Comparison of simulators for fog computing*

Christian Kunde and Zoltán Ádám Mann

University of Duisburg-Essen, Essen, Germany

Abstract

Fog computing is gaining popularity as a new distributed computing paradigm. Several simulators have been proposed for the evaluation of new approaches for fog computing. This paper compares four simulators for fog computing: iFogSim, MyiFogSim, EdgeCloudSim, and YAFS. The comparison is based on both publicly available information about the simulators, and on our experience with their practical use. The results show strengths and weaknesses of the simulators, and also some potentially anomalous behaviors.

Keywords: fog computing, edge computing, simulator

1 Introduction

Fog computing was proposed to address the need for low-latency access to compute resources by end devices [2]. Fog computing is based on *fog nodes* deployed near the network edge, in a geographically distributed way. End devices can offload computations to nearby fog nodes. Computations that require higher capacity than what fog nodes offer can be offloaded to the cloud. Fog computing combines end devices, fog nodes, and the cloud to a system in which computations can be distributed dynamically, optimizing important metrics like latency or energy consumption [10].

Several approaches have been proposed for leveraging fog computing, e.g., resource management algorithms or methods for optimally distributing IoT applications [11]. Before applying new approaches in real environments, it is beneficial to test them using simulation. To foster the simulation-based evaluation of fog computing approaches, several fog simulators have been proposed. In this paper, we focus on four promising fog simulators that were created to support fog computing research in general: iFogSim [5], MyiFogSim [7], EdgeCloudSim [13], and YAFS [6]. We use the latest version of the simulators available on May 28, 2019.

In a good fog simulator, it should be easy to simulate different fog computing environments and applications, simulations should run quickly and deliver realistic results in terms of latency, energy consumption etc. Earlier experience with cloud simulators has shown that different simulators tend to realize different trade-offs between the desired properties [1, 9]. The aim of this paper is to compare the four fog simulators and showcase their strengths and weaknesses. We performed (1) a theoretical comparison of the simulators based on information publicly available about them, and (2) a practical comparison by simulating the same scenarios in the simulators. The practical comparison yielded some surprising insights, like different simulators exhibiting different results for the same metric on the same scenario, or counter-intuitive impact of some parameters on the simulation results.

2 Theoretical comparison

Based on comparisons of simulators for related technologies like cloud and IoT [1, 3, 4, 14, 6], we identified three categories of criteria: general properties, technical details, and simulation capabilities.

The results of the comparison according to the *general properties* are shown in Table 1. As can be seen, there are many similarities between the simulators, e.g., the source code of each is available on github. iFogSim is the oldest and YAFS is the youngest among the simulators. iFogSim and MyiFogSim have not been updated for years, whereas EdgeCloudSim and YAFS were updated this year. The amount and type of available documentation is quite different.

*This paper has been published in the *Proceedings of the 35th Annual ACM Symposium on Applied Computing (SAC'20)*, pp. 1792-1795, 2020

Table 1: General properties

	iFogSim	MyiFogSim	EdgeCloudSim	YAFS
availability of source code [3, 1, 4, 12, 8]	yes (github) [4, 12, 3]	yes (github) [12]	yes (github) [12]	yes (github) [12]
initial publication of source code [3]	01.03.2016 [3]	19.11.2017	18.02.2017	27.02.2018
license [3]	not specified, Apache 2 [3]	not specified	GNU General Public License v3.0	MIT License
target audience [4]	academic [4]	academic	academic	academic
last update of source code [3]	21.09.2016 [3]	19.11.2017	14.03.2019	24.05.2019
installation documentation [3]	yes	no	no	yes
comments in source code	15,587 lines	16,049 lines	1,410 lines	7,579 lines
other forms of documentation	no	no	discussion forum (unavailable)	website
binary executable [3]	no [3]	no	no	no

Table 2: Technical details

	iFogSim	MyiFogSim	EdgeCloudSim	YAFS
GUI [3, 1]	yes [3]	yes	no	no
web API [3]	no [3]	no	no	no
configurabi- lity [6, 3]	code [6], JSON	code, JSON	code [6], XML	code, JSON [6]
result formats [3]	XLSX, PDF [3]	XLSX, PDF	CSV	CSV
programming language [3, 1]	Java (100%) [12, 6, 3]	Java (100%) [12]	Java (80.3%), MATLAB (16.3%), shell (2.7%), Limbo (0.7%) [12, 6]	Python (81.5%), R (18%), shell (0.5%) [6]
technology stack [3, 1]	CloudSim [3]	CloudSim	CloudSim	various Python libraries
portability [3]	all JVM supported platforms [12, 3]	all JVM supported platforms [12, 3]	all JVM supported platforms [12, 3]; MATLAB for generating plots	Windows, with manually compiled dependencies; Unix
lines of code [3]	27,467	32,477	11,580	41,704
lines of code, without dependencies	8,397 [3]	13,388	11,580	41,704
headless execution [3]	yes	yes	yes	yes
distributed architecture [3]	no [3]	no	no	no

The second category (see Table 2) encompasses *technical details* of the simulator software that are relevant to both users and developers working with the simulator. (Headless execution means that all configuration is done by command line arguments.) Technically, iFogSim, MyiFogSim and EdgeCloudSim are similar: they are all built on top of CloudSim, hence implemented in Java. YAFS is independent from CloudSim and is built in Python.

The lines of code are counted for the entire repository. YAFS has so many more lines, because its website is fully included in the repository; the python code base is only 14,050 lines of code. “Lines of code without dependencies” means that the source code of the dependencies, like CloudSim, is not counted. In the case of EdgeCloudSim, CloudSim is not included as source code but as a library. When excluding the dependencies and the YAFS website artefacts, the size of the simulators is in a similar order of magnitude.

The comparison of *simulation capabilities* is shown in Table 3. All simulators have a limited network model, which does not conform to a standard like TCP/IP or BRITE. They all feature a federation policy, allowing the coordination of multiple cloud platforms. All simulators are event-based (i.e., simulation is based on events and not on the packets sent over the network). They all support mobile nodes that can change their geographical location. Only YAFS supports device handover, i.e., transitioning work from one node to another, in the case of location changes or capacity exhaustion. In principle, all simulators support some kind of a cost model, energy model, and network model, but there are some important

Table 3: Simulation capabilities

	iFogSim	MyiFogSim	EdgeCloudSim	YAFS
cost model [1]	yes (CloudSim) [1]	yes (CloudSim) [1]	yes (CloudSim) [1], but not implemented by default	yes, but currently commented out
energy model [1]	yes (CloudSim) [1]	yes (CloudSim) [1]	yes (CloudSim) [1], but not implemented by default	yes
network model [8]	limited (CloudSim) [8]	limited (CloudSim) [8]	limited (CloudSim) [8]	limited
network topology [6]	tree [6]	tree	tree [6]	graph [6]
customizable scheduling algorithm [12]	yes [12]	yes [12]	yes [12]	yes
federation policy [1]	yes (CloudSim) [1]	yes (CloudSim) [1]	yes (CloudSim) [1]	yes
type of simulator [8]	event based	event based	event based	event based
mobile nodes [12]	yes [12]	yes[12]	yes [12]	yes
customizable mobility model [12]	not supported [12]	not supported [12]	not supported [12]	not supported
device handover [12]	no [12]	no [12]	no [12]	yes

differences in the details, as will become clear in the next section.

3 Practical comparison

We aim at a more in-depth comparison of the practical use of the simulators. As a shared scenario for the comparison, the "EEG Beam Tractor Game" by Gupta et al. [5], also called VRGameFog, was chosen. A distributed game with real-time interaction requirements, VRGameFog is a typical example of a scenario for fog computing. VRGameFog is already implemented by iFogSim, MyiFogSim and YAFS and only the implementation in EdgeCloudSim is missing.

After analyzing the scenario, we had to conclude that an expedient implementation was not possible in EdgeCloudSim. EdgeCloudSim has a very limited default implementation of a network model, which is not sufficient for a scenario like VRGameFog. A custom implementation of a network model would have unknown ramifications on the simulation results, so that the comparison of EdgeCloudSim to the other simulators would not be fair. This problem is further compounded by the fact that EdgeCloudSim lacks an equivalent to sensors and actuators, which are part of VRGameFog, as well as an implementation of a cost and energy model. Hence, the practical comparison is limited to iFogSim, MyiFogSim and YAFS.

3.1 Implementation

To enable a meaningful comparison, we chose iFogSim's VRGameFog configuration as the reference and adjusted the configuration of the other simulators to match it. Afterwards, we made the necessary modifications to enable the experiments of Sections 3.2–3.3. We also implemented an automated process to run the experiments multiple times and storing the results for each run.

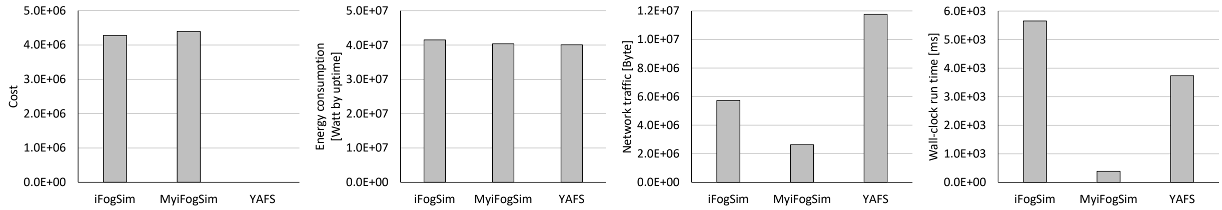
Based on this experience, we can at least partially assess the simulators from a developer perspective. In all the simulators the base scenario was under 300 lines of code, suggesting that the APIs are effective encapsulations of the logic required to set up such a scenario. We found it relatively easy to perform the necessary modifications, and did not encounter major problems.

3.2 Results – base configuration

All experiments are repeated 20 times[1], and the average value is used for the comparisons.

We first compare the results of the simulators in the base configuration. As Figure 1a shows, iFogSim and MyiFogSim yield similar costs. YAFS yields none, since the cost model did not provide data. In

[1] Exception: For iFogSim with more than one application (see Section 3.3), only a single run was measured, since the run time increased drastically.



(a) Cost (b) Energy consumption (c) Network traffic (d) Wall-clock simulation time

Figure 1: Results in base configuration

terms of energy consumption, all three simulators give similar values, as shown in Figure 1b, descending from iFogSim to YAFS.

For the amount of network traffic, shown in Figure 1c, the simulators give significantly differing values: iFogSim about 5.7 MB, MyiFogSim about half of that with 2.6 MB, and YAFS yielding 11.7 MB, roughly double the value of iFogSim. The reasons for these large differences deserve further investigation in future work. In any case, this experience indicates that results of the simulators concerning network traffic should be treated with caution.

Figure 1d shows the wall-clock time of the simulation. Again the three simulators differ strongly, with iFogSim being the slowest (5,654 milliseconds), followed by YAFS (3,735 milliseconds), and MyiFogSim is the fastest (385 milliseconds). In contrast to the case of network traffic, where the significant differences were alarming, the large differences in wall-clock simulation run time are not problematic per se. However, for users running many and/or large simulations, high run time can be a show-stopper.

3.3 Results – modified configurations

We performed four sets of experiments with different modifications:

- Scaling, by running multiple applications in parallel on shared hardware (the same application is deployed in two or three copies) or by changing the number of mobile end devices.
- Modifying hardware capabilities: CPU processing power and available memory of each device are doubled or halved.
- Adjusting the cost and energy models, by doubling or halving the energy consumption and cost of each device.
- Modifying the network configuration, by doubling or halving the available bandwidth and latency of each device.

The results are shown relative to the results of the base configuration, to make the effect of the changes clear. E.g., 1.1 means a 10% increase compared to the result in the base configuration.

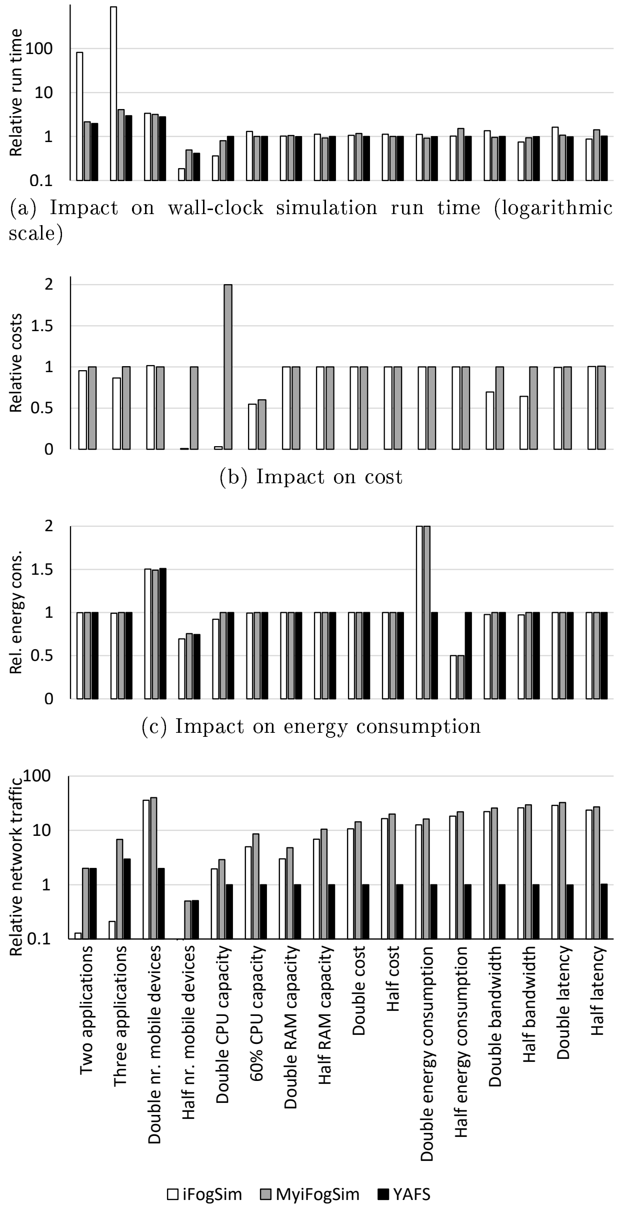
3.3.1 Impact on wall-clock run time (Figure 2a)

In most cases iFogSim is associated with the largest changes in wall-clock simulation run time. The run time of YAFS is largely stable.

YAFS exhibits the best scaling behavior: its run time increase is the smallest both for increasing the number of applications and for increasing the number of mobile devices. For MyiFogSim, the run time increase is higher but still acceptable. For iFogSim, scaling seems to be problematic, especially scaling the number of applications. This may become a critical issue for large-scale simulations.

Changing the devices' CPU processing power leads to plausible changes in the behavior of iFogSim, while YAFS shows no effect at all. Changing the devices' available RAM has no effect, in line with our expectations. For adjusting the devices' cost and energy consumption, the results are mostly in line with our expectations, except for the somewhat surprising increase in wall-clock run time for MyiFogSim with halved energy consumption.

Changing the network links' available bandwidth and latency leads to different results in the simulators. For YAFS, the wall-clock simulation run time did not change. For MyiFogSim, it only changed when the latency was reduced to half, which surprisingly led to an increase in the wall-clock simulation run



(a) Impact on wall-clock simulation run time (logarithmic scale)

(b) Impact on cost

(c) Impact on energy consumption

(d) Impact on network traffic (logarithmic scale)

Figure 2: Impact of different configuration changes

time. iFogSim reacts opposite to the expectations when the available bandwidth changes, and exhibits an unexpectedly large effect to the changes in latency.

3.3.2 Impact on costs (Figure 2b)

As already mentioned, the cost model of YAFS is not available, hence YAFS is excluded. For the scaling experiments, both simulators lead to unexpected results. In MyiFogSim, the costs are constant for all four experiments, whereas in iFogSim an increase of the number of applications actually leads to decreased cost, and for a lower number of mobile devices the costs plummeted extremely.

Changes to the available RAM did not influence the costs, which is plausible. However, doubling the devices' CPU capacity led to an extreme drop in cost for iFogSim, whereas in MyiFogSim costs doubled in that case. For a reduction to 60% CPU power, both simulators yielded decreased costs, contrary to the expectations.

Surprisingly, modifying the devices' cost did not show any effect on overall costs in either simulator.

3.3.3 Impact on energy consumption (Figure 2c)

In several cases, the results are in line with the expectations: e.g., increasing the number of mobile devices in the simulation scenario leads to higher energy consumption consistently in all simulators. However, there are some unexpected results as well. Running more applications does not increase energy consumption in any of the simulators. When the energy consumption of the devices is doubled or halved, this leads to appropriate changes in overall energy consumption in iFogSim and MyiFogSim, but to no change in YAFS.

3.3.4 Impact on network data transfer (Figure 2d)

The results are again counter-intuitive in several cases. For example, running multiple applications in parallel led to lower network traffic in iFogSim than running a single application.

In the experiments with changing device capacities, changing costs and energy consumption, and changing link parameters, the same pattern can be recognized. In YAFS, there is no influence in network traffic, which is plausible. In contrast, iFogSim and MyiFogSim show a considerable increase in network traffic in all these cases – for changes of the parameters in both directions.

4 Conclusions

We conducted a comparison of the fog computing simulators iFogSim, MyiFogSim, EdgeCloudSim, and YAFS based on publicly available information. Moreover, we performed an in-depth comparison of three of the simulators by making different changes to an existing simulation scenario and running the same simulations in each simulator. From a developer perspective, we found it easy to make the intended modifications in each of the three simulators. However, from a user perspective, we made some potentially anomalous findings: the results obtained from different simulators sometimes showed large differences, some changes in the simulation parameters led to counter-intuitive changes in the results.

Our experiments are limited, and so we have to be careful with making far-reaching conclusions. Nevertheless, our results indicate that relying on existing fog simulators may incur risks. As future work, it would be important to validate the simulators by comparing their results with those measured in real systems. Moreover, it would be interesting to investigate in more depth the causes of the counter-intuitive behavior documented in this paper.

Acknowledgment. This work was supported by the European Union's Horizon 2020 research and innovation program under grants 731678 (RestAssured) and 871525 (FogProtect).

References

[1] A. Ahmed and A. S. Sabyasachi. Cloud computing simulators: A detailed survey and future direction. In *IEEE IACC*, pages 866–872, 2014.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the Internet of Things. In *1st Mobile Cloud Computing Workshop*, pages 13–16, 2012.

[3] J. Byrne et al. A review of cloud computing simulation platforms and related environments. In *Proceedings of CLOSER 2017*, pages 651–663, 2017.

[4] J. P. Dias, F. Couto, A. C. R. Paiva, and H. S. Ferreira. A brief overview of existing tools for testing the internet-of-things. In *IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 104–109, 2018.

[5] H. Gupta, A. V. 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*, 47(9):1275–1296, 2017.

[6] I. Lera, C. Guerrero, and C. Juiz. YAFS: A simulator for IoT scenarios in fog computing. *arXiv preprint arXiv:1902.01091*, 2019.

[7] M. M. Lopes, W. A. Higashino, M. A. M. Capretz, and L. F. Bittencourt. MyiFogSim: A simulator for virtual machine migration in fog computing. In *UCC '17 Companion*, pages 47–52, 2017.

[8] R. Malhotra and P. Jain. Study and comparison of CloudSim simulators in the cloud computing. *The SIJ Transactions on Computer Science Engineering & its Applications*, 1(4):111–115, 2013.

[9] Z. Á. Mann. Cloud simulators in the implementation and evaluation of virtual machine placement algorithms. *Software: Practice and Experience*, 48(7):1368–1389, 2018.

[10] Z. Á. Mann. Optimization problems in fog and edge computing. In *Fog and Edge Computing: Principles and Paradigms*, pages 103–121. Wiley, 2019.

[11] Z. Á. Mann, A. Metzger, J. Prade, and R. Seidl. Optimized application deployment in the fog. In *International Conference on Service-Oriented Computing*, pages 283–298. Springer, 2019.

[12] T. Qayyum, A. W. Malik, M. A. K. Khattak, O. Khalid, and S. U. Khan. FogNetSim++: A toolkit for modeling and simulation of distributed fog environment. *IEEE Access*, 6:63570–63583, 2018.

[13] C. Sonmez, A. Ozgovde, and C. Ersoy. EdgeCloudSim: An environment for performance evaluation of edge computing systems. *Transactions on Emerging Telecommunications Technologies*, 29(11):e3493, 2018.

[14] S. Svorobej et al. Simulating fog and edge computing scenarios: An overview and research challenges. *Future Internet*, 11(3):55, 2019.