WQR-UD: An online scheduling algorithm for FemtoClouds

Please, cite this paper as:
Cosimo Anglano, Massimo Canonico, Marco Guazzone
"WQR-UD: An online scheduling algorithm for FemtoClouds,"
In Proc. of the 12th EAI International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS). ACM, New York, NY, USA, 179-182.
DOI:10.1145/3306309.3306338
Publisher: https://doi.org/10.1145/3306309.3306338

WQR-UD: An online scheduling algorithm for FemtoClouds

Cosimo Anglano
University of Piemonte Orientale
Italy
cosimo.anglano@uniupo.it

Massimo Canonico
University of Piemonte Orientale
Italy
massimo.canonico@uniupo.it

Marco Guazzone
University of Piemonte Orientale
Italy
marco.guazzone@uniupo.it

ABSTRACT

FemtoClouds are computing platforms, implementing the Fog Computing paradigm, consisting in an ensemble of heterogeneous mobile devices whose users agree to run the tasks offloaded by other users. FemtoClouds are well suited for the execution of Bag-of-Tasks (BoTs) applications, but, being characterized by high resource heterogeneity and volatility, require the availability of scheduling techniques able to effectively deal with ensembles of independently-owned, heterogeneous devices that can suddenly leave the system. In this paper we propose WQR-UD, an online scheduling algorithm that, thanks to the combination of simple task and device selection policies (that do not require any information concerning the applications and the devices) with effective heterogeneity and volatility tolerance mechanisms, is able to effectively schedule a stream of BoT applications on FemtoCloud systems. We assess the ability of WQR-UD to meet its design goals by running an extensive simulation study for a large set of realistic operational scenarios. Our results clearly indicate that WQR-UD is able to effectively schedule a stream of BoT applications on FemtoCloud systems.

CCS CONCEPTS

• **Networks** → **Cloud computing**; • **Computer systems organization** → **Cloud computing**;

KEYWORDS

FemtoCloud, Scheduling policy, Fog computing, Cloud computing.

ACM Reference Format:
Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2019. WQR-UD: An online scheduling algorithm for FemtoClouds. In *Proceedings of 12th EAI International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS'19).* ACM, New York, NY, USA, 6 pages.

1 INTRODUCTION

Driven by the emerging *Internet of Things* (IoTs) and by the proliferation of mobile devices, the number of connected devices is predicted to reach 50 billion by 2020 and data generated by such devices will be 40 trillion gigabytes [29].

The traditional approach to deal with the huge computing and storage capacity demand, needed to process these data, is to resort to *Cloud Computing*. However, these data often require (near) real-time processing (e.g., for augmented reality services or smart traffic light systems) [4, 7, 11]. Therefore, the inherent high latency of the core network makes Cloud Computing unsuitable to meet these stringent requirements. *Fog Computing* [28] has recently emerged as a new paradigm to mitigate the escalation in resource congestion and to support low-latency services. With Fog Computing, a number of computation nodes distributed across the network "edge" (nearby to the users) can offload the computational requests from the Cloud Computing infrastructure, and can significantly reduce the latency with respect to centralized Clouds.

Recently it has been argued that the processing capability in Fog Computing can be provided by underutilized mobile edge devices (e.g., smartphones and tablets), that are aggregated into a so-called *FemtoCloud*, whose users agree to run the tasks offloaded by other users (this can be achieved by resorting to suitable incentive mechanisms) [17]. The FemtoCloud approach is considered to be very promising, since the computational power provided by mobile devices is becoming more and more pervasive, making mobile devices the most used platform in recent years [9, 13].

The architecture of a typical FemtoCloud system, composed by a set of distinct FemtoClouds, is shown in Figure 1, where we see that each FemtoCloud is coordinated by a *Fog Node*, an always-present/always-on machine which is in charge of receiving offloaded tasks from a population of users, of dispatching them on the devices it coordinates, of receiving the results they generate, and of forwarding these results to the corresponding users. Specifically, the Fog Node runs a *Scheduler* component, whose main goal is to schedule offloaded tasks on available devices so as to minimize their completion time. In the following, without loss of generality, we use the terms "scheduler" and "Fog Node" interchangeably.

To successfully carry out scheduling, the Fog Node has to properly take into account device *heterogeneity* (edge devices are very different both in terms of hardware and software characteristics) and *availability* (an edge device could become unreachable at any moment due either to network connection issues, to device owner mobility, or to battery depletion).

VALUETOOLS'19, March 2019, Palma de Mallorca, Spain

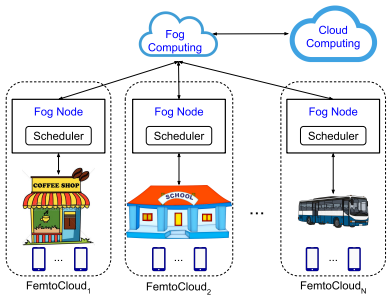

Figure 1: FemtoCloud system architecture.

These problems have already been addressed in the past by resorting to combinations of *replication* (i.e., several instances of the same task are created and scheduled on different devices) and *periodic checkpointing* (i.e., the state of a task is periodically saved, so that its execution can be restarted from the last saved state), first in the context of clusters of non-dedicated PCs [2] and, later, of desktop grids [3]. Hence, a natural question arises about the applicability of these solutions to FemtoCloud systems.

However, the efficacy of the above solutions is somehow limited by the lack of reliable information about resource availability. In fact, while replication does not require any information concerning the characteristics of tasks and of devices, checkpointing requires the choice of a suitable frequency with which state is saved. It is known [12] that the optimal checkpoint frequency can be computed only under specific assumptions about the behavior of machines and applications so, in real cases, it can only be approximated, and this approximation usually results in the creation of many more checkpoints than strictly needed.

In FemtoCloud systems, an excessive checkpointing activity would severely hurt the stability of the system, given that checkpointing is a resource-intensive activity not only in terms of storage space, but also in terms of computing capacity. On edge devices the creation of a checkpoint would indeed not only slow down the device, but also deplete its battery more quickly, hence either forcing the device owner to leave the system or causing its switch off. Therefore, existing scheduling algorithms based on replication and periodic checkpointing would exhibit potentially unsatisfactory performance also on FemtoCloud systems.

However, we observe that – unlike clusters of PCs and desktop grids – in FemtoClouds the computational capacity is narrowed in restricted area such as a public transit, a classroom, a movie theater, a coffee shop and so on, as illustrated in Figure 1. Hence, we can exploit this fact to collect more precise information about when the device will leave the system, and carry out only one checkpoint about when the device is going to leave, thus minimizing the number of checkpoints created.

In this paper we argue that, by exploiting the information about device presence gathered as discussed before, it is possible to build a scheduling algorithm for FemtoCloud systems that – by combining replication and checkpointing – can effectively schedule a stream of *Bag-of-Tasks applications* (BoTs) [20] (parallel applications whose tasks are completely independent from one another) have been shown to be particularly able to exploit the computing power provided in many distributed platforms [21], and, despite their simplicity, are used in a variety of domains (e.g., [14, 23, 24]). Each BoT is characterized by two parameters, namely the number of tasks and the computation requirements of each task.

We prove our claim by presenting a novel online scheduling algorithm for FemtoCloud systems based on the above principle, that we call *Work Queue with Replication and User Driven Checkpoint* (WQR-UD), and we experimentally evaluate its performance by carrying out an extensive simulation study for a variety of real-world operational conditions. The results we obtain clearly show that WQR-UD is able to effectively schedule a stream of BoT applications in all the scenarios we considered.

The rest of the paper is organized as follows: Section 2 summarizes related works, while Section 3 presents the details of our algorithm. In Section 4, we evaluate the performance of WQR-UD and we discuss the results obtained. Finally, Section 5 presents a summary of our findings and a discussion of future directions.

2 RELATED WORK

The literature focusing on computation offloading for mobile devices offers several scheduling algorithms that can be applied also to FemtoCloud systems (e.g., [18, 22, 27]).

However, the above scheduling algorithms require the knowledge of information concerning submitted tasks (i.e., their arrival rate, the amount of work they require, or both) and device characteristics (i.e., their computation capacity). These information, however, may be very difficult to collect reliably in a FemtoCloud system, because of the lack of control on the user population providing devices, and on the possible difficulty in convincing them to install suitable software on their devices. Also, in many cases these algorithms work offline (i.e., they compute a scheduling plan ahead of time before tasks are actually submitted), which, however, implies that precise information about tasks and devices are known in advance.

In contrast, our WQR-UD algorithm works online (i.e., tasks are scheduled as soon as they arrive to the system), and does not require any information about the characteristics of tasks and devices. The only information that it requires is, as already mentioned, the presence time of each device in the system, which in many cases can be easily inferred from the context or can be requested to the user.

3 SCHEDULING ALGORITHM

In this section, we first describe the system model (Section 3.1), and finally describe how WQR-UD works (Section 3.2).

3.1 System model

We consider a set of heterogeneous mobile devices connected to a Fog Node. Each device is characterized by its *nominal computing power* , a real number whose value is directly proportional to its speed , and by the *communication bandwidth* (in bits/sec.) towards the Fog Node.

Each device may join and leave the FemtoCloud at any time, but we assume that the corresponding user declares how long (s)he will stay in that location when (s)he joins. We think this assumption is realistic because (1) the user should be able to predict how long (s)he will stay in the location, and (2) we postulate that the user receives incentives if (s)he communicate its *presence time* (i.e., the time elapsed between the arrival and the departure of the device from the system) to the local Fog Node. Finally, we assume that FemtoCloud applications run encapsulated in virtual machines on a mobile virtualization platform [19, 25], so that their execution state can be easily saved and restored later on any device.

3.2 The WQR-UD scheduling policy

WQR-UD combines very simple task and device selection policies with several mechanisms specifically tailored to cope with device departures from the FemtoCloud. In particular, it is derived from the WQR-FT scheduling algorithm [3], that targets desktop grids, by replacing its checkpointing mechanism as follows.

WQR-UD selects in an arbitrary order the tasks of the oldest BoT (i.e., the BoT who arrived first in the system with respect to the others) and submits them to devices as soon as they become available. No information concerning task or device characteristics is taken into consideration for these selections. In order to tolerate device departures, WQR-UD exploits the following three mechanisms:

(1) *task replication*: replication is used to cope with device heterogeneity and departures. As a matter of fact, by submitting the same task to different devices, we increase the chances of task completion, since to complete the task it suffices that at least one of these devices stays in the system. At the same time, we increase the chance of reducing its execution time, since in general both faster and slower devices will be chosen for its replicas, and the faster device will complete the tasks (at that point, the slower replicas that are still running will be aborted, as they become useless). The maximum number of replicas for each task is a scheduler parameter called *replication threshold*;
(2) *task selection*: anytime a device becomes available (because it has completed the task assigned or it has just joined the FemtoCloud system) WQR-UD selects the oldest task with the lowest number of replicas and submits it to the idle device. If all tasks have already a number of replicas that is equal to the replication threshold, WQR-UD waits for a task to submit (that is, a new task from a new BoT just arrived in the system or a task died due to a device failure). A task remains in the system until it has been completed;
(3) *checkpointing*: before a device leaves the FemtoCloud system as claimed by its owner, a snapshot of the virtual machine running its task t is saved by the Fog Node, so that when the task has been selected by the scheduler, this snapshot will be used to restart task t.

As shown in Section 4, the combination of these three mechanisms with the task and machine selection policies results into an effective scheduling algorithm.

4 PERFORMANCE EVALUATION

In order to assess the ability of WQR-UD to successfully schedule BoT applications on FemtoClouds, we perform an exhaustive simulation study.

To carry out our study, we developed a discrete-event simulator, written in Python and C++, and based on the *Salabim library* [26].

Our study has been carried out by using as metric the *Average BoT Completion Time* ($ABCT$), that is the time elapsed between the submission of a BoT and the termination of all its tasks. This average value has been computed by using the independent replication method [10], where each independent replica corresponds to the time to complete 2000 BoTs and where the whole simulation stops when the relative precision of the 95% confidence interval is $\leq 4\%$.

In order to obtain realistic results, in our simulations we considered a set of realistic scenarios and workloads, obtained from the experimental setup used in [18]. For the *device arrival rate*, we consider 3 different scenarios, called *FewDev*, *MedDev* and *ManyDev*, corresponding to FemtoClouds with a small, moderate and large number of mobile devices, respectively, and we characterize them according to 3 different Exponential probability distributions whose rate parameters are set to 7.5, 15 and 22.5, respectively. Finally, for the *device presence time*, we also consider 3 different scenarios, namely *LowAvail*, *MedAvail* and *HighAvail*, corresponding to the case of low, medium and high presence time, respectively, and we characterize them according to 3 different Normal probability distributions whose mean parameters are set to 10.38, 13.84, 17.30 and whose standard deviation parameters are set to 2.08, 2.77, 3.46, respectively.

By combining the 3 presence time scenarios with the 3 device arrival scenarios, we consider 9 different scenarios, that we denote as "X-Y", where $X \in \{\text{LowAvail}, \text{MedAvail}, \text{HighAvail}\}$ and $Y \in \{\text{FewDev}, \text{MedDev}, \text{ManyDev}\}$.

In all the scenarios, the communication bandwidth between each device and the Fog Node is drawn from a Normal probability distribution with mean set to 30 Mbps and standard deviation set to 6 Mbps.

In Figure 2, we present the results obtained in our study as a bar chart with error bars, where each bar denotes the average BoT completion time ($ABCT$) achieved by WQR-UD in a specific scenario and the associated error bar represents its 95% confidence interval.

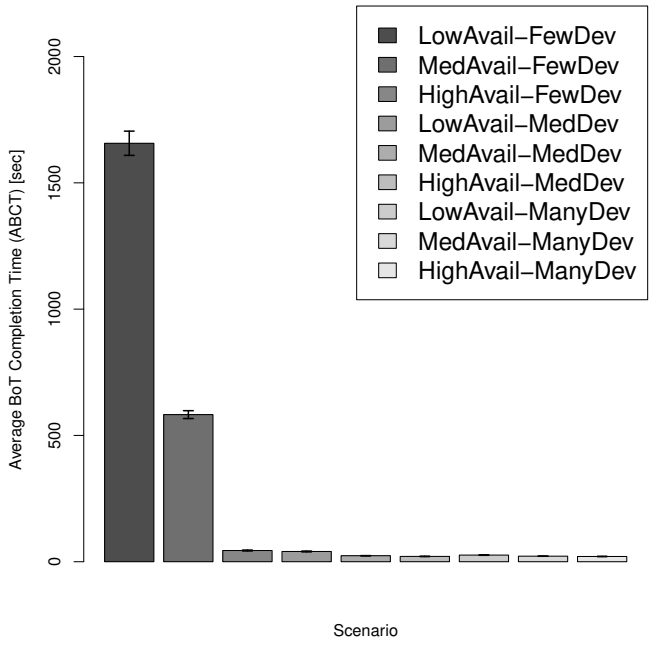

Figure 2: Results for the considered scenarios.

Results reported in this figure show that WQR-UD is able to scale with the number of devices available in the system where to run tasks. Specifically, in scenario "LowAvail-FewDev" (i.e., where, on average, the number of tasks to execute is much larger than the number of available devices where to run them), the $ABCT$ achieved by WQR-UD is very high, that is 1656.70 seconds. However, as soon as each device stays in the system longer (e.g., see scenarios "MedAvail-FewDev" and "HighAvail-FewDev") or the number of devices in the systems increases (e.g., see scenarios "LowAvail-MedDev" and "LowAvail-ManyDev"), the $ABCT$ achieved by WQR-UD decreases dramatically. For instance, in the scenario "MedAvail-FewDev", $ABCT$ is equal to 582.40 seconds, while in the scenario "LowAvail-MedDev", $ABCT$ is equal to 40.58 seconds, and in the scenario "HighAvail-ManyDev", $ABCT$ is equal to 20.80 seconds. These results also show that the impact of the presence time on the performance of WQR-UD decreases as the number of devices in the system increases. Specifically, when the number of devices is very limited (i.e., scenarios "$*$-FewDev"), the presence time greatly affects the performance of WQR-UD (e.g., in scenarios "LowAvail-FewDev" and "HighAvail-FewDev", $ABCT$ drops from 1656.70 to 44.40 seconds); instead, when the number of devices is high (i.e., scenarios "$*$-ManyDev"), the impact of the presence time is very low (e.g., in scenarios "LowAvail-ManyDev" and "HighAvail-ManyDev", $ABCT$ drops from 26.34 to 20.80 seconds). This is a consequence of the use of task replication and checkpointing as well as of the availability of a large number of devices where to run tasks. In fact, in this case, when a device leaves the system, it is likely that WQR-UD finds another free device where to restore the execution of the task that was running on the device that has just left the system.

5 CONCLUSIONS

In this paper, we have considered the problem of scheduling a stream of BoT applications on a FemtoCloud system, composed of an ensemble of heterogeneous mobile devices that join and leave the system anytime without notice. We addressed this problem by proposing an online scheduling algorithm, named WQR-UD, able to effectively schedule a stream of BoT applications on a FemtoCloud system, thanks to the combination of simple task and machine selection policies (that do not require any information concerning the applications and the devices) with mechanisms specifically conceived to tolerate device heterogeneity and volatility.

We assessed the ability of WQR-UD to meet its design goals by performing an extensive simulation study for a large set of realistic operational scenarios. Our results clearly indicate that WQR-UD is able to effectively schedule a stream of BoT applications on FemtoCloud systems.

As future work, we want to compare WQR-UD with existing scheduling alternatives. Furthermore, we plan to study the effects of other scheduling mechanisms, such as the use of a dynamic replication threshold and of task-dependent replication thresholds, on the performance of WQR-UD. We are also interested in implementing a fuzzy controller inside the Fog Node as proposed in [1, 5, 6]. Furthermore, we plan to implement a prototype of WQR-UD using Fog Computing platforms like *OpenStack++* [16], *Edgent* [15], *CirrusCloud* [18], or the *Prometheus* toolkit [8].

ACKNOWLEDGMENTS

This research is original and has a partial financial support of the Università del Piemonte Orientale.

REFERENCES

[1] Luca Albano, Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2013. Fuzzy-Q&E: Achieving QoS Guarantees and Energy Savings for Cloud Applications with Fuzzy Control. In *2013 International Conference on Cloud and Green Computing (CGC'13)*. 159–166.

[2] C. Anglano and M. Botta. 2002. NOW G-Net: learning classification programs on networks of workstations. *IEEE Transactions on Evolutionary Computation* 6, 5 (Oct 2002), 463–480.

[3] Cosimo Anglano and Massimo Canonico. 2005. Fault-tolerant scheduling for bag-of-tasks grid applications. In *European Grid Conference*. Springer, 630–639.

[4] Cosimo Anglano, Massimo Canonico, Paolo Castagno, Marco Guazzone, and Matteo Sereno. 2018. A game-theoretic approach to coalition formation in fog provider federations. In *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC'18)*. 123–130.

[5] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2015. FC2Q: exploiting fuzzy control in server consolidation for cloud applications with SLA constraints. *Concurrency and Computation: Practice and Experience* 27, 17 (2015), 4491–4514.

[6] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2017. FCMS: A fuzzy controller for CPU and memory consolidation under SLA constraints. *Concurrency and Computation: Practice and Experience* 29, 5 (2017).

[7] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2018. Profit-aware Resource Management for Edge Computing Systems.

In *Proc. of the 1st International Workshop on Edge Systems, Analytics and Networking (EdgeSys'18)*. 25–30.

[8] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. 2018. Prometheus: A flexible toolkit for the experimentation with virtualized infrastructures. *Concurrency and Computation: Practice and Experience* 30, 11 (2018).

[9] Cosimo Anglano, Marco Guazzone, and Matteo Sereno. 2014. Maximizing profit in green cellular networks through collaborative games. *Computer Networks* 75, Part A (2014), 260–275. https://doi.org/10.1016/j.comnet.2014.10.003

[10] J. Banks et al. 2010. *Discrete-Event System Simulation* (5th ed.). Prentice Hall.

[11] F. Bonomi et al. 2012. Fog computing and its role in the internet of things. In *Proc. of the MCC'12*. 13–16.

[12] M. Bougeret et al. 2011. Checkpointing strategies for parallel jobs. In *Proc. of SC'11*. 33:1–33:11.

[13] D. Chaffey. 2018. Mobile marketing statistics compilation. Retrieved Oct. 5, 2018 from https://bit.ly/1wXFQtJ

[14] W. Cirne et al. 2003. Grid computing for bag of tasks applications. In *Proc. of the 3rd IFIP Conference on E-Commerce, E-Business and EGovernment*.

[15] The Apache foundation. [n. d.]. Edgent: a community for accelerating analytics at the edge. Retrieved Oct. 5, 2018 from https://edgent.apache.org/

[16] K. Ha et al. 2015. *Openstack++ for cloudlet deployment*. Technical Report CMU-CS-15-123. Carnegie Mellon University. Retrieved Oct. 5, 2018 from https://bit.ly/2NFEZr2

[17] K. Habak et al. 2015. FemtoClouds: Leveraging Mobile Devices to Provide Cloud Service at the Edge. In *Proc. of the CLOUD'15*. 9–16.

[18] K. Habak et al. 2017. Workload management for dynamic mobile device clusters in edge femtoclouds. In *Proc. of the SEC'17*. 6.

[19] D. Jaramillo et al. (Ed.). 2014. *Virtualization Techniques for Mobile Systems*. Springer.

[20] K. H. Kim et al. 2007. Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-enabled Clusters.. In *Proc. of the CCGrid'07*, Vol. 7. 541–548.

[21] D. Kondo et al. 2009. Cost-benefit analysis of cloud computing versus desktop grids.. In *Proc. of the IPDPS'09*, Vol. 9. 1–12.

[22] Z. Lu et al. 2015. Task allocation for mobile cloud computing in heterogeneous wireless networks. In *Proc. of the ICCCN'15*. 1–9.

[23] A. J. V. Neto et al. 2018. Fog-Based Crime-Assistance in Smart IoT Transportation System. *IEEE Access* 6 (2018), 11101–11111.

[24] J. L. Pérez et al. 2018. A resilient and distributed near real-time traffic forecasting application for Fog computing environments. *Future Generat. Comput. Syst.* 87 (2018), 198–212.

[25] J. Shuja et al. 2016. A Survey of Mobile Device Virtualization: Taxonomy and State of the Art. *ACM Comput. Surv.* 49, 1 (April 2016), 1:1–1:36.

[26] R. Van der Ham. 2018. Salabim: discrete event simulation and animation in Python. *J. Open Source Softw.* 3, 27 (2018), 2.

[27] M. Xiao et al. 2015. Multi-task assignment for crowdsensing in mobile social networks. In *Proc. of the INFOCOM'15*. 2227–2235.

[28] W. Yu et al. 2017. A survey on the edge computing for the Internet of Things. *IEEE Access* 6 (2017), 6900–6919.

[29] N. Zhang et al. 2018. Synergy of big data and 5g wireless networks: opportunities, approaches, and challenges. *IEEE Wireless Commun.* 25, 1 (2018), 12–18.