The Fog Balancing: Load Distribution for Small Cell Cloud Computing

Jessica Oueis[1], Emilio Calvanese Strinati[1], and Sergio Barbarossa[2]

[1]CEA, LETI, Grenoble, France
[2]Sapienza University of Rome, Rome, Italy

***Abstract*—In 5G future wireless networks, the (ultra)-dense deployment of radio access points is a key drive for satisfying the increase of traffic demand and improving perceived users' quality. (Ultra)-dense deployment combined with capillary edge cloud, the *fog*, leads the way for optimization of users' Quality of Experience (QoE) and network performance. In this paper, we focus on improving users' QoE by addressing the issue of load balancing in *fog* computing. In this paper, we consider the challenging case of multiple users requiring computation offloading, where all requests should be processed by local computation clusters resources. We propose a low complexity small cell clusters establishment and resources management customizable algorithm for *fog* clustering. Our simulation results show that the proposed algorithm yields high users' satisfaction percentage of a minimum of 90% for up to 4 users per small cell, moderate power consumption, and/or high latency gain.**

I. Introduction

Computation offloading to cloud servers is becoming a powerful tool for improving mobile terminals performance. It allows mobile users to have access to greater computational resources and storage capacity. With computation offloading, mobile terminals are alleviated from the burden of performing complex and power consuming computations. Several frameworks and methods that enable and allow computation offloading have been presented over the last years [1]-[5]. Moreover, mobile applications are growing in number and variety, but also in complexity and time criticality. Mobile users have as well the ability to launch more than one application at a time, each with different latency constraint. The latency of accessing the cloud servers through a wide area network (WAN) cannot be controlled, which may result in service delivery failure. A new approach is to offload computations to pools of resources that are closer to the user [6]. The European project TROPIC [7] proposed a new architecture in which small cell (SC) access points are characterized by computation and storage capacities [8]. This is an example of *fog* computing architecture where cloud computing is extended to the edge of the network [9]. Small Cell Cloud (SCC) provides the ability of using pooled resources at a much closer distance than the traditional cloud. Small cells cooperate for computation purposes by forming computation clusters. However, effective small cells cluster assisted cooperation is subject to momentary and local resources availability, base stations deployment scenarios, offloaded applications delay constraints, and power consumption budgets. These parameters affect the choice of the cluster size and the computation load distribution. One of the main challenges in this framework is the joint distribution of computational and radio resources for mobile terminals and the local cloud. This distribution should respect all sorts of imposed constraints such as latency limitations and power budgets. In [10], a computation offloading distribution between local computation (on the mobile terminal) and a SCC server has been derived for the case of a single user latency constrained request and a single cloud with known computational capacity. The case of multiple users asking for computation offloading to a single SC has been studied in [11]. In both papers, the problem has been cast as a convex optimization problem and a solution has been derived. However, the computations are considered to be offloaded to a single cloud entity with known computational capacity. Nevertheless, SCs cannot offer the same computation and storage capacities as traditional servers used for cloud computing. Computational resources of only one SC is not always enough, particularly when a large number of requests is received. In the SCC platform, SCs can cooperate together through cluster formation. The SC clustering will provide mobile users with near server farms of cloud services that have significant computational capacities. In [12], a multi-user, multi-cell, multi-cloud scenario has been considered. In this work, cloud SC clusters are considered to be pre-established. The paper deals with associating each mobile user to a small cell and a SCC. The cloud is considered as a whole entity, at an architectural layer different than the SC itself. The authors propose a joint optimization of computational and radio resources that minimizes the power consumption per user while respecting all the latency constraints imposed by each user. The problem has been cast as a non convex optimization problem. A convexification of the objective function and the constraints is derived in order to solve the original problem.

A first novel aspect of our approach is the changing composition of computation clusters depending on the requested task and the system parameters. In fact, we do not consider the SCC as a pre-established entity, but as a cluster that is dynamically built. When the user sends the computation offloading request to its serving SC, the latter will have the responsibility to satisfy this request either by processing all of the tasks, or by taking part of a dynamically formed SCC. In [13], the authors proposed various strategies of cluster formation that could be adopted for the single user case. These strategies differ in their objective. A first strategy has the objective of minimizing the experienced SCC latency, and therefore it has high power consumption. A second strategy has been proposed to exploit the latency - power consumption trade-off and remove some power costly small cells from the clusters. Two others strategies with power consumption minimization objectives from the system and the small cell point of views have also been proposed.

978-1-4799-8088-8/15/$31.00 ©2015 IEEE

In a multi-user case, several serving SCs form computation clusters using the same pool of resources. Computation clusters provisioning and resources allocation should be jointly and simultaneously optimized for all users for maximal, nay optimal performance.

We formulate the clustering problem for multiple users as a joint optimization of radio and computational resources with the objective of minimizing the overall communication power consumption in the SCC. The optimization problem is a combinatorial NP-hard problem with a number of variables that increases with the number of users and small cells. Finding the optimal solution, if possible, is penalizing in the mobile computation offloading context. Furthermore, exhaustive search is impractical due to the high number of related variables. Thus, we designed a heuristic method in order to cope with the intractability of the problem. The proposed approach joins both tasks scheduling and cluster formation. In the European project TROPIC, three resources allocation algorithms for SCC clustering are proposed [14]. The first algorithm, *'Path'*, is based on transmission quality between UEs (Users' Equipments) and SCs. The SCs with the best 'path' quality are selected for participating in the computation process. The *'Comp'* algorithm is based on the computational power available at each SC. It is the SCs with the highest computational power at the estimated data delivery time that participates in the computation process. In addition, a combination of both algorithms is proposed under the name of *'ACA'* algorithm. By estimating the number of instructions to be computed (according to the application type), an overall delay is computed taking into account both path quality and available computational power. SCs with lowest overall delay are selected for participating in the computation process. The tasks are treated in all of these algorithms in a FIFO (First In First Out) manner. Treating tasks as FIFO may not always be the best scheduling solution especially in the case where tasks characteristics vary in latency constraints and computation load. Furthermore, the proposed algorithms are applied in a scenario where the number of participating SC is predefined. At the contrary, in the approach we propose, the SCC can include as much SCs as needed (in the limits of SC availability) to compute the tasks. Hence, the second novel aspect of our proposal.

In this work, our main concern is to be able to allocate radio and computational resources in order to satisfy as much user requests as possible while keeping low both the SCC power consumption and the process complexity. We propose to split the resources allocation process into two major steps. In a first step resources are allocated at serving SCs following a specific metric based scheduling rule. In a second step, computation clusters are optimally built for unsatisfied requests, following as well a scheduling rule and an optimization objective.

II. System Model

We consider a mobile cloud computing environment where all SCs are endowed with computational and storage capacities. UEs send offloading requests to their serving SC. SCs are connected together via wireless backhaul. They communicate through a point-to-point wireless link that implements an OFDMA air interface with an uplink bandwidth B. In this environment, we consider a set of users $\mathcal{K} = \{1,...,K\}$ where K is the total number of mobile users, and a set of active small cells $\mathcal{N} = \{1,...,N\}$ where N is the total number of active small cells. Each mobile user $k \in \mathcal{K}$ is associated with a single small cell $n \in \mathcal{N}$ to which offloading requests are sent. The set $\mathcal{S} = \{1,...,S\}$ indicates the set of serving small cells such that $S \leq K$ and $\mathcal{S} \subset \mathcal{N}$. The subset $\mathcal{K}_s \subset \mathcal{K}$ represents the set of mobile users that are associated with small cell s. The offloading request (W_k, Δ_k) sent by user $k \in \mathcal{K}$ consists of asking for the computation of W_k instructions in a maximum time of Δ_k seconds. Serving small cells ($\mathcal{S}$) have the possibility to form a computation cluster for computing the received users requests. In this case, the instructions are distributed among the SCs of the cluster. We denote by W_{kn} the instructions block of user k handed for computation at base station n. Each SC $n \in \mathcal{N}$ has a computational capacity of F_n instructions/second. Therefore, the computation time for processing user's k instructions at SC n is equal to $\frac{W_{kn}}{f_{kn}}$, where f_{kn} is the computational capacity allocated for computing user's k instructions at SC n.

We assume that the number of bits to be transmitted to each small cell through uplink and downlink communication is proportional to the instruction block size W_{kn}: $N_{UL}^{kn} = W_{kn}\theta_{UL}$ for uplink, and $N_{DL}^{kn} = W_{kn}\theta_{DL}$ for downlink. θ_{UL} and θ_{DL} are constants that account respectively for the overhead due to the uplink and downlink communications and for the ratio between output and input bits associated to the execution of the instruction block at the SC [10]. The information rate that can be achieved through the wireless channel link between small cells, taking into account packet retransmission is:

$$R_{sn} = B_{sn} log(1 + \frac{\sigma_c |h_{sn}|^2 P_{sn}}{(1 - PER)\Gamma d^{\beta} N_0}) \quad (1)$$

where σ_c is the shadow fading coefficient of the adopted Rayleigh channel model. The channel fading is assumed constant for a whole transmission period. We assume perfect estimation of the coefficients h_{sn} of the channel between small cells $s \in \mathcal{S}$ and $n \in \mathcal{N}$. PER is the target packet error rate, Γ indicates the SNR margin to guarantee a minimum bit error rate BER ($\Gamma(BER) = -\frac{2log(5BER)}{3}$), d represents the distance between s and n, β indicates the path loss exponent which depends, in an indoor environment, on the number of walls separating the two communicating SCs [15], N_0 is the noise power, and P_{sn} the transmission power between small cells s and n.

III. Optimal multi-user small cell clustering

We consider that UEs decide between computing the tasks on the mobile device and computation offloading by comparing the power consumption of both strategies. Whenever an offloading decision is made, the serving SCs receive the requests each from its connected UEs. For optimal performance, resources should be allocated for all users jointly. In this section, we present the clustering problem with an objective of minimizing the overall SCC communication power consumption.

In order to optimally allocate both communication and computation resources for offloading, they should be jointly optimized. In fact, with every cluster formation, computation load should be distributed among the participating small cells: each small cell will be assigned $W_{kn} \leq W_k$ instructions to

compute. In addition, computational resources f_{kn} must be allocated at each of these small cells. On top of that, transmission power for sending the necessary data from the serving small cell to each of the SCC small cells should be adapted. With the objective of minimizing the overall communication power consumption in all the formed clusters, we cast the optimization problem as follows:

$$\begin{aligned} \min_{P^k_{sn}, W_{kn}, f_{kn}} \quad & \sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}_s}\sum_{n\in\mathcal{N}} P^k_{sn} \\ \text{s.t.} \quad & \text{(a) } W_{kn} \geq 0, \forall n \in \mathcal{N}; \\ & \text{(b) } \sum_{n\in\mathcal{N}} W_{kn} = W_k, \forall k \in \mathcal{K}; \\ & \text{(c) } 0 \leq P^k_{sn} \leq P_{max}, \forall s \in \mathcal{S}, \forall k \in \mathcal{K}_s, \forall n \in \mathcal{N}; \\ & \text{(d) } \sum_{k\in\mathcal{K}} f_{kn} \leq F_n, \forall n \in \mathcal{N}; \\ & \text{(e) } \frac{W_{kn}}{f_{kn}} + \frac{W_{kn}(\theta_{UL}+\theta_{DL})}{R^k_{sn}} \leq \Delta_k, \forall n \in \mathcal{N}, \forall k \in \mathcal{K}. \end{aligned} \tag{2}$$

Conditions (a) and (b) in problem (2) guarantee that for each user k, all instructions are computed. Condition (c) states that the communication power spent for sending necessary data for cluster small cells for each user request must be lower than a power budget P_{max}. Condition (d) states that the sum of computational capacities allocated at each SC n could not exceed its total computational capacity F_n. Condition (e) guarantees that the experienced latency at small cell n for computing instructions for user k must not exceed the latency constraint Δ_k imposed by the user request. Problem (2) is a combinatorial optimization problem, with a number of variables to optimize that increases quickly with the number of mobile users K and the number of active SCs N. Solving such a problem in an environment where latency is an important issue, could be penalizing and could result in preventing the system from delivering good quality service for mobile users.

IV. Algorithm Design

Finding the optimal *fog* clusters and adequate resources allocation in the multi-user case is time and power consuming. This is not suitable for mobile computation offloading environments. We propose a five steps clusters establishment and their resources allocation process that can be split into two major phases.

[Step 1] *Local computational resources allocation*
First, local computational resources at each serving SC are allocated for domestic users.

[Step 1.a] Each serving SC sorts tasks (users offloading requests) waiting to be processed according to a specified metric such as latency constraint, computation load, minimum required computational capacity, minimum required energy efficiency, arrival time, etc. In a way, this ordering also defines the scheduling rule to be adopted for local resources allocation (EDF, PF, FIFO, ...). This step gives different priorities to users requests depending on the sorting metric.

[Step 1.b] Serving small cell computational resources are allocated to users requests following the ordered list established in step 1.a. The resources allocation policy follows a certain objective. For example, the objective of local resources allocation can be increasing latency gain, or increasing resources availability time.

[Step 2] *SCC establishment*
User requests that could not be covered by the first step due to resources scarcity at small cells, will be computed by a SCC built to comply with their specific requirements.

[Step 2.a] Feedback on offloading requests and remaining computational capacities remaining at each serving small cell are reported to the small cell manager. This step does not introduce additional overhead comparing to centralized solutions proposed in literature and discussed in section I. In fact, all centralized algorithms and solutions suppose the presence of an entity called small cell manager (SCM) that receives and stores system parameters and requests requirements. This step requires no parameter definition, and therefore won't be defined for the algorithm realizations.

[Step 2.b] Unserved requests of all SCs are classified at the SCM according to a specific metric that can be different from the metric of step 1.a.

[Step 2.c] Computation cluster are built for each of the unserved requests following the order established in step 2.b. The cluster formation can be done with different objectives as presented in [13]. In this step, optimal cluster is computed for each user independently of the others. This reduces the complexity of the optimization problem since the number of variables is extremely reduced. It is the smart choice of the metric and the scheduling in step 2.a in accordance with the optimization objective in step 2.b that helps the system to satisfy a higher number of users. After each cluster formation, SCs with no computational resources left will not take part in the next cluster and therefore will be eliminated from the following optimizations until resources are restored.

These steps constitute a customizable algorithm for small cell clustering for a multi-user scenario. Several versions can be formed using this algorithm by varying the metrics, scheduling rules, and optimization objectives. We propose three algorithms realizations whose performance will be evaluated and compared in section V. These algorithms variants are the following:

EDF-PC:
[1.a] requests are sorted in ascending order of latency constraints. This choice of sorting metric imposes an EDF (Earliest Deadline First) scheduling. Priority is given to tasks with lowest latency margin. FIFO, used for testing algorithms in [14], was considered as a bad candidate for the specific case of our testing scenario because of tight latency constraint and the simultaneity of computation requests, which is not the case in [14].

[1.b] local resources are allocated following a policy that blocks the minimal required computational capacity for each request. This capacity can be expressed as the ratio between the computation load and the latency constraint ($\frac{W_k}{\Delta_k}$).

[2.b] unserved requests are sorted in ascending order of available latency.

[2.c] clusters are formed for users with the objective of minimizing overall clusters communication power consumptions. The problem formulation used for this clustering strategy is cast for user k served by SC s as follows:

$$\begin{aligned}
\min_{P^k_{sn},W_{kn}} \quad & \sum_{n=1,n\neq s}^{N} P^k_{sn} \\
\text{s.t.} \quad & \frac{W_{kn}}{f_{kn}} + \frac{W_{kn}(\theta_{UL}+\theta_{DL})}{log(1+a_{sn}P_{kn})} \leq \Delta_k, \quad \forall k \in \mathcal{K}, n \in \mathcal{N}\setminus\{s\}; \\
& W_{kn} \geq 0, \quad \forall k \in \mathcal{K}, n \in \mathcal{N}\setminus\{s\}; \\
& \sum_{n=1,n\neq s}^{N} W_{kn} = W_k - W_{ks}, \quad \forall k \in \mathcal{K}; \\
& 0 \leq P^k_{sn} \leq P_{max}, \quad \forall n \in \mathcal{N}\setminus\{s\};
\end{aligned} \tag{3}$$

where $a_{sn} = \frac{\sigma_c |h_{sn}|^2}{(1-PER)\Gamma d^{\beta} N_0}$.

EDF-LAT:
This implementation has the same steps of EDF-PC except for step 2.c where clusters are formed with the objective of minimizing overall cluster latency. The latency minimizing clustering problem is cast for user k served by SC s as follows:

$$\begin{aligned}
\min_{W_{kn}} \quad & \max_{n\in\mathcal{N}\setminus\{s\}} \{\frac{W_{kn}}{f_{kn}} + \frac{W_{kn}(\theta_{UL}+\theta_{DL})}{log(1+a_{sn}P_{max})}\} \\
\text{s.t.} \quad & W_{kn} \geq 0, \quad \forall k \in \mathcal{K}, n \in \mathcal{N}\setminus\{s\}; \\
& \sum_{n=1,n\neq s}^{N} W_{kn} = W_k - W_{ks}, \quad \forall k \in \mathcal{K}; \\
& \frac{W_{kn}}{f_{kn}} + \frac{W_{kn}(\theta_{UL}+\theta_{DL})}{log(1+a_{sn}P_{max})} \leq \Delta_k, \quad \forall k \in \mathcal{K}, n \in \mathcal{N}\setminus\{s\}.
\end{aligned} \tag{4}$$

CS-LAT:
[1.a]: requests are sorted in ascending order of requested computations size.
[1.b] local resources are allocated following a policy that blocks the maximal computational capacity available for each request. This will help liberating local resources rapidly and achieving high latency gain.
[2.a] unserved requests are sorted in ascending order of latency tolerance.
[2.c] clusters are formed for users with the objective of minimizing overall clusters latencies (Equation 4).

V. Numerical Evaluation

In this section, we evaluate the proposed strategy of small cell contribution in computation clusters. We consider the case of femtocell deployment for urban scenarios model of the 3GPP framework [15]. This scenario is represented by a single floor building of a 25 apartments grid. In each of these apartments a femtocell is deployed. Parameter ρ_a determines the ratio of active femtocells in the grid. Parameter ρ_s determines the ratio of active femtocells that are connected to mobile users (serving femtocells). We adopt the same system model described in Section II with parameters values listed in Table I.

TABLE I: Simulation parameters values

Parameter	value	Parameter	value
ρ_a	0.5	ρ_s	0.32
B	20MHz	σ_c	10
N_0	-118.4 [dB/Hz]	BER	10^{-6}
W	$[2.10^6; 10.10^6]$	Δ_{app}	[0.5;3.5]
F_n	$[10.10^6; 15.10^6]$	P_{max}	1 [W]
θ_{DL}	0.2	θ_{UL}	1

We compare these algorithms to the case where all requests are handled by the serving SC ('No Clustering'), and to the case where a static clustering rule of equal load distribution between active neighbor SCs is imposed ('Static Clustering').

Figure 1 shows the users' satisfaction ratio for the compared algorithms depending on the maximal number of users simultaneously asking for computation offloading to the same small cell. With the rising number of users, i.e. the number of incoming requests, pooled resources at the small cells are to be shared by more actors. Thus, satisfying all users becomes harder and, in some cases, impossible. In the case where each small cell computes the requests it receives, the users satisfaction ratio drops quickly with the increasing number of users as seen in Figure 1. What is interesting in this plot, is the low performance achieved by the static clustering strategy where computation load is equally distributed on neighbor small cells. This shows that clustering can be a bad solution if it is not adaptively orchestrated. All of the three variants of the proposed method show important gain in satisfaction ratio even for a high number of users per small cell. We can see that EDF-PC and EDF-LAT can maintain over 95% of satisfaction ratio for up to 4 users per femto cell. In fact, a femto cell cannot be associated to a large number of users. Classical usage of femto-cells is about an average of 4 mobile users [16]. Since both of these algorithms schedule the users based on latency urgency, they manage to serve a larger portion of users compared to the CS-LAT that tries to achieve larger latency gain. Nevertheless, CS-LAT manages to keep a satisfaction ratio of at least 90% for less than 4 users per femto cell.

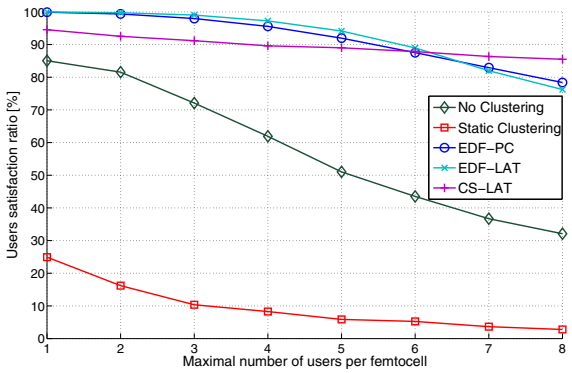

Fig. 1: Users satisfaction ratio in dependence on number of users per small cell

In Figure 2 we show the average latency gain per user. The latency gain for each user is defined as the ratio between the experienced latency for computing the request and the imposed latency constraint. As shown in the plot, the CS-LAT algorithm achieves the highest average latency gain. This due to both local and cluster resources allocation. In fact, in this algorithm, the local serving femtocell allocates its full computational capacity to compute the requests, which results in completing the task with the lowest possible latency. Furthermore, clusters are formed for unserved requests with the objective of minimizing the overall cluster latency. On the other hand, the EDF-LAT latency gain comes only from the latency minimizing clustering strategy. As a matter of fact, serving femtocell local resources allocation policy exploits all the available time for each computation request by allocating the minimal required computational capacity. The more users

are connected to the femtocells, the fewer requests are able to be computed locally on the serving femtocell. Therefore, higher latency gain can be achieved with this algorithm with the increasing number of users since more requests are handled to latency minimizing clusters, as can be seen in the plot of Figure 2.

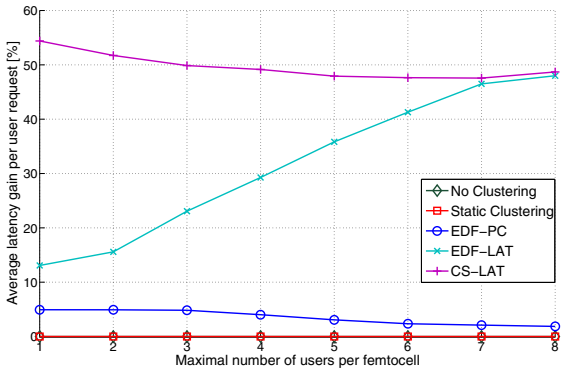

Fig. 2: Average latency gain in dependence on number of users per small cell

The high increase in latency gain in algorithms that adopt a latency minimizing clustering policy comes at the cost of increasing the communication power consumption in the cluster. This is due to the basic wireless communication trade-off between power consumption and latency. Figure 3 shows the average power consumption per user request for the compared algorithms. For the case of No clustering, no data is sent between small cells and therefore there is no communication power cost. An interesting result is the very low communication power consumption for the EDF-PC algorithm even for high number of users per femtocell. This shows the convenience in the choice of both *step 1* and *step 2* metrics and rules. This algorithm can in fact achieve high energy efficiency while keeping a very high quality of service. It is a very good solution to implement whenever latency minimization is not an issue. In fact, when traffic is elastic, optimization could focus on respecting the elasticity limits instead of minimizing latency. Even though EDF-LAT and CS-LAT implement both latency minimizing clusters which implies high power consumption, it is clearly seen that CS-LAT is less power consuming. In fact, since CS-LAT schedules requests in *step 1* based on their computation size instead of adopting and EDF rule, it can serve more users request locally communication cost free. This comes at the cost of lower users satisfaction ratio as can be seen in Figure 1 since users with high requirements of computational capacity (computation size and latency ratio) can be found dropped using such strategy.

VI. Conclusion

In this paper, we proposed an innovative algorithm designed for cluster formation and load balancing for *fog* computing. This algorithm has two advantages. First, it has a customizable design where metrics, scheduling rules, and clustering objectives can be set according to specific applications and network requirements. Second, it consists on a reduced complexity, multi-parameters optimization method. We propose a low complexity multi-fold optimization that guarantees high perceived

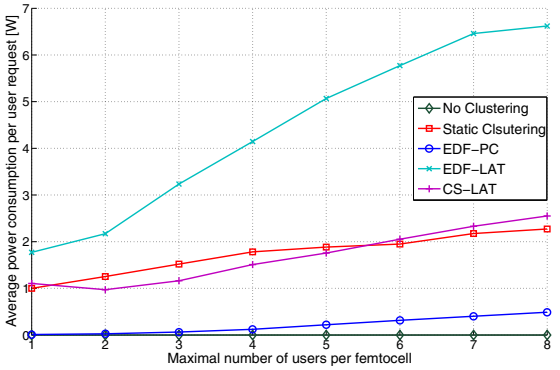

Fig. 3: Average power consumption per user in dependence on number of users per small cell

users' quality. In a first step, computational local resources at the serving small cell are allocated. Then, clusters are established for *fog* computing, for each user request, following a specific optimization objective. Our numerical results evaluate the effectiveness of the proposed method compared to static clustering. Furthermore, for comparison purpose, we consider the representative case where no clustering is possible due to network configuration and applications requirements. Three different variants of the proposed algorithms are tested, EDF-PC, EDF-LAT, and CS-LAT. These implementations differ by the choice of scheduling metrics and rules, and cluster optimization objective. All of these algorithms outperformed static clustering and no clustering scenarios in terms of users' satisfaction ratio. Furthermore, both latency centric optimization algorithms, EDF-LAT and CS-LAT, are proved to achieve significant latency gain compared to other evaluated algorithms. And in particular, the EDF-PC algorithm proved to be an effective solution to adopt for power efficient small cell clustering, due its low average power consumption per user, and its very high users' satisfaction ratio of a minimum of 95% for up to 4 users per femtocell and for a minimum of 80% for extended usage up to 8 users per femtocell.

References

[1] Kumar, K.; Yung-Hsiang Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?," Computer , vol.43, no.4, pp.51,56, April 2010.

[2] Lagerspetz, E.; Tarkoma, S., "Mobile search and the cloud: The benefits of offloading," Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on, vol., no., pp.117,122, 21-25 March 2011.

[3] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload", in Proceedings of the 8th international conference on Mobile systems, applications, and services (MobiSys '10). ACM, New York, NY, USA, 49-62.

[4] B. G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in Proceedings of the sixth conference on Computer systems (EuroSys '11). ACM, New York, NY, USA, 301-314.

[5] Kosta, S.; Aucinas, A.; Pan Hui; Mortier, R.; Xinwen Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," INFOCOM, 2012 Proceedings IEEE , vol., no., pp.945,953, 25-30 March 2012.

[6] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, "The case for VM-based cloudlets in mobile computing," IEEE Pervasive Computing, pp. 14-23, Oct.-Dec. 2009.

[7] FP7 project TROPIC: "Distributed computing storage and radio resource allocation over cooperative femtocells," ICT-318784, www.ict-tropic.eu.

[8] Lobillo, F.; Becvar, Z.; Puente, M.A.; Mach, P.; Lo Presti, F.; Gambetti, F.; Goldhamer, M.; Vidal, J.; Widiawan, A.K.; Calvanesse, E., "An architecture for mobile computation offloading on cloud-enabled LTE small cells," Wireless Communications and Networking Conference Workshops (WCNCW), 2014 IEEE , vol., no., pp.1,6, 6-9 April 2014.

[9] Bonomi, F; Milito, R; Zhu, J; and Addepalli, S., "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, New York, NY, USA, 13-16.

[10] Munoz-Medina, O.; Pascual-Iserte, A.; Vidal, J., "Optimization of Radio and Computational Resources for Energy Efficiency in Latency-Constrained Application Offloading," Vehicular Technology, IEEE Transactions on , vol.PP, no.99, pp.1,1.

[11] Barbarossa, S.; Sardellitti, S.; Di Lorenzo, P., "Joint allocation of computation and communication resources in multiuser mobile cloud computing," Signal Processing Advances in Wireless Communications (SPAWC), 2013 IEEE 14th Workshop on , vol., no., pp.26,30, 16-19 June 2013.

[12] Sardelliti, S.: Barbarossa, S.; Scutari, G., "Distributed Mobile Cloud Computing: Joint Optimization of Radio and Computational Resources," Global Telecommunications Conference, 2014. Globecom'14, December 2014.

[13] Oueis, J.; Calvanese Strinati, E.; Barbarossa, S., "Small Cell Clustering for Efficient Distributed Cloud Computing," Personal Indoor and Mobile Radio Communications (PIMRC), 2014 IEEE 25th International Symposium on, 9-12 Sept. 2012.

[14] Puente, M.A.; Lobillo, F.; Vondra, M.;Dolezal, J; Mach, P; Becvar, Z., LoPresti, F.; Wibowo, A; (2014, June). TROPIC Deliverables. Retrieved from ICT- TROPIC: http://www.ict-tropic.eu/documents/deliverables/TROPIC_D52ATOSf.pdf

[15] 3GPP TSG-RAN4#51, Alcatel-Lucent, picoChip Designs, and Vodafone, "R4-092042, Simulation assumptions and parameters for FDD HENB RF requirements," May 2009.

[16] Chandrasekhar, V.; Andrews, J.G.; Gatherer, Alan, "Femtocell networks: a survey," Communications Magazine, IEEE , vol.46, no.9, pp.59,67, September 2008.