Studying the end-to-end performance, energy consumption and carbon footprint of fog applications

Clément Courageux-Sudan, Anne-Cécile Orgerie, Martin Quinson

Studying the end-to-end performance, energy consumption and carbon footprint of fog applications

Clément Courageux-Sudan*, Anne-Cécile Orgerie*, Martin Quinson*
*Univ. Rennes, Inria, CNRS, IRISA, France
Email: {clement.courageux-sudan, anne-cecile.orgerie, martin.quinson}@irisa.fr

***Abstract*—The deployment of applications closer to end-users through fog computing has shown promise in improving network communication times and reducing contention. However, the use of fog applications such as microservices necessitates intricate network interactions among heterogeneous devices. Consequently, understanding the impact of different application and infrastructure parameters on performance becomes crucial. Current literature either offers end-to-end models that lack granularity and validation or fine-grained models that only consider a portion of the infrastructure. Our research first compares experimentally the accuracy of the existing integrated frameworks. We then combine one of these tools with a collection of validated models to obtain comprehensive metrics regarding microservice applications operating in the fog. Through a use-case, we demonstrate the effectiveness of our approach in investigating fog environments, from examining application latencies to greenhouse gas emissions.**

***Index Terms*—Modeling and simulation, Fog-computing, Microservice, Performance evaluation, Energy consumption**

I. Introduction

Cloud computing consists in centralizing the computations of distributed systems into large datacenters to reduce the costs through improved operational efficiency. Fog computing adapts this approach by moving some application components closer to the end-user to reduce the network congestion and delays that are hindering pure cloud architectures. To that extent, fog platforms add a layer of micro-datacenters with less efficient, geographically distributed resources. Such platforms prevent the use of monolithic applications, which must instead be split into multiple components. Under the *microservices* software architecture, the applications consist of lightweight interconnected services processing and exchanging data to fulfill requests [23]. Latency-sensitive services can be placed in the fog while CPU-intensive processes can be placed in the cloud to optimize their performance.

Regardless of its potential, fog performance can be jeopardized by threats that are unknown in traditional clouds. The platform heterogeneity along with the application complexity lead to intricate placement decisions [17], further complicated by the geographical distribution of nodes resulting in a tradeoff between communication delays and performance gains enabled by powerful nodes located in the cloud, potentially far from the user. In addition, the changing platform conditions and workload call for automatic adaptation of the application through the migration, replication or consolidation of services. Finally, microservice applications can be evaluated under several loosely related metrics, such as response time, resource occupation, energy consumption, or greenhouse gas emissions.

It is thus crucial to analyze the benefits and drawbacks of the deployment and operation strategies of fog applications. The intrinsic constraints of real experiments hinder their applicability to this context, since back-to-back experiments can lead to different results depending on the platform conditions. Another approach is to transpose applications into models to be used in simulations for better scalability and control. Despite the potential of simulation to understand fog applications' performance, existing tools and models make it very difficult to globally assess fog performance from the users' devices up to the Cloud servers along with all networking components according to several metrics and criteria of interest.

Different models are proposed in the literature to analyze individual aspects of the fog such as network, computing nodes, or energy. This focus restrains their usage to specific applications or metrics, due to the difficulty of combining different simulation frameworks. In addition, the models' granularity does not always enable simulating large infrastructures without prohibitively long simulations. Other frameworks propose to simulate entire fog infrastructures from end-users to cloud servers [23], [21], [2], but the experimental validation of their models is often very partial at best. There is thus a need for a unified tool to analyze fog applications using validated models in a single framework to allow the simulation of different network and application architectures.

The main contributions of this paper are:

- Reviewing energy and performance models for fog infrastructures: devices, communications, and data centers;
- Comparing the predictions from four simulation frameworks against real-world network measurements.
- Combining validated models to simulate fog applications with performance, energy, and GHG metrics in order to compare microservices deployment strategies;
- Simulating an application under different deployment and infrastructure configurations in an end-to-end manner. This use-case illustrates the effectiveness of our approach for studying trade-offs between energy and performance, communication delays and computing capabilities.

The rest of this paper is structured as follows. Section II proposes an overview of simulation tools for the study of fog

TABLE I: Comparison of state-of-the-art simulation models

Simulator	Network		Computing		Energy		Applications	Scalability
	Wired	Wi-Fi	CPU	Memory	Network	Computing		
ns-3	✓	✓	✗	✗	✓	✗	-	Packet-level
OMNET++	✓	✓	✓	✓	✓	✓	-	Packet-level
CloudSim Plus	✓	✗	✓	✓	✗	✓	Cloud	Packet-level
DISSECT-CF	✓	✗	✓	✓	✓	✓	Cloud	Provider/Consumer
μqsim	✓	✗	✓	✓	✗	✗	Microservices	Queue-based
YAFS	✓	✗	✓	✓	✗	✓	Microservices	Queue-based
IFogSim	✓	✗	✓	✓	✗	✓	Microservices	CloudSim-based
SimGrid	✓	✓	✓	✗	✓	✓	Microservices/Other	Flow-level

infrastructures. In Section III, we compare the accuracy of four simulation frameworks to real-world measurements. Based on this comparison, Section IV presents validated models to study end-to-end fog infrastructures and applications. Section V proposes an overview of the metrics available using our approach through a use-case. Section VI concludes this work.

II. State of the Art

Extensive literature is available on the simulation of fog applications. The existing works fall into two categories: models intended to study a single part of fog infrastructures and frameworks enabling end-to-end analysis of fog applications.

A. Single component models

Models exist to study various parts of fog infrastructures and applications. Network simulators such as ns-3 [24] and OMNET++ [31] provide validated models for many communication technologies such as Wi-Fi or Ethernet, and can be used to study for example the impact of the network configuration on applications' latency. Other models focus on the simulation of processing infrastructures used to execute applications. For example, CloudSim Plus [28] makes it possible to simulate cloud datacenters. It can reproduce complex cloud scenarios, but lacks wireless network models used in the fog. Application models such as μqsim for microservices [35] evaluate the application's use of resources. But μqsim is based on simple network models not reflecting the complexity of fog networks. Estimating the energy consumption of fog applications is also important since it induces most of the operational costs [20]. DISSECT-CF [19] proposes a framework to study the energy consumption of Infrastructure-as-a-Service clouds.

These models can provide metrics for different parts of fog infrastructures. However, the models' implementation within separate frameworks complicates the study of their interdependencies and requires running multiple simulations for the end-to-end analysis of a single scenario [12]. Separate tools also do not allow capturing the interdependencies between all models. Other models are specific to one type of application, like smart-health applications [16], and thus difficult to extend to other scenarios.

B. End-to-end fog models

Since models are created to study well-defined scientific questions [30], some simulation frameworks focus on the study of fog infrastructures. The IFogSim and IFogSim2 [13], [23] simulators aim to simulate end-to-end fog infrastructures running microservices. The many models available in IFogSim enable simulations considering mobility, scheduling, and energy usage. Despite the capability of this framework, some works show validity limitations of the underlying CloudSim execution and network models [26], [33]. Frameworks such as IoTSim-Osmosis [2] or YAFS [21] have similar features to study end-to-end fogs, but their coarse-grained models have not been validated. Their communication models do not permit defining shared communication channels, very common in the fog with Wi-Fi. Their cost per usage energy models also lack validation, despite the importance of understanding models' validity limitations to avoid biased results [32].

Distributed systems simulators such as SimGrid do not target fog infrastructures but propose validated models for the technologies used in the fog. Table I summarizes the models available in the frameworks introduced in this section.

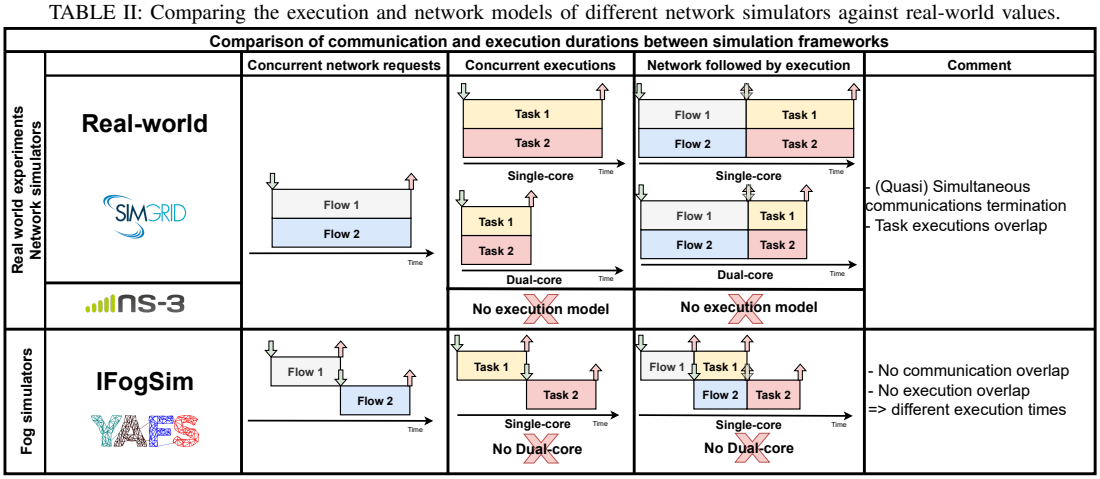
III. Experimental comparison of fog simulation frameworks

In this section, we compare real-world measurements of task executions and network communications to the predictions of four state-of-the-art simulation frameworks: ns-3 [24], SimGrid [32], YAFS [21], and IFogSim [13], [23]. As shown in Table I, YAFS and IFogSim are simulation frameworks tailored for fog infrastructure studies. SimGrid is more versatile, incorporating network models, execution, and application models. ns-3 is limited to the study of network communications, but the accuracy of its results makes it a popular framework. Other simulators exist but we consider the present selection because of their popularity in their research fields, and their representativity of the different models' granularity (flow-, packet-, and queue-based simulation).

A. Methodology

We describe the setup used to measure real and simulated network and task durations. We use the latest versions of IFogSim (2.0.0) and YAFS, ns-3 (v3.39), and SimGrid (v3.32). We compare simulators to results of two servers from the Grid'5000 testbed [3] with 2xIntel Xeon E5-2630 v3 CPUs and 2 x 10Gbps Ethernet interfaces. Information with reproducible experiments and results are available online: https://github.com/klementc/end-to-end-fog-reproducibility

TABLE II: Comparing the execution and network models of different network simulators against real-world values.



1) Network flows: We limit the bandwidth between the testbed servers to 50Mbps using the network emulation feature of EnosLib [4]. The latency between nodes is negligible in this setup, below 0.2ms from our measurements, which is much lower than the data transmission duration. We create between 1 and 10 TCP flows of equal size starting simultaneously from the first node towards the second node using iperf [29]. Simulators are calibrated to match this setup with two nodes, a 50Mbps Ethernet link between the nodes, and a 0ms latency. We compare the start and finish timestamps of each flow.

2) Task executions duration: A script runs on testbed servers to execute a fixed amount of CPU work. Between one and ten executions of this script start simultaneously. We measure each task's beginning and termination timestamps. Simulations are calibrated by setting the capacity of the nodes to 1 GFlops with the same cost of tasks as in the real-world results. Real-world and SimGrid scenarios are executed either on a single core or across two cores. We did not manage to use IFogSim to perform multi-core executions, while YAFS models do not permit it. ns-3 has no execution model and is only used to measure network durations.

B. Observations

Table II summarizes the results. We use Gantt charts to illustrate the start and termination timestamps of communications and executions. We classify the results into two groups.

1) Resource sharing models: As shown in Table II, ns-3 and SimGrid network models share the bandwidth of the network links between active flows. SimGrid flows start and terminate at the same time given the use of a fair bandwidth allocation model. ns-3 results are similar, with small variations due to more fine-grained packet-level bandwidth-sharing models. Task executions use the same sharing approach in SimGrid, where using two cores divides the execution time of parallel tasks by two. These results are consistent with the real-world measurements for both network and tasks.

2) Exclusive resource usage models: Table II shows variations between IFogSim [23], YAFS [21], and real-world results. Both simulators provide exclusive access to network and execution resources: a communication or execution makes full use of the resource until termination. Network communications and executions starting simultaneously terminate at different timestamps. Despite coherent overall communication duration, single network and execution termination timestamps are very different from real measurements.

C. Discussion

Based on these observations, YAFS [21] and IFogSim [23] are limited to the study of scenarios with exclusive access to resources. Otherwise, the estimations of these simulators can greatly vary from real-world values. Also, their results are limited to wired communications since YAFS and IFogSim do not differentiate between Ethernet and Wi-Fi channels. This can impact the results of fog studies since Wi-Fi channels are popular at the edge of the network, especially in mobility scenarios. Our experiments using IFogSim also show unexplained results highly impacted by undocumented parameters that can not be adjusted without side effects (i.e. *schedulingInterval*).

In the rest of this paper, we chose to simulate fog infrastructures using models validated in the literature, matching our real-world observations. Our choice is to use SimGrid since it provides both network and execution models.

IV. End-to-End Modeling of a Fog Infrastructure and its Applications

We propose a methodology based on the validated SimGrid models to accurately simulate computing devices, network interfaces, and their energy consumption. We show how these models can be combined to experiment with fog platforms.

A. Infrastructure model

All fog infrastructures have common components. We split it into **1)** end-users, **2)** fog micro-datacenters, **3)** cloud datacenter. This infrastructure corresponds to a graph $G(N, L)$ where vertices $n \in N$ are computing nodes, and edges $l \in L$ are network links (Ethernet or Wi-Fi) between pairs of nodes.

All machines in the fog infrastructure can execute tasks. The optimal amount of concurrent tasks on a node depends

on its number of CPU cores N_{core}. Each core has a processing capacity C_{core} expressed in *floating-point operations per second* (flops), for a maximum capacity of $N_{core} * C_{core}$ flops. Table III provides an example of calibration values for a server from the experimental testbed Grid'5000 [3] and a Raspberry Pi 4 model B [18]. To simulate task executions, we use the execution model of SimGrid. Properly calibrated, it estimates task execution durations for different types of applications, including microservices [6].

The capacity of a network link depends on the link's bandwidth BW_l in bits per second, and Lat_l latency in milliseconds. Different technologies can be used for different parts of fog networks. In this paper, we consider Ethernet and Wi-Fi, but other technologies such as cellular, LoRa, or Bluetooth networks could be considered. We use separate models to simulate each communication technology. While Ethernet links connect pairs of nodes, Wi-Fi channels are shared between all stations (STA) connected to an access point (AP), thus they use different protocols. In this work, we leverage already validated flow-based models to simulate Ethernet links [32] and Wi-Fi channels [5] implemented in SimGrid and validated by comparison with ns-3 results. Compared to packet-level network models like ns-3, flow models offer better scalability with sufficient accuracy for our purposes.

End-user devices are connected to APs using Wi-Fi. APs are then connected to the fog using Ethernet links. The core network consists of a set of core routers enabling nodes of the fog layer to communicate with the cloud datacenter. Table III gives realistic bandwidths values from the literature.

TABLE III: Models' calibration values

Machines' CPU capacity		
Device	N_{core}	C_{core}
Grid'5000 Taurus [3] (Cloud)	32	4 GFLOPS
Raspberry Pi 4B [18] (Fog)	4	1 GFLOPS
Network interfaces capacity		
Interface	Bandwidth	source
Intra-Cloud	10Gbps	[15]
Core/Edge router	48x1Gbps (avg_U=25%)	[10]
Intra-Fog	1Gbps	/
WLAN	44.23Mbps	[5]
Devices Power consumption		
Parameter	Value	source
$[P^{cloud}_{idle}, P^{cloud}_{max}]$	[94.75, 178.88]W	[14]
$[P^{fog}_{idle}, P^{fog}_{max}]$	[2.28, 6.82]W	[18]
$[P^{EU}_{idle}, P^{EU}_{max}]$	[2.28, 6.82]W	/
$[P^{eth}_{min}, P^{eth}_{max}]_{corerouter}$	[0,21.168]W	[12], [10]
$[P^{eth}_{min}, P^{eth}_{max}]_{ETH1GBPS}$	{0,0.441}W	[12]
$\{P_{idle}, P_{Tx}, P_{Rx}, P_{sleep}\}_{WiFi}$	{0.82:1.14:0.94:0.1}W	[34]
$P^{edgerouter}_{static}$	150 W	[12], [10]
$P^{corerouter}_{static}$	555 W	[12], [10]
P^{AP}_{static}	11 W	[1]
PUE_{fog}	1.7	[27]
PUE_{cloud}	1.1	[9]
GHG emissions rates in different countries		
Country	gCO2e/kWh	Source
France	56	[25]
Spain	141	[25]
Great-Britain	184	[25]
USA	388	[7]

B. Microservice application model

A microservice application is represented by a Directed Acyclic Graph (DAG) [35], [23], [6], where nodes are services and edges communications between services. When a service receives a request, it executes the corresponding task before forwarding the result to output services. As an example, Figure 1 shows the DAG of an application with four services.

We use the microservice model proposed in [6], implemented in SimGrid, where microservices are modeled as a three-step pipeline: **1)** storing received requests in a queue; **2)** executing requests on the host's resources; **3)** forwarding the result to output services. Requests are characterized by their CPU cost C_{req} (in flops) and their network size $size_{req}$ (in bits). We note the ratio between the received request's size and the size of the result sent to the output services $ratio_{I/O}$.

This model can be fed with traces from real application executions. Implemented in SimGrid, it has been validated against the DeathStarBench microservice benchmark [8].

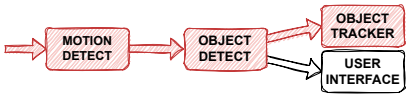


Fig. 1: DAG of the application, inspired from the surveillance camera use-case of [13]. The critical path is in red (used to measure end-to-end request latency).

C. Energy and gas emission models

Based on the infrastructure and application performance models, we estimate the energy consumption of an application. It corresponds to the sum of the energy of the computing nodes, the network interfaces, and the rest of the infrastructure (cooling, lighting). We estimate the power consumed by a device as the sum of **a)** the *static power consumption* (the minimum power to operate the device) and **b)** the *dynamic power consumption* (depending on the device's activity).

The static power consumption corresponds to the power used by the server when *idle*, noted P^{node}_{idle}. When processing some requests, the dynamic power depends on the number of active CPU cores. The device power at maximum use is P^{node}_{max}. If not used at full capacity, its power consumption is extrapolated between P^{node}_{idle} and P^{node}_{max} using linear regression as in [14]. The energy consumption of core network routers is proportional to the ratio between the data sent by the executing applications over the total data that could be sent over the router, as used in previous works [12], [22]. We use the energy model for servers from [14] implemented in SimGrid and validated against real-world measurements.

The power consumption of a network interface depends on its communication technology. We note P^{eth}_{min} the power of an Ethernet interface when *idle*, and P^{eth}_{max} at maximum use. Its power usage is proportional to the average utilization of the device $u \in [0, 1]$. Wi-Fi interfaces switch between states, each state having a different power usage. Power usage is P_{idle} when *idle*, P_{Rx} when receiving data, P_{Tx} when sending data, and P_{sleep} in sleep mode. We estimate the energy of network interfaces using the wired power model of [11] and Wi-Fi power model from [5] both validated against ns-3, calibrated with the values of Table III.

The *Power Usage Effectiveness* indicator (PUE) is employed to estimate the additional consumption to operate cloud and fog infrastructures such as cooling or lighting. The PUE value depends on the location and scale of datacenters [27] and is multiplied by the power consumption of datacenters' machines to estimate total power consumption. We use PUE values from the literature: 1.1 for high-end cloud datacenters, and 1.7 for fog micro-datacenters, as summarized in Table III. Energy consumption values allow to estimate GHG emissions. Table III provides CO_2e emissions per kWh in different countries that depend on the energy sources. We multiply energy results by this factor to estimate GHG emissions.

V. Use-Case

Based on our fog infrastructure model we evaluate the performance, energy usage, and GHG emissions of an application.

A. Setup and methodology

This study uses an application loosely inspired from [13]. This application uses four services to detect movements by processing data from video cameras. Figure 1 shows the DAG of this application, and Table IV the request processing cost for each service in MFLOPS. Experimental scenarios can be tuned to be more or less compute-intensive using different values of the computing ratio ρ: $\rho \in \{0.1, 0.5, 1\}$, and data-intensive by modifying the size of network requests: $size_{req} \in \{80kb, 1Mb\}$. We provide results for simulations considering TCP communications. We compare two application placement policies. **a)** cloud only: deployment of services in the cloud datacenter, **b)** fog only: deployment of services using fog nodes, each micro-datacenter having a copy of the application.

The application runs in an infrastructure composed of one cloud datacenter with 64 servers and 14 fog micro-datacenters. Each micro-datacenter hosts 64 Raspberry Pi model 4B. Four 802.11n access points are connected to each micro-datacenter and used by four clients to send requests to the application.

Clients send between 1 and 5 requests per second for 230 seconds. The distance between the fog and cloud datacenters varies by the number of intermediate core routers in $\{2, 5, 7\}$, each adding 5ms of latency. The latency between micro-datacenter nodes of the same cluster varies between 2 and 5 ms and is fixed to 0.5 ms between cloud servers.

At the start of the simulation, each service has one instance. Every 5 seconds, an autoscaler deploys additional service instances if the average CPU usage of the nodes hosting a service exceeds 70%. Idle nodes are initially powered off and turned on by the autoscaler when needed.

The code, and notebooks used to generate the figures are available online: https://github.com/klementc/end-to-end-fog-reproducibility. The remaining of this section details representative results obtained using the visualization interface: https://klementc.github.io/end-to-end-fog-reproducibility/energy_analysis/visualize/.

TABLE IV: CPU cost of executing a request in each service. $\rho \in \{0.1, 0.5, 1\}$ is used to modify processing intensity.

Service	C_{req} (MFLOPS)
MOTION DETECT	$\rho * 500$
OBJECT DETECT	$\rho * 250$
OBJECT TRACKER	$\rho * 500$
USER INTERFACE	$\rho * 250$

B. End-to-end latency of requests

We study a processing-intensive scenario: high CPU usage and relatively small network requests. The size of each request is 80 kbit. There are seven hops between the cloud and the fog datacenters, 2ms of latency between fog nodes, and 0.5ms between cloud nodes. The CPU execution ratio is $\rho = 1$, and $ratio_{I/O} = 1$. Figure 2 shows the evolution of end-to-end request execution times and the number of service instances.

At the beginning of the simulation, the end-to-end latency rapidly increases in both cases because of the time taken to scale the application to the workload. The cloud deployment can manage the load with 16 service instances after 20 seconds. Since fog nodes are less powerful, more than 550 service instances are used over all micro-datacenters. This scaling thus takes longer in the fog because the autoscaler adds one replica of each service per trigger.

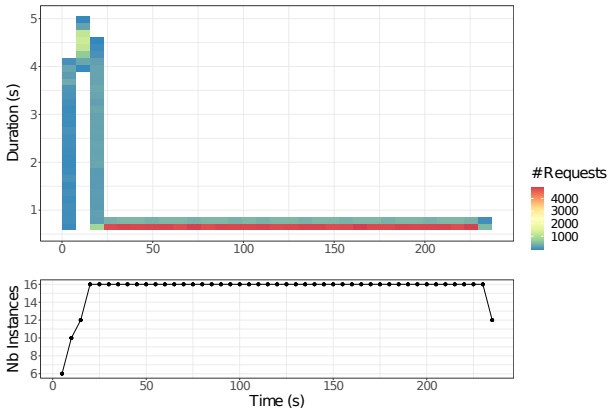
Once the number of replicas is stable, we observe more variations between the end-to-end latency of fog requests compared to cloud ones. A large number of service replicas increases the number of possible execution paths. Here, execution times dominate communication times. Lower ρ values can invert this trend, decreasing execution times and leading to faster fog end-to-end latency.

C. Impacts of network configuration

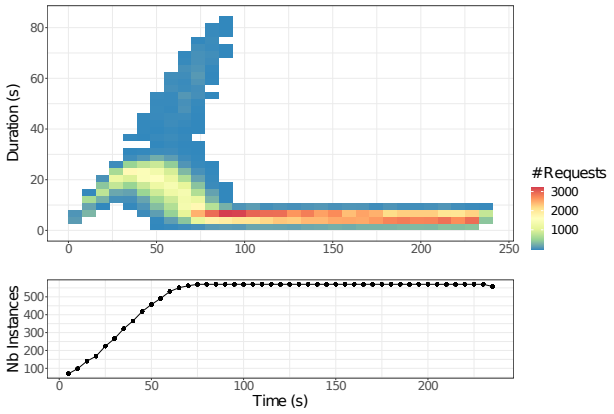
Table V compares output metrics between different network configurations. In all scenarios, $\rho = 0.1$, and $size_{req} = 1Mb$, which corresponds to a network-intensive application.

Com_{first} is the average communication time between end-users and the first service. Com_{other} is the average service-to-service communication duration. When the distance between the end-user and the cloud increases from 2 to 7 core hops, we observe a significant 198% increase of Com_{first}. It does not affect Com_{other} since the latency between cloud nodes does not change. In the fog, modifying the latency between fog nodes from 2 to 5 ms does not impact significantly Com_{first} but increases by 200% the latency between services. Regarding service executions, the average execution time of requests D_{exec} is higher for the fog application since fog micro-datacenter nodes have fewer CPU resources.

The network configuration has little impact on energy consumption: the energy used by network interfaces E_{netdev} and core routers $E_{routers}$ do not significantly change between scenarios. $E_{routers}$ in the cloud increases when adding more network hops (i.e. core routers), while no core routers are necessary in the fog. Since the workload is the same, E_{hosts} does not significantly vary with modified network parameters.



(a) End-to-end requests execution time in a cloud deployment.



(b) End-to-end requests execution time in a fog deployment.

Fig. 2: Simulating the execution time of requests under different deployment policies

TABLE V: Impact of network configuration on application metrics in the network-intensive scenario. Energy results do not consider power usage effectiveness.

	Cloud (#Hops)			Fog (lat)	
	2	5	7	2ms	5ms
Com_{first}	331ms	526ms	657ms	162ms	197ms
Com_{other}	16ms	16ms	16ms	74ms	148ms
D_{exec}	9ms	9ms	9ms	148ms	149ms
E_{netdev}	66.0kJ	66.7kJ	72.8kJ	70.2kJ	70.8kJ
$E_{routers}$	3.7kJ	9.2kJ	12.8kJ	0kJ	0kJ
E_{hosts}	425.1kJ	425.0kJ	425.1kJ	368.5kJ	368.5kJ

TABLE VI: Estimated energy consumption in infrastructure

Scenario	Total	Cloud / Edge	End-users	Network
Cloud	944.6 kJ	576.4 kJ	304.6 kJ	63.6 kJ
Fog	958.0 kJ	587.0 kJ	304.6 kJ	66.4 kJ

D. Energy consumption and GHG emissions

Table VI shows the energy consumption of different parts of the infrastructure with the CPU-intensive scenario of Section V-B. The energy of the end-users is the sum of the energy of the camera nodes and the Wi-Fi APs. Network energy is the sum of the energy of the Wi-Fi interfaces, the links between datacenters nodes, and the application's use of core routers. Network communications consume less than 10% of the overall energy. Fog nodes individually consume less power than cloud nodes, but their restricted processing capacity necessitates deploying more replicas for each service, partially offsetting their energy efficiency. This leads to higher fog energy consumption due to a high number of service replicas and less efficient PUE. This shows that efficient energy management in the fog is crucial. Data-intensive applications with low CPU usage would decrease the number of service replicas in the fog, leading to lower energy values.

Table VII shows estimated GHG emissions to run the application in the same scenario. The emissions are extrapolated for a year using the GHG emission rates in different countries from Table III.

TABLE VII: GHG emissions of the fog and cloud datacenters

Scenario	Emissions (tCO2e) by country			
	France	Spain	Great-Britain	USA
Cloud	1.05	2.64	3.44	7.25
Fog	1.07	2.69	3.50	7.39

Running a fog application in the USA, where the end-users and processing nodes are in the same neighborhood leads to more than 7 tCO_2e. Switching to a cloud placement in a French datacenter, GHG emissions drop to approximately 1.05 tCO_2e. We observe that the difference between emissions of a fog and a cloud application in the same country is very small since they have similar energy usage. While local execution in fogs sometimes improves performance and latency, datacenters' location can lead to increased GHG emissions. Traveling long distances to clusters using clean energy production methods can be more efficient from a GHG emissions point-of-view.

VI. Conclusion and Future Work

Despite its potential, fog computing remains challenging to use efficiently. Developers must tune their applications while infrastructure operators need to constantly adapt to dynamic conditions. Reducing the GHG emissions of an infrastructure without hindering the applicative performance is nearly impossible without proper evaluation tools and methodologies.

In this paper, we compare classical fog simulators through direct evidence obtained by comparing their predictions to data measured on a real platform. We discard ns-3 despite its accuracy because it lacks computations modeling. We discard iFogSim and YAFS because they only offer a one-port model e.g. at most one computation/communication per resource at a given time. SimGrid predictions are consistent with reality.

We then leverage SimGrid to propose an end-to-end modeling of fog infrastructures and their applications. We show how the existing validated models for each part of the infrastructure can be instantiated with values from the literature. Then we study a typical use case with this framework to evaluate

an application's performance, energy consumption, and GHG emissions of the underlying infrastructure in the use phase.

The experimental results on this use case show that application placement, workload, and network configuration have significant impacts. Considering and improving the PUE of fogs could allow consequent energy gains. Finally, fog computing can be negatively impacted in terms of GHG emissions depending on the location of fog clusters.

In the future, this approach could be leveraged for thorough studies of resource placement algorithms and operation strategies for fog applications. Understanding the trade-offs at stake through *what-if* scenarios before real deployments could help optimize the performance and consumption of real systems. Integrating life-cycle assessment values into our methodology would also help in understanding the complete environmental costs of the numerous fog nodes.

Acknowledgment

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr). This work received funding from the France 2030 program, managed by the French National Research Agency under grant agreement No. ANR-23-PECL-0003.

References

[1] "Électricité : combien consomment les appareils de la maison ?" report, https://agirpourlatransition.ademe.fr/particuliers/maison/economies-denergie/electricite-combien-consomment-appareils-maison, 2022.

[2] K. Alwasel *et al.*, "IoTSim-Osmosis: A framework for modeling and simulating IoT applications over an edge-cloud continuum," *J. of Systems Architecture*, vol. 116, 2021.

[3] D. Balouek *et al.*, "Adding Virtualization Capabilities to the Grid'5000 Testbed," in *Cloud Computing and Services Science*, 2013, vol. 367.

[4] R.-A. Cherrueau *et al.*, "Enoslib: A library for experiment-driven research in distributed computing," *IEEE Trans. on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1464–1477, 2021.

[5] C. Courageux-Sudan, L. Guegan, A.-C. Orgerie, and M. Quinson, "A Flow-Level Wi-Fi Model for Large Scale Network Simulation," in *Int. Conf. on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2022, pp. 111–119.

[6] C. Courageux-Sudan, A.-C. Orgerie, and M. Quinson, "Automated performance prediction of microservice applications using simulation," in *Int. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2021, pp. 1–8.

[7] "How much carbon dioxide is produced per kWh of U.S. electricity generation?" report, https://www.eia.gov/tools/faqs/faq.php?id=74, 2023.

[8] Y. Gan *et al.*, "An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems," in *Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 3–18.

[9] "Google Data Center PUE performance," https://www.google.com/about/datacenters/efficiency/, 2023.

[10] A. Gougeon, F. Lemercier, A. Blavette, and A.-C. Orgerie, "Modeling the End-to-End Energy Consumption of a Nation-Wide Smart Metering Infrastructure," in *IEEE Symp. on Computers and Communications (ISCC)*, 2022, pp. 1–7.

[11] L. Guegan, B. L. Amersho, A.-C. Orgerie, and M. Quinson, "A Large-Scale Wired Network Energy Model for Flow-Level Simulations," in *Int. Conf. on Advanced Information Networking and Applications (AINA)*, 2020, pp. 1047–1058.

[12] L. Guegan and A.-C. Orgerie, "Estimating the end-to-end energy consumption of low-bandwidth IoT applications for WiFi devices," in *IEEE Int. Conf. on Cloud Computing Technology and Science*, 2019.

[13] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017.

[14] F. C. Heinrich, T. Cornebize, A. Degomme, A. Legrand, A. Carpen-Amarie, S. Hunold, A.-C. Orgerie, and M. Quinson, "Predicting the energy-consumption of MPI applications at scale using only a single node," in *IEEE Int. Conf. on Cluster Computing*, 2017, pp. 92–102.

[15] IEEE Standards Association, "IEEE 802.3 ae-2002 IEEE standard for information technology," 2002.

[16] I. S. B. M. Isa, T. E. El-Gorashi, M. O. Musa, and J. M. Elmirghani, "Energy efficient fog-based healthcare monitoring infrastructure," *IEEE Access*, vol. 8, pp. 197 828–197 852, 2020.

[17] M. M. Islam, F. Ramezani, H. Y. Lu, and M. Naderpour, "Optimal Placement of Applications in the Fog Environment: A Systematic Literature Review," *J. of Parallel and Distributed Computing*, 2022.

[18] H. Kanso, A. Noureddine, and E. Exposito, "Automated power modeling of computing devices: Implementation and use case for Raspberry Pis," *Sustainable Computing: Informatics and Systems*, vol. 37, 2023.

[19] G. Kecskemeti, "DISSECT-CF: a simulator to foster energy-aware scheduling in infrastructure clouds," *Simulation Modelling Practice and Theory*, vol. 58, pp. 188–218, 2015.

[20] M. Koot and F. Wijnhoven, "Usage impact on data center electricity needs: A system dynamic forecasting model," *Applied Energy*, 2021.

[21] I. Lera, C. Guerrero, and C. Juiz, "YAFS: A simulator for IoT scenarios in fog computing," *IEEE Access*, vol. 7, pp. 91 745–91 758, 2019.

[22] Y. Li, A.-C. Orgerie, I. Rodero, B. L. Amersho, M. Parashar, and J.-M. Menaud, "End-to-end energy models for Edge Cloud-based IoT platforms: Application to data stream analysis in IoT," *Future Generation Computer Systems*, vol. 87, pp. 667–678, 2018.

[23] R. Mahmud, S. Pallewatta, M. Goudarzi, and R. Buyya, "iFogSim2: An extended iFogSim simulator for mobility, clustering, and microservice management in edge and fog computing environments," *J. of Systems and Software*, vol. 190, 2022.

[24] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," *Modeling and tools for network simulation*, pp. 15–34, 2010.

[25] "Bilan électrique 2022," https://analysesetdonnees.rte-france.com/bilan-electrique-synthese, 2023.

[26] D. H. Sallo and G. Kecskemeti, "Enriching computing simulators by generating realistic serverless traces," *J. of Cloud Computing*, vol. 12, no. 1, pp. 1–13, 2023.

[27] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner, "United states data center energy usage report," LBNL-1005775, 2016.

[28] M. C. Silva Filho, R. L. Oliveira, C. C. Monteiro, P. R. Inácio, and M. M. Freire, "CloudSim plus: a cloud computing simulation framework pursuing software engineering principles for improved modularity, extensibility and correctness," in *IFIP/IEEE Symp. on Integrated Network and Service Management (IM)*, 2017, pp. 400–406.

[29] A. Tirumala, "Iperf: The TCP/UDP bandwidth measurement tool," *http://dast. nlanr. net/Projects/Iperf/*, 1999.

[30] A. M. Uhrmacher, "Seven pitfalls in modeling and simulation research," in *Winter Simulation Conf. (WSC)*, 2012, pp. 1–12.

[31] A. Varga, "OMNeT++," *Modeling and tools for network simulation*, pp. 35–59, 2010.

[32] P. Velho and A. Legrand, "Accuracy study and improvement of network simulation in the SimGrid framework," in *Int. Conf. on Simulation Tools and Techniques*, 2010.

[33] P. Velho, L. M. Schnorr, H. Casanova, and A. Legrand, "On the validity of flow-level TCP network models for grid and cloud simulations," *ACM Trans. on Modeling and Computer Simulation*, vol. 23, no. 4, 2013.

[34] H. Wu, S. Nabar, and R. Poovendran, "An energy framework for the network simulator 3 (ns-3)," in *Int. Conf. on Simulation Tools and Techniques*, 2011, pp. 222–230.

[35] Y. Zhang, Y. Gan, and C. Delimitrou, "μqsim: Enabling accurate and scalable simulation for interactive microservices," in *IEEE Int. Symp. on Performance Analysis of Systems and Software*, 2019, pp. 212–222.