestration of Fog Computing applications, while Phileas is a discrete event simulator we designed to reenact the reproducible evaluation of Fog Computing scenarios. Both are built on top of the Adaptive, Information-centric, and Value-based (AIV) information maturity model that we adopt in this paper to formalize the VoI based model for service fabric optimization in Fog Computing applications.

## II. RELATED WORK

Originally born from the seminal research by Howard in 1966, Value of Information (VoI) has long been a widely explored topic in decision sciences. Recently, VoI has started attracting the attention of several researchers in the wireless communications and network and service management research areas as a natural criterion for optimization of (scarce) resource allocation.

Initially, VoI has been applied to military research in [11] to optimize bandwidth consumption and information delivery in tactical networks. In such situations, VoI based information delivering techniques have proven their abilities in dealing with low-bandwidth and disrupted networks.

A different perspective is the one provided in [8]. In this article Turgut et al. discuss the value of information and the cost of privacy in the Internet of Things (IoT). A formulation to quantify the value of an information/service that considers the costs (privacy, infrastructure, and maintenance) is given and discussed. Focusing instead on Cloud Computing applications, Boloni et al. propose an interesting VoI-based computation scheduling in [9]. The proposed model considers both the costs and the benefits associated with the processing of information on Cloud Computing platforms and shows how their VoI scheduling approach can outperform other traditional methodologies.

In [10], Giordani et al. discuss the application of VoI to prioritize the most important transmissions in future vehicular networks. The authors quantify VoI utilizing information quality, space, and time interdependencies criteria and consider its adoption as proxy for broadcasting decision in vehicular networks. Other works suggest the adoption of VoI to quantify the relevance of information and sensors such as the authors in [2], which propose VoI as a technique to design a sensor service ranking mechanism. Finally, another relevant work considering the possibility of measuring user-utility in the VoI formulation is [12].

An interesting scenario where VoI can provide its utility is Fog Computing, a recent and compelling research topic which attracted the attention of several works from both the academia and the industry. In fact, Fog Computing can be particularly

beneficial for resource management and IoT applications [15], [16]. Some solutions focus instead on optimizing Fog resource provisioning by minimizing the system (infrastructure) costs without invalidating QoS requirements [17].

With regard to resource management, Skarlat et al. propose an interesting formulation in [18]. In this work, the authors formalize the Fog service placement problem and propose a genetic algorithm as a problem heuristic solution. This demonstrates to achieve reduced network latency and better utilization of Fog resources. In [19], the authors discuss the Fog service placement problem in a smart city scenario and present a genetic algorithm for its resolution. Then, other works address resource management to achieve reduced energy consumption such as [20].

## III. SERVICE AND INFORMATION MATURITY MODEL

In this work we will adopt the *Adaptive, Information-centric, and Value-based (AIV)* service and information maturity model [14]. AIV proposes an innovative Fog service model, that enables the development of services capable of automatically scaling their resource requirements to match users' QoE requirements to the best of the system's capabilities given the current execution context.

More specifically, AIV assumes that the processing function of a Fog service emerges as the result of the coordinated orchestration of adaptive and composition-friendly service components. This loose definition of Fog services allows to rather easily support dynamic architectures in which the single instances of service components can be migrated to different devices along the Cloud-IoT continuum according to the current execution context (service requirements, resource availability, user preferences, etc.).

Developers will define and implement Fog services as a topology of service components that are connected together in a dynamic service fabric, according to a service description that defines the service semantics and characteristics and the interactions between service components. In turn, service components represent the basic building block of Fog services.

AIV also proposes an information maturity model which divides messages in three different categories, according to the corresponding information processing phases. The first processing phase regards *raw data*: input feeds of (typically sensing) data gathered from Wireless Sensor Networks (WSN), smartphones, wearable devices, and other IoT devices in general. According to the AIV model, raw data are analyzed by generic lower level service components to produce higher-order data constructs called *Information Objects (IOs)*. In particular, IOs are generated by multiple aggregated and/or distilled raw data fused to obtain more valuable information. Finally, the last processing phase makes use of the IOs to generate *Consumption Ready Information Objects (CRIOs)*, which represent the information in its final stage, ready to be consumed by the users who requested it for. Let us note that the generation of an IO could require many raw data objects, and that, in a similar fashion, a CRIO can be the result of the aggregation of multiple IOs provided by many different

intermediate service components. Generally, the AIV model considers IOs as the output of service components and CRIOS as the output of Fog services.

#### IV. AIV MODEL AND NOTATION

Building upon the AIV model discussed in the previous Section, we introduce the following notation, which we will use as a basis to formulate the optimal service fabric allocation problem.  $\mathbb{DS}$  is the list of data sources, which represent IoT sensors or other types of device that generate raw-data to be processed. Each data source  $ds_i \in \mathbb{DS}$  has location  $ds_i^l$ .

With regards to processing devices,  $\mathbb{D}$  is the list of edge devices, on which it is possible to execute service components that implement a (portion of a) Fog service. Each device  $d_i \in \mathbb{D}$  has location  $d_i^l$  and total resources  $d_i^{tr}$ .

$\mathbb{S}$  is the set of Fog services to allocate.  $s_j \in \mathbb{S}$  represents the j-th service. In turn, service  $s_j$  is realized as a workflow  $s_j^{wf}$  built on top of a set of different service components  $sc_k$ , with a corresponding amount of requested resources at time  $t$   $sc_k^{rr}(t)$ . More specifically, a workflow  $s_j^{wf}$  represents an order list of service components interacting each other, e.g.  $s_j^{wf} = sc_1, sc_2, sc_3, \dots, sc_n$ .

A service  $s_j$  is then defined as the combination of its service components  $\mathbb{SC}_j$ .  $\mathbb{SC}$  is the set of service components that need to be allocated. In this work, we assume service isolation to simplify the system model notation, i.e., no service component shared between services:

$$\mathbb{SC} = \bigcup_j \mathbb{SC}_j \quad (1)$$

However, by assuming service isolation we are not considering service components shared between different services. To overcome this simplification, we will relax this assumption in future work.

$\mathbb{U}$  is the set of users. Each user  $u_l \in \mathbb{U}$  has location  $u_l^l$  and it is subscribed to a set of services  $u_l^s$ .

It is possible to model the allocation of service components  $sc_k$  on fog devices by using matrix notation.  $\mathbb{A}$  represents the current service allocation configuration: it is a matrix of size  $|\mathbb{SC}| \times |\mathbb{D}|$ , where  $|\mathbb{SC}|$  is the number of service components and  $|\mathbb{D}|$  is the number of fog devices. The matrix  $\mathbb{A}$  is composed of  $\alpha_{k,i}$  elements:  $\alpha_{k,i}$  is equal to 1 if the service component  $sc_k$  is allocated on fog device  $d_i$ . Therefore, according to this formulation, a service  $s_j$  is “fully instantiated” if all its service components  $sc_k \in \mathbb{SC}_k$  result allocated at time  $t$ . Or, equivalently, if:

$$\sum_{k \in \mathbb{SC}_k, i \in \mathbb{D}} \alpha_{k,i} = |\mathbb{SC}_k|. \quad (2)$$

(Note that service components might, and typically will, be allocated on different edge devices.)

The resources allocated at time  $t$  on device  $d_i$  are  $ra_i(t) = \sum_{j \in \mathbb{A}} \alpha_{j,i} \times sc_j^{rr}$ , thus leaving  $ar_i(t) = 1 - ra_i(t)$  available resources at device  $d_i$  and time  $t$ . The overall assigned and

TABLE I  
SUMMARY OF USED NOTATION

| Term                          | Meaning                                                                            |
|-------------------------------|------------------------------------------------------------------------------------|
| $\mathbb{DS}$                 | Set of data sources                                                                |
| $ds_i \in \mathbb{DS}$        | i-th data source                                                                   |
| $ds_i^l$                      | Location of i-th data source                                                       |
| $\mathbb{D}$                  | Set of edge devices                                                                |
| $d_i \in \mathbb{D}$          | i-th device                                                                        |
| $d_i^l$                       | Location of i-th device                                                            |
| $d_i^{tr}$                    | Total resources at i-th device                                                     |
| $\mathbb{S}$                  | Set of Fog services                                                                |
| $s_j \in \mathbb{S}$          | j-th service                                                                       |
| $s_j^{wf}$                    | Workflow of j-th service                                                           |
| $sc_k$                        | k-th information processing component                                              |
| $sc_k^{rr}(t)$                | Amount of resources requested by k-th information processing component at time $t$ |
| $\alpha_{k,i} \in \mathbb{A}$ | Service component $sc_k$ allocated on device $d_i$                                 |
| $ra_i(t)$                     | Amount of resource allocated on device $d_i$                                       |
| $ar_i(t)$                     | Amount of available resources on device $d_i$                                      |
| $\mathbb{U}$                  | Set of end users                                                                   |
| $u_l \in \mathbb{U}$          | l-th end user                                                                      |
| $u_l^l$                       | Location of l-th end user                                                          |
| $u_l^s$                       | Set of services the l-th end user subscribed to                                    |
| $r^t$                         | Generation time of request $r$                                                     |
| $r^l$                         | Location of request $r$                                                            |
| $m^{ot}$                      | Originating time of message $m$                                                    |
| $m^{ol}$                      | Originating location of message $m$                                                |

available resources at time  $t$  can then be expressed as  $ra(t) = \sum_{j,i \in \mathbb{A}} \alpha_{j,i} \times sc_j^{rr}$  and  $ar(t) = 1 - ra(t)$  respectively.

$$\forall i \quad \sum_{j \in \mathbb{A}} \alpha_{j,i} \times sc_j^{rr} \leq d_i^{tr}. \quad (3)$$

Equation (3) defines a constraint on the edge resources in use at time  $t$  in the overall Fog Computing scenario. In fact, the amount of allocated computational resources allocated by service components running on a device  $d_i \in \mathbb{D}$  must be less or equal than amount of total resources  $d_i^{tr}$ .

$\Gamma$  is the communication configuration. In this manuscript, we will focus solely on the VoI-optimal instantiation of service components. We will consider the joint optimization of service instantiation and network configuration in future works.

#### V. OPTIMAL RESOURCE ALLOCATION PROBLEM

We formulate the problem of resource allocation for service fabric in Fog Computing applications using a mathematical optimization framework. More specifically, we aim to maximize a function  $f_n$  that captures the behavior of the system according to the optimization criterion in the n-th window of time (which goes from  $t_n$  to  $t_{n+1}$ ):

$$\begin{aligned} & \arg \max_{\mathbb{A}, \Gamma} f_n(\mathbb{A}, \Gamma, \mathbb{D}, \mathbb{S}, \mathbb{U}) \\ & \text{s.t.} \quad \forall k \quad \sum_{m \in \mathbb{SC}_k, i \in \mathbb{D}} \alpha_{m,i} = |\mathbb{SC}_k| \end{aligned} \quad (4)$$

Let us note that function  $f_n$  changes every time the global system state  $\Theta = (\mathbb{D}, \mathbb{S}, \mathbb{U})$  changes, e.g., when new resources become available or new service components are instantiated. In that case we have a new function to optimize, i.e.,  $f_{n+1}$ .

Using a sequence of functions to model the system implies that we are capable of finding out when the function changes, i.e., when the system changes – which is a quite complex task per se (see for instance [21], [22], [23], and [24] chapter 2).

Since it is going to be difficult to detect (or predict) changes to the global system state, and consequently to find out when the optimization procedure should be re-triggered, it might be convenient to reformulate the optimization problem in equation (4), switching from a model based on a sequence of static (i.e., time-invariant) optimization functions  $f_n$  to one based on a single dynamic (i.e., time-varying) optimization function  $f$ :

$$\begin{aligned} & \arg \max_{\alpha, \gamma} f(\mathbb{A}, \Gamma, \mathbb{D}, \mathbb{S}, \mathbb{U}, t) \\ \text{s.t.} \quad & \forall k \sum_{m \in \mathbb{SC}_k, i \in \mathbb{D}} \alpha_{m,i} = |\mathbb{SC}_k| \end{aligned} \quad (5)$$

This turns the optimal resource allocation problem from a sequence of (relatively) static optimization problems to a single – but significantly more complicated – dynamic optimization problem.

#### A. VoI-based optimization

Considering the assessment of the total VoI delivered by the service fabric to end users as the main criterion for optimization, we have:

$$f_n = \text{TOTAL}_{VoI}(t_n, t_{n+1}) = \sum_{m \in \mathbb{M}(t_n, t_{n+1})} VoI_{\Theta}(m). \quad (6)$$

where  $\mathbb{M}(t_n, t_{n+1})$  are the messages received by end users within the  $(t_n, t_{n+1})$  time window.

Note that for the purposes of the VoI maximization criterion the VoI contribution of messages exchanged between sensors and service components are considered indirectly in the calculation. In fact, the starting VoI of a CRIO message is the result of the aggregation of the multiple raw-data and IOs who generated it. Also note that a message can be received by different end users.

Also note that  $VoI_{\Theta}$  depends on the current state  $\Theta$  of the system. The next Section presents some proposals to define the  $VoI_{\Theta}$  model, adopting different perspectives and assumptions on the knowledge of the system.

### VI. VOI MODELS

Most VoI models proposed so far in literature only consider the (simpler) case in which the VoI of a message does not depend from the specific user that receives it. These approaches implicitly assume an “objective VoI perspective”, according to which the value of an information object is an intrinsic characteristic and does not depend from the user consuming it. More sophisticated approaches based on a “subjective VoI perspective” require a deep knowledge and differentiation for each possible user, thus further increasing the complexity of the model. This Section attempts to present generically applicable models of both types.

#### A. User-agnostic VoI Model

We assume that at the moment of its generation each message (of either raw data, IO and CRIO type) has an originating VoI. The starting VoI of IOs and CRIOs depends from the message- and service component-specific result of the processing of previous messages. The starting VoI associated to each IO is calculated as a service component-specific function of the raw data messages consumed in the generation of the IO.

The starting VoI associated to each CRIO can be calculated as a function of the priority of the corresponding service and of the output of a service-specific VoI calculation function, that every service implementation is supposed to provide:

$$VoI_0(m, \mathbb{I}_m, s) = SSP(\mathbb{I}(m)) \times FSP(\mathbb{I}(m)) \quad (7)$$

where  $\mathbb{I}_m$  is the set of input IOs processed in the generation of CRIO  $m$ ,  $s$  is the Fog service that generated  $m$ , and  $SSP(\mathbb{I}(m))$  is a factor that allows to take into account service-specific considerations when assessing the value of the information extracted from the  $\mathbb{I}(m)$  IOs. Finally,  $FSP(s)$  is the priority of  $s$ , thus representing a weight factor that permits to assign higher Vols to the CRIOs produced by higher priority Fog services.

The starting VoI in equation (7) is the basis to calculate the VoI of message  $m$  at its recipients. More specifically, the total VoI delivered by a CRIO  $m$  needs to consider factors such as the number of its recipients, timeliness relevance decay (a factor taking into account the decrease in the value for time sensitive IOs by applying a penalty according to the delivery time), proximity relevance decay (a factor taking into account the decrease in the value for location-aware IOs by applying a penalty according to the delivery location):

$$VoI_{\Theta}(m, \mathbb{I}_m, fs, \mathbb{MR}(m)) = VoI_0(m, \mathbb{I}_m, fs) \times \sum_{r \in \mathbb{MR}(m)} [TRD(r^t, m^{ot}) \times PRD(r^l, m^{ol})] \quad (8)$$

where  $\mathbb{MR}$  is the (set of) receivals for message  $m$ ,  $TRD(r^t, m^{ot})$ , as in “Timeliness Relevance (of Request) Decay”, is a factor that takes into account the loss of VoI in the time passed between request generation time  $r^t$ , and response generation (the current time  $m^{ot}$ ). Finally,  $PRD(r^l, m^{ol})$ , as in “Proximity Relevance (of Request) Decay”, is similar to the  $TRD(r^t, m^{ot})$  component, but its impact on the final VoI score depends on the physical distance between the position of the requester, given by  $r^l$ , and that of the source of IO, provided by  $m^{ol}$  (in case message  $m$  was simultaneously issued by several requesters,  $r^l$  returns the location of the requester closest to the IO source). With regards to the general case in which an IO  $m$  is constructed from more raw-data messages, the centroid of the origin points of each raw-data that contributed to its generation is considered for the purpose of PRD calculation. This also applies to CRIO messages resulting from the composition of multiple IOs.

### B. User-specific VoI Model

To account for user specific utilities we could add to equation (8) a component ( $U$ ) that considers utility for each user receiving message  $m$ :

$$VoI_{\Theta}(m, \mathbb{I}_m, fs, r, m, t) = VoI_0(m, \mathbb{I}_m, fs) \times \sum_{r \in \mathbb{MR}(m)} [TRD(t, m^{ot}) \times PRD(r^l, m^{ol}) \times U(r^u, m, t)] \quad (9)$$

where  $U(r^u, m, t)$  is the utility of message  $m$  for user  $r^u$  at time  $t$ .

Note that this model does not specify the behavior of utility function  $U$ , which is likely to be user-specific and context dependent. Most importantly, forecasting the  $U$  component in equation (9) represents a **challenging task**, even more than the forecasting of the sum component in equation (8).

The practical adoption of a high complexity model such as this requires the adoption of *simulation-based optimization*. Simulation-based optimization [25] is based on the creation of a simulated model and the optimization of its input parameters. Therefore, it is possible to verify how different service component allocations perform within a defined scenario by evaluating the total VoI collected within the given time window.

### C. Model Simplifications

We formulate a simplification of the model defined in equation (9) to reduce its complexity to a manageable degree in order to avoid simulation based approaches for its evaluation. More specifically, we first assume that  $VoI_0$  only depends on message type ( $m_t$ ) and service profile:

$$VoI_0(m, \mathbb{I}_m, fs) \approx VoI(m_t, fs) \quad (10)$$

In addition, we assume that it is possible to approximate the sum component in equation (9) by clustering messages and users in a limited number of classes, calculating an average TRD and PRD for each class of messages, and calculating an average utility per message and user type. More specifically, we have:

$$\sum_{r \in \mathbb{MR}(m)} TRD(t, m^{ot}) \times PRD(r^l, m^{ol}) \times U(r^u, m, t) \approx \sum_{m_t \in M} [TRD(m_t) \times PRD(m_t) \times \sum_{u_t \in U} U(u_t, m_t)] \quad (11)$$

where  $m_t$  represent a class of messages.

With regard to message clustering, we can use different policies. For instance, a natural way of clustering messages is according to their content or service type. User clustering could be performed according to several criteria, including location, profile type, and other criteria. For location-based user clustering, we could model the number of users at a given time  $t$  thought the concept of user group. A user group  $u_k$  represents a set of users in a defined location  $d^{u_k}$  with share

of interest  $w_{sj}(k)$  in consuming CRIOS produced by a service  $s_j$ . At a given time  $t$ , the number of users within a user group  $u_k$  is fixed and it is expressed as  $u_k(t)$ .

However, considering the PRD and TRD of equation (11) is still challenging since they are IO specific. In addition, service components defining the workflow of a service, are likely to be instantiated on different fog devices. Therefore, the calculation of each PRD and TRD requires to consider the location of the fog device where the service component  $sc_k$  is instantiated on.

It is possible to improve the approximations made in equation (11) by analyzing in depth the processing stages defined in the AIV information maturity model: raw-data collection, IOs processing, CRIOS delivering. Since, information latency is strictly related to the distance each information has to travel, let us consider the PRD components first. More specifically, we can relate PRD components to their processing stage to express respectively: proximity to data sources and devices (raw-data), proximity to devices running service components belonging to the same service (IOs), and proximity of fog devices and users (CRIOS). The first takes into account the decay the information is subjected to from the generation of raw-data to its first processing phase. The second is to consider the information-delay in the processing chain, e.g. related service components should be allocated close to each other to obtain reduced distance and latency. Finally, the last PRD component considers the proximity of the device generating the information in its final stage (CRIOS) to the users interested in consuming it. It is worth nothing that the same considerations can be done for TRD components.

PRD components can be feasible or infeasible with respect to the distance from information source to information recipient. Feasibility is determined accordingly to the constraints depicted in equation (12), where  $d^{l_i, sc_j}$  indicates the location of the device where the service component  $sc_j$  is allocated on,  $ds_i^l$  the location of the data source  $ds_i$ ,  $sc_{j+1}$  the service component next to  $sc_j$  in the service workflow, and  $u_k^l$  the location of the user group  $u_k$ .

$$\begin{aligned} distance(d^{l_i, sc_j}, ds_i^l) &\leq max\_distance \\ latency(d^{l_i, sc_j}, ds_i^l) &\leq max\_latency \\ distance(d^{l_i, sc_j}, d^{l_i, sc_{j+1}}) &\leq max\_distance \\ latency(d^{l_i, sc_j}, d^{l_i, sc_{j+1}}) &\leq max\_latency \\ distance(d^{l_i, sc_j}, u_k^l) &\leq max\_distance \\ latency(d^{l_i, sc_j}, u_k^l) &\leq max\_latency \end{aligned} \quad (12)$$

$$\min_{DS, D, U} (\omega_{RD} \times PRD_{RD} + \omega_{ios} \times PRD_{ios} + \omega_{crios} \times PRD_{crios})$$

$$\begin{aligned} PRD_{RD} &\approx \sum_{sc_i \in SC} [distance(d_i^l, ds_i^l)] \\ PRD_{ios} &\approx \sum_{sc_i \in SC} [distance(d_i^l, d_j^l)] \\ PRD_{crios} &\approx \sum_{sc_i \in SC} [distance(d_i^l, u_k^l)] \end{aligned} \quad (13)$$

TABLE II  
DISTANCE BETWEEN USER GROUPS AND FOG DEVICES.

|       | $u_1$   | $u_2$   | $u_3$   | $u_4$   |
|-------|---------|---------|---------|---------|
| $d_1$ | 723.55  | 1586.61 | 0       | 339.74  |
| $d_2$ | 158.90  | 2240.61 | 702.95  | 366.34  |
| $d_3$ | 414.92  | 1884.99 | 339.74  | 0       |
| $d_4$ | 0       | 2297.23 | 723.55  | 414.92  |
| $d_5$ | 2297.23 | 0       | 1586.61 | 1884.99 |
| $d_6$ | 2091.34 | 206.00  | 1380.80 | 1679.33 |
| $d_7$ | 136.19  | 2189.24 | 606.02  | 327.48  |

We can formalize these considerations into equation (13), which defines the objective function of the approximated model and all its components, where  $RD$  stands for raw-data. The allocation resulting from the optimization of equation (13) will privilege lower distance, thus minimizing the delay within the overall chain defining a service.

More specifically, the resulting allocation will try to allocate service components within the same workflow in a way that the distance between the devices where those service components are allocated is minimal. Furthermore, the objective function tries to maximize the single service component utility, e.g. service component allocations allowing the processing of raw-data from multiple data sources (or to serve multiple user groups) are preferred to other allocations.

## VII. EXPERIMENTAL EVALUATION

To validate the proposed models, we used meta-heuristics algorithms to optimize the service fabric of a Fog Computing applications using both simulation-based optimization implementing the user-specific model in equation (9) and the approximated model defined in equation (13). In particular, for this work we consider the adoption of Genetic Algorithm (GA) and Quantum Particle Swarm Optimization (QPSO) to find a solution for both formulations. As optimization library, we exploited the capabilities of ruby-mhl (<https://github.com/mtortonesi/ruby-mhl>) for both GA and QPSO.

In order to test the performance of the two models, we devise a testbed reenacting a realistic Fog Computing scenario. This testbed contains the description of 7 devices, 9 data sources, 4 user groups, and 8 service components. Within the scenario, each Fog Computing service is defined as a workflow of 2 service components. The locations of devices, data sources, and user groups are defined in the scenario with a latitude and longitude position in accordance with the GPS position system. In particular, the distance between user groups and devices is depicted in Table II.

### A. Evolutionary Algorithms Configuration

Evolutionary algorithms represent an interesting and valuable technique to solve the value-based service placement optimization problems defined in equations (9) and (13). In particular, we used genetic algorithms to find an approximated solution to the VoI service placement problem. To this end, Fig. 1 depicts the chromosome representation for the allocation

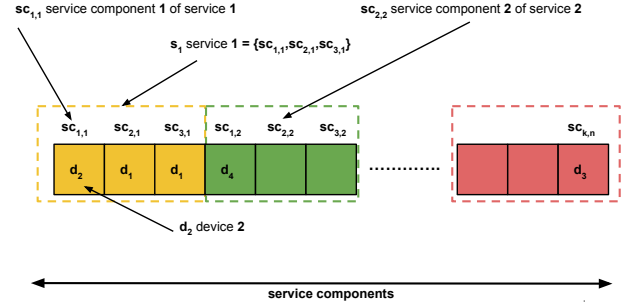


Fig. 1. Chromosome representation. Each gene in the chromosome represents the mapping between a service component and a fog device.

problem. In particular, we used an integer chromosome representation, in which genes represents the mapping of service components on fog devices. In more words, the value of a gene indicates the device where that service component will be allocated. More specifically, it is possible to identify each service component  $sc_k$  as part of service  $s_j$  by adopting the notation  $sc_{k,j}$ , where  $k$  identifies the component within the workflow of service  $s_j$ . In the representation depicted in Fig. 1, service components belonging to the same workflow are adjacent genes. Furthermore, let us note that this representation implicitly solves the constraint defined in equation (2), by allocating each service component to a fog device. On the whole, it is possible to use this notation with other types of evolutionary algorithms such as Particle Swarm Optimization (PSO) and its variants.

Instead, from a resource consumption perspective, each service component  $sc_k$  requires an amount of  $sc_k^{rr}$  resources at time  $t$ . Considering that, this optimization problem is solved for a given system state, the amount of resources allocated is monitored for the corresponding time window. Therefore, service components allocated on a device  $d_i$  must respect the constraint defined in equation (3) for the given system state. Finally, service component allocations must respect the constraints defined in equation (12).

### B. Implementation of VoI Models

To help readers understand the following results we give a description of the implementations of the user-specific model defined in equation (9) and the approximated model defined in equation (13). Let us note that, in the implementation of the model (9), we defined the utility function  $U$  as a table depicting the utility of a message of class  $m_t$  to cluster of users.

On the one hand, the user-specific model is implemented in the Phileas simulator. More specifically, Phileas takes as input a configuration describing devices, users, service of a realistic Fog Computing and an allocation array describing the allocation of service components on devices to return as output the total VoI collected in a one hour time window. For the sake of these experiments, the input configuration is fixed, while the input allocation array is the parameter that the optimization process changes to maximize the output VoI. To guarantee

the reproducibility of the simulation, all the random variables responsible for the event generation have a fixed seed.

On the other hand, to find a solution to the approximated model defined in equation (13), we implement a fitness function algorithm that tries to minimize the overall PRD components. The fitness function algorithm illustrated in Alg. 1 is a function that evaluates an input allocation and checks for its feasibility.

---

**Algorithm 1:** Fitness function algorithm

---

**Data:** An allocation array  $x$   
**Result:** Sum of the PRD components associated to allocation  $x$

```

prds_sum = 0;
forall Device  $d$  do
    resources_on_d = check_assigned_resource(d);
    if unfeasible(resources_on_d) then return  $\infty$ ;
sc_raw_data = find_sc_raw_data(x);
forall  $sc$  in sc_raw_data do
    device =  $x[sc]$ ;
    sc_ds_distance = calculate_distance(device, DS);
    min_distance = min(sc_ds_distance);
    if unfeasible(min_distance) then return  $\infty$ ;
    prds_sum += sum(sc_ds_distance);
forall  $s$  in  $\mathbb{S}$  do // service-chain feasibility
    scs = find_scs(s,x);
    forall  $sc$  in scs do
         $d_i = x[sc]$ ;  $d_j = x[sc + 1]$ ;
        processing_distance = distance( $d_i, d_j$ );
        if unfeasible(processing_distance) then return  $\infty$ ;
        prds_sum += processing_distance;
sc_crios = find_sc_crios(x);
forall  $sc$  in sc_crios do // CRIOS delivering feasibility
    device =  $x[sc]$ ;
    sc_ugs_distance = calculate_distance(device, U);
    min_distance = min(sc_ugs_distance);
    if unfeasible(sc_ugs_distance) then return  $\infty$ ;
    s_ugs = select_unfeasible(sc_ugs_distance);
    prds_sum += sum(selected_ugs);
return prds_sum;

```

---

In Alg. 1 each PRD component is evaluated to verify the feasibility of the input allocation considering both the constraints defined in equation (3) and equation (12), thus avoiding to select infeasible allocations. Solutions not respecting those constraints are invalidated by assigning them infinite values, and thus discarded from the population. Therefore, the function defined in Alg. 1 drives the optimization algorithm to solutions capable of minimizing the overall distance of data sources, devices, and users, thus finding an optimal allocation that privileges lower PRD and TRD components.

Finally, the result of the fitness function is the sum of PRD components related to the input allocation array. To relate the output of the fitness function into a VoI value, the best allocations found during the optimization process are given as input to the Phileas simulator, which returns the total VoI collected in a given system state.

### C. Results

Let us specify that the output values deriving from the optimization of the approximated model and the simulated model are not directly comparable since the first does not express a VoI value but the summation of PRD components, which the function defined in Alg. 1 tries to minimize. On the other hand, the output value of the simulation-based optimization represents the total VoI collected in the given system state. Since both models require an allocation array (where to allocate service components) as input parameter, we evaluated the allocation arrays obtained iterating the evolutionary algorithms on the approximated model within Phileas. In this way, it is possible to observe on the same scale and to verify if the approximated model is a good approximation of the simulated one.

The first experiment consists in testing the two evolutionary algorithms on the simulation-based model. The input parameter for the optimization process is the allocation array and the output is the total VoI of the receivals generated during the simulation of one hour time window. The optimization algorithm iterates over to find the best allocation for service components on fog devices for a maximum of 100 generations. To this end, Fig. 2 depicts how the two evolutionary algorithms behave in optimizing the total VoI in the given time-window. In this comparison, QPSO proved its validity in finding an allocation resulting in a higher total VoI and a faster convergence speed.

Another goal of this work is to evaluate the performance of two evolutionary algorithms in optimizing the fitness function of the approximated model. We run this experiment by setting the number of generations for both the QPSO and GA to 850. We chose to increase the number of generations in the approximated model since the optimization process does not require to evaluate each allocation array within the simulator. Fig. 3 illustrates the performance of QPSO and GA in optimizing the fitness function. In this experiment, the GA relatively outperforms Quantum PSO in minimizing the fitness function. More specifically, GA finds an optimal allocation around the 400-th iteration. However, both algorithms demonstrate good performance in finding an optimal solution with a tiny gap during the entire process.

At the other end, Fig. 4 shows the total VoI produced by the allocations generated from the optimization of the fitness function. As shown in Fig. 4 both GA and PSO find acceptable solutions. GA finds an allocation capable of generating a greater VoI around the 400-th iteration. These results indicate that the approximated model is good enough as first estimation but still requires investigations and improvements. However, considering that approximating the VoI function is a challeng-

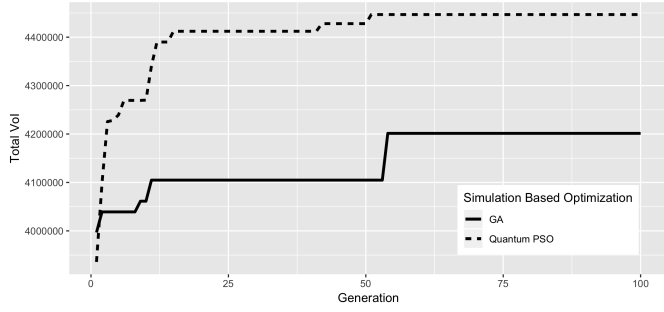


Fig. 2. Total VoI for each iteration of the two evolutionary algorithms on the oracle model.

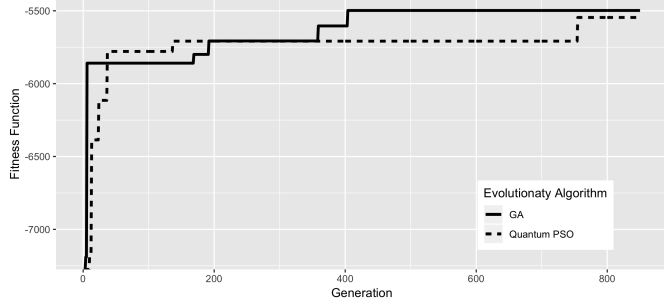


Fig. 3. Fitness function value obtained with the two evolutionary algorithms.

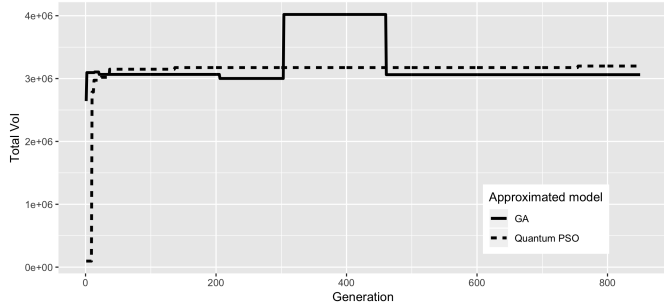


Fig. 4. Total VoI obtained using the allocations produced by the optimization of the fitness function.

ing task, those results are promising in all those scenarios requiring a quick allocation or re-allocation of resources.

Then, we compared the results of the approximated model and the simulated model in Fig. 5 and Fig. 6. More specifically, Fig. 5 compares the gap between the approximated and simulated model for 100-th generations of QPSO, while Fig. 6 shows the same comparison for GA. The results show a gap between the approximated and simulated model in finding an optimal VoI allocation, which is mainly due to the approximations made to define the fitness function. However, we believe that those results are acceptable as a first approximation in finding a quick and satisfying resource allocation in Fog Computing environments. In addition, further enhancements can be made to improve the performance of the approximated model. Finally, Table III reports the configurations and the execution time, measured via the time UNIX utility, to run the optimization on both the simulated-based model and the approximated one.

## VIII. CONCLUSION

This work presented our efforts in formalizing a system model notation for value-based service fabric management for

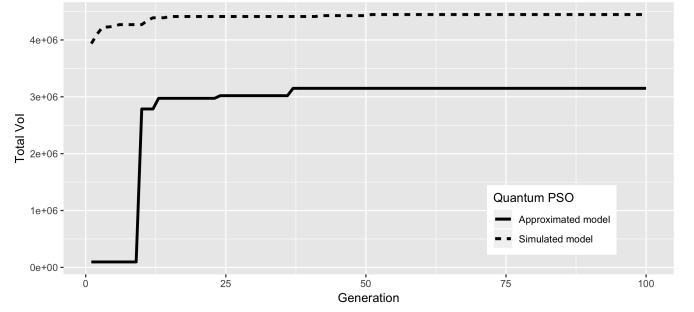


Fig. 5. Comparison between the simulated model and the approximated model using QPSO.

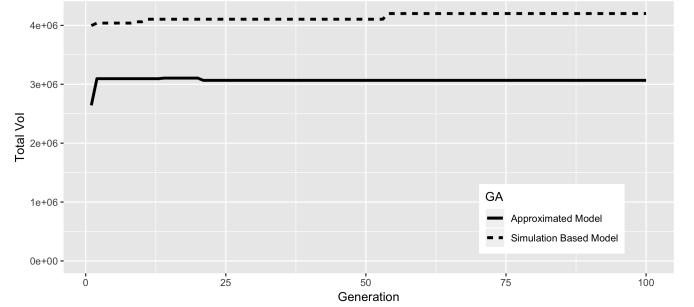


Fig. 6. Comparison between the simulated model and the approximated model using GA.

TABLE III  
EXECUTION TIME OF THE OPTIMIZATION MODELS

| Model            | EA   | Population Size | Iterations | Execution Time |
|------------------|------|-----------------|------------|----------------|
| Approximated     | GA   | 128             | 850        | 6.53s          |
| Approximated     | QPSO | 40              | 850        | 2.87s          |
| Simulation based | GA   | 128             | 100        | 10030.43s      |
| Simulation based | QPSO | 40              | 100        | 4113.16s       |

Fog Computing. In particular, we proposed two formulations: a user-specific VoI model that we implemented within the Phileas simulator and an approximated model that tries to allocate resources by minimizing the overall distance and latency within the scenario. Finally, the two approaches are compared with detailed experimental results that show how GA and QPSO are capable of finding feasible and acceptable solutions.

It is worth noting that optimizing the approximated model is much less computationally expensive than the simulation-based optimization process. In fact, the latter requires to run a simulator for each member of the population, while the former only requires to evaluate the value of the fitness function defined in equation (13).

Future works will entail the adoption of reinforcement learning [26] techniques as another optimization option for service fabric management of Fog Computing applications. This technique could be applied to our model by measuring the delivered VoI (or a portion of it) within a time-window and feed it to a self-learning allocator component within a reinforcement learning framework.

## REFERENCES

- [1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC 12*. ACM Press, 2012. [Online]. Available: <https://doi.org/10.1145/2342509.2342513>
- [2] S. Bharti, K. K. Pattanaik, and P. Bellavista, "Value of information based sensor ranking for efficient sensor service allocation in service oriented wireless sensor networks," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2019.
- [3] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, vol. 98, pp. 289–330, Sep. 2019. [Online]. Available: <https://doi.org/10.1016/j.sysarc.2019.02.009>
- [4] C. Wu and L. Wang, "A deadline-aware estimation of distribution algorithm for resource scheduling in fog computing systems," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, June 2019, pp. 660–666.
- [5] R. Kouah, A. Alleg, A. Laraba, and T. Ahmed, "Energy-aware placement for iot-service function chain," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, Sep. 2018, pp. 1–7.
- [6] D. Omer, R. Aburukba, and T. Landolsi, "Optimization model for time sensitive iot requests," in *2019 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, March 2019, pp. 1–4.
- [7] S. Zheng, Z. Gao, X. Shan, W. Zhou, and Y. Wang, "Tail Latency Optimized Resource Allocation in Fogbased 5G Networks," in *2018 IEEE Symposium on Computers and Communications (ISCC)*, June 2018, pp. 00 249–00 254.
- [8] D. Turgut and L. Boloni, "Value of information and cost of privacy in the internet of things," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 62–66, Sep. 2017.
- [9] L. Bölöni and D. Turgut, "Value of information based scheduling of cloud computing resources," *Future Generation Computer Systems*, vol. 71, pp. 212 – 220, 2017. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167739X16304472>
- [10] M. Giordani, T. Higuchi, A. Zanella, O. Altintas, and M. Zorzi, "A framework to assess value of information in future vehicular networks," in *Proceedings of the 1st ACM MobiHoc Workshop on Technologies, mOdelS, and Protocols for Cooperative Connected Cars*, ser. TOP-Cars '19. New York, NY, USA: ACM, 2019, pp. 31–36. [Online]. Available: <http://doi.acm.org/10.1145/3331054.3331551>
- [11] N. Suri, G. Benincasa, R. Lenzi, M. Tortonesi, C. Stefanelli, and L. Sadler, "Exploring value-of-information-based approaches to support effective communications in tactical networks," *IEEE Communications Magazine*, vol. 53, no. 10, pp. 39–45, October 2015.
- [12] J. R. Michaelis, "Value of information driven content management in mixed reality infrastructures," vol. 10653, 2018. [Online]. Available: <https://doi.org/10.1117/12.2303997>
- [13] M. Tortonesi, M. Govoni, A. Morelli, G. Riberto, C. Stefanelli, and N. Suri, "Taming the IoT data deluge: An innovative information-centric service model for fog computing applications," *Future Generation Computer Systems*, vol. 93, pp. 888 – 902, 2019.
- [14] F. Poltronieri, C. Stefanelli, N. Suri, and M. Tortonesi, "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)*, Sep. 2018, pp. 1–6.
- [15] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, "The internet of things, fog and cloud continuum: Integration and challenges," *Internet of Things*, vol. 3-4, pp. 134 – 155, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2542660518300635>
- [16] C. Guerrero, I. Lera, and C. Juiz, "Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures," *Future Generation Computer Systems*, vol. 97, pp. 131 – 144, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167739X18325147>
- [17] J. Yao and N. Ansari, "Qos-aware fog resource provisioning and mobile device power control in iot networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 167–175, March 2019.
- [18] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner, "Optimized iot service placement in the fog," *Service Oriented Computing and Applications*, vol. 11, no. 4, pp. 427–443, Dec 2017. [Online]. Available: <https://doi.org/10.1007/s11761-017-0219-8>
- [19] C. Canali and R. Lancellotti, "A fog computing service placement for smart cities based on genetic algorithms," in *CLOSER*, 2019.
- [20] S. Saraswat, H. P. Gupta, T. Dutta, and S. K. Das, "Energy efficient data forwarding scheme in fog based ubiquitous system with deadline constraints," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2019.
- [21] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and Information Systems*, vol. 51, no. 2, pp. 339–367, Sep. 2016.
- [22] T. Dasu, S. Krishnan, and G. M. Pomann, "Robustness of change detection algorithms," in *Advances in Intelligent Data Analysis X*. Springer Berlin Heidelberg, 2011, pp. 125–137.
- [23] R. Killick, *Introduction to optimal changepoint detection algorithms*. 2017 (accessed June 21st, 2019). [Online]. Available: <http://members.cbio.mines-paristech.fr/~thocking/change-tutorial/RK-CptWorkshop.html>
- [24] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes - Theory and Application*, 1993 (accessed June 21st, 2019). [Online]. Available: <http://people.irisa.fr/Michele.Basseville/kniga/>
- [25] A. Gosavi, *Simulation-Based Optimization*, 2nd ed. Springer US, 2015.
- [26] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Bradford Books, 2018.