Human-Enabled Edge Computing: Exploiting the Crowd as a Dynamic Extension of Mobile Edge Computing

Paolo Bellavista, Stefano Chessa, Luca Foschini, Leo Gioia, and Michele Girolami

ABSTRACT

The MEC vision leverages the availability of powerful and low-cost middleboxes, statically deployed at suitable edges of the network and acting as local proxies for the centralized cloud backbone; this potentially enables, among other things, better scalability and better reactivity in the interaction with mobile nodes via local control decisions and actuation. MEC has already been proposed as an enabler for several Internet of Things and cyber-physical systems application scenarios, and also mutual benefits due to the integration of MEC and mobile crowdsensing (MCS). The article originally proposes human-driven edge computing (HEC) as a new model to ease the provisioning and to extend the coverage of traditional MEC solutions. From a methodological perspective, we show how it is possible to exploit MCS i) to support the effective deployment of fixed MEC (FMEC) proxies and ii) to further extend their coverage through the introduction of impromptu and human-enabled mobile MEC (M^{2}EC) proxies. In addition, we describe how we have implemented these novel concepts in the MCS ParticipAct platform through the integration of the MEC Elijah platform in the ParticipAct living lab, an ongoing MCS real-world experiment that involved about 170 students at the University of Bologna for more than two years. Reported experimental results quantitatively show the effectiveness of the proposed techniques in elastically scaling the load at edge nodes according to run-time provisioning needs.

INTRODUCTION

Mobile edge computing (MEC) is a recent architectural model and specification proposal by the European Telecommunications Standards Institute (ETSI) that aims to evolve the traditional two-layer cloud-device integration model, where mobile nodes directly communicate with a central cloud through the Internet, with the introduction of a third intermediate middleware layer that executes at so-called network edges. This promotes a new three-layer device-edge-cloud hierarchical architecture, which is recognized as very promising for several application domains [1]. In fact, the new MEC model allows moving and hosting computing/storage resources at network edges close to the targeted mobile devices [1].

In MEC solutions, edges are commonly deployed statically (by telco providers) and logically coupled with radio access network (RAN) base stations (i.e., being deployed close to either the eNodeB or the serving gateway); we claim that this model still faces some weaknesses. First, the number of edges is generally limited, since edges introduce additional costs of operation for the supported services, such as deployment and maintenance. Second, edges are rarely are re-deployed, which might be highly inefficient. Finally, some geographical areas might be highly populated and interesting for a service only during specific time slots (daily, weekly, or yearly patterns).

At the same time, the worldwide use of smartphones has recently boosted mobile crowdsensing (MCS) as the possibility to leverage people roaming though the city with their sensor-rich devices as intelligent sensing probes. In fact, by installing an MCS application, any smartphone can become part of a large-scale sensor network. However, for some highly demanding MCS services (e.g., a surveillance application monitoring an environment with smartphone cameras and implementing face recognition to trace suspicious users' movements), regular smartphones often do not have enough capabilities to perform the requested local tasks in a timely manner (e.g., in hostile environments with disconnections from the cloud).

In other words, we claim that there are several practical cases of large and growing relevance where the joint exploitation of MEC and MCS solutions would bring highly significant benefits in terms of efficient resource usage and perceived service quality. To the best of our knowledge, only a very limited number of seminal works have explored the mutual advantages in the joint use of these two classes of solutions, and they are mostly focused on pure technical communication aspects without considering the crucial importance of having humans as central contributors in the loop [2–4].

This article aims to fill that gap and proposes human-driven edge computing (HEC) as a new model to ease the provisioning and deployment of MEC platforms as well as to enable more powerful MEC-enabled MCS applications by contributing several core elements of technical novelty. First and foremost, *HEC eases the planning and deployment of the basic MEC model*: it mitigates the potential weaknesses of having only fixed MEC (FMEC) entities by exploiting MCS to

The authors propose human-driven edge computing (HEC) as a new model to ease the provisioning and extend the coverage of traditional MEC solutions. From a methodological perspective, they show how it is possible to exploit MCS i) to support the effective deployment of fixed MEC (FMEC) proxies and ii) to further extend their coverage through the introduction of impromptu and human-enabled mobile MEC (M^{2}EC) proxies.

Paolo Bellavista, Luca Foschini, and Leo Gioia are with the University of Bologna; Stefano Chessa is with the University of Pisa; Michele Girolami is with ISTI-CNR.
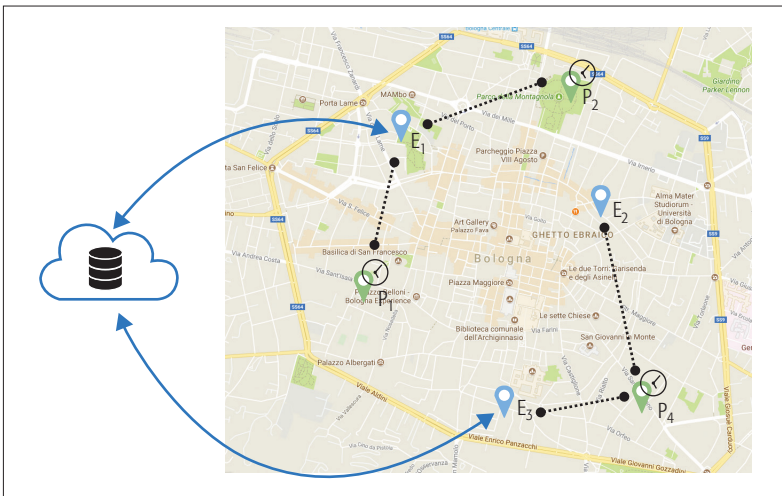
Figure 1. FMEC, M²EC, and couriers in our HEC model.

continuously monitor humans and their mobility patterns, as well as to dynamically re-identify hot locations of potential interest for the deployment of new edges. Second, to overcome FMEC limitations, *HEC enables the implementation and dynamic activation of impromptu and temporary mobile MEC (M²EC) entities* that leverage resources of locally available mobile devices. Hence, an M²EC is a local middleware proxy dynamically activated in a logical bounded location where people tend to stay for a while in a repetitive and predictive way [5], thus realizing a mobile, opportunistic, and participatory edge node. Third, given that M²EC entities, different from FMEC entities, do not implement powerful backhaul links toward the core cloud, *HEC exploits local one-hop communications and the store-and-forward principle by using humans* (moving with their devices) as VM/container couriers to enable migrations between well-connected FMEC and local M²EC entities.

Finally, the article also originally contributes to the literature in the field by presenting *the new HEC model implemented into the real-world ParticipAct MCS living lab* [6], an experiment that involved about 170 students of the University of Bologna for more than two years. This large-scale experiment allowed us to practically characterize M²EC locations and to collect interesting performance indicators about the improvements obtainable via HEC, as well as its drawbacks in terms of stability of M²EC, opportunistically enabled thanks to the participation and contribution of humans in the loop, and possible connectivity issues between FMEC and M²EC.

BACKGROUND AND RELATED WORK

Without claiming completeness, this section first reports some needed background about both MEC and the usage of human mobility features to efficiently enable the new HEC scenario; then it briefly overviews the current state of the art in these very active fields.

MOBILE EDGE COMPUTING ARCHITECTURE AND ITS HUMAN-ENABLED EXTENSION

Our reference scenario is shown in Fig. 1 and enriches the usual three-layer device-MEC-cloud hierarchical architecture (based on the interposition of FMEC entities) by adding new impromptu

and temporarily active M²EC entities. Indeed, the use of MCS and the continuous tracing of humans volunteering for the completion of MCS tasks (monitoring both their mobility and their performance in terms of completion rates of assigned sensing tasks) enable:

• The identification of the best locations, where users tend to aggregate more, thus easing the effective deployment of FMEC and M²EC

• The selection of the users who are more willing to host M²EC acting as local access points to the hierarchical HEC

To take a simple example, we learned from the ParticipAct MCS living lab that there are some aggregation zones through most of the day (they are good candidates as FMEC; E_1, E_2, and E_3 in Fig. 1), while there are other areas that are populated only during shorter and different timeslots (e.g., P_1 and P_4 from 9:00 a.m. to 10:30 a.m., while P_2 is frequented only from 4:00 p.m. to 6:00 p.m.). These latest areas, out of the highly frequented people paths, would highly benefit from being served by a local (in time and space) M²EC, while it would be inefficient and overprovisioned to have additional FMEC there (Fig. 1).

Moreover, from mobile social network (MSN) research, we borrow the key idea of exploiting opportunistic interactions among devices to share virtual machines (VM)/containers that have to be migrated by couriers from/to FMEC (Fig. 1) [7]. We assume one-hop *direct* mode communications that can span from short Bluetooth coverage (i.e., up to 25 m), to broader WiFi-Direct (i.e., up to 150 m) and LTE-Direct (i.e., up to 500 m) ones. Similar to MSN, we consider that *courier* devices automatically down/upload VM/containers from the FMEC as soon as they are close to them; in turn, they share the data gathered with devices roaming in the same M²EC (dotted lines in Fig. 1).

Let us note that the goal of this article is to assess the feasibility of the solution proposed. Accordingly, in our experimental results, adopting an assumption typical of the MSN research area [7], we consider an ideal situation where nodes within transmission ranges can communicate, while, also due to space limitations, we leave a thorough analysis of the issues related to wireless medium interferences, network congestion, and bandwidth allocation for moving data from/to the FMECs and M²ECs as part of our future work. As for the description of the algorithms used to identify FMEC and M²EC, as well as the most suitable human couriers, we refer to the next section.

RELATED WORK

One of the key open challenges in MCS is the possibility of exploiting socio-technical network effects to consolidate and extend the crowd by leveraging communities of volunteers [6]. In the present work, we leverage these new research trends at the crossroads of MCS and MSN to facilitate the identification and formation of M²EC. In addition to the strong novelty in terms of the HEC model, what makes our research efforts unique is also the fact that the ParticipAct living lab [6] closely mimics a very realistic MEC deployment scenario [6].

Focusing on architectural aspects of HEC, the MEC/fog literature has already produced some

relevant modeling work and some seminal design/implementation results. Narrowing down to efforts close to ours, as reported in [1], some first exploratory research activities considered cooperation issues between edges and the core, but only a very few works concentrated on the opportunities of having cooperation between devices and the edges. Considering MCS as application scenario, [2, 3] propose to enhance the MCS process by leveraging intermediate MEC nodes, namely, FMECs, to boost data upload from mobile nodes to the infrastructure [2] and to provide more computing/storage capabilities closer to end mobile devices [3]. To address relevant energy efficiency issues, [8] proposes an interesting approach based on game theory in order to select tasks and manage the scheduling of the edge of the network by introducing the notion of container as a service. A very recent and interesting work, the closest to our HEC concept in what is related to enabling more collaboration between entities co-located at edges is [4]: it proposes not only to have the traditional "vertical" collaboration between devices, MEC, and the cloud level, but also "horizontal" collaboration between entities at the same level via ad hoc communications; however, it neglects the human and social/mobility effects, namely, there is no idea to dynamically identify and form M^2ECs impromptu as in our novel HEC proposal.

Concerning the system and implementation aspects, only a very few research activities have focused on VM/container migration on MEC middleware for mobile services over hostile environments, which is a relevant challenging aspect of modern CPS. Reference [9] highlights the limitations of traditional live VM migration on edge devices by proposing live migration in response to client handoff in cloudlets, with less involvement of the hypervisor, and by promoting migration to optimal offload sites, adapting to changing network conditions and processing capacity. Reference[10] presents the foglets programming infrastructure that handles mechanisms for quality/workload-sensitive migration of service components among fog nodes. Reference [11] proposes the usage of cloudlets to support mobile multimedia services and adjusts resource allocation triggered by runtime handoffs. Referrence [12] evaluates handoff conditions in relation to various parameters (e.g., signal strength, bit rate, number of interactions between cloudlets and associated devices) and decides to offload the computation based on fuzzy techniques. Reference [13] proposes a multi-agent-based code offloading mechanism using reinforcement learning and code blocks migration to reduce both execution time and energy consumption of mobile devices. While these papers explore the highly relevant issue of integrated management of handover operations with VM/container migration, none of them explore the possibility of using humans as storage/VM/container carriers.

EXTENDING MOBILE EDGE COMPUTING THROUGH THE CROWD

This section first overviews the HEC distributed architecture and its main components and functions, and then presents the main guidelines and engineering trade-offs we implement for the selection of FMEC, M^2EC, and human couriers.

ARCHITECTURE AND SOME IMPLEMENTATION INSIGHTS ABOUT OUR ORIGINAL HEC MIDDLEWARE

HEC extends the emerging MEC three-layer hierarchical architecture through the distinction of two types of MECs: FMEC and M^2EC. In the following, we present the main functions of our proposed HEC middleware that implements FMEC and M^2EC at fixed edges and at mobile devices temporarily serving as local edges, respectively.

Our MEC middleware consists of two primary components:
1. The available Elijah platform (http://elijah.cs.cmu.edu/), one of the most promising and complete open source projects to realize general-purpose support for MEC. Elijah supports a subset of basic OpenStack mechanisms for cloud resource management, including functions to migrate computing/storage resources in the form of VM near the dynamically identified edges of the network. In addition, basic mechanisms for VM synchronization and orchestration in the proximity of mobile device localities are included [14].
2. Our Elijah-based original extension module, called HEC middleware, that relevantly increases the basic MEC support with facilities for:
   – Runtime enabling of M^2EC features at selected mobile devices
   – (Docker-oriented) container support for more lightweight migration of computing/storage functions at HEC edges
   – Complete decoupling of regular mobile devices and their Elijah-based support at edges, for example, via seamless and transparent management of needed VM/container migration in the case of mobile device handoffs between different edges during service provisioning [14].

In particular, concerning the dynamic migration of VM/containers (which typically realize MCS services) at HEC nodes, our middleware implements two types of related support, either proactive or reactive. On one hand, the proactive migration allows minimization and automation in an efficient way virtualized function migration, by pre-loading the needed functions in advance on target HEC nodes. For instance, this approach is exploited to support end-user services with continuity requirements when device handoff prediction algorithms can be used with an adequate success probability. In this case, VM/container migration is started before receiving explicit location update by mobile devices due to their actual handoff procedures (note that VM/containers may be synthesized for migration purposes from scratch or by considering only the modified overlays/slices to enable more lightweight migrations [14]). On the other hand, our reactive migration support is triggered when a mobile device explicitly requests to move a virtualized function from one HEC edge to one another due to a device handoff (note that this is typically the solution we use when there is temporary unavailability of connectivity to the cloud, where typically sophisticated handoff prediction algorithms are run). Once triggered, our HEC middleware checks the possibility to execute the virtualized function required

Our architecture considers both spatial and temporal dimensions for selecting FMECs and M^2ECs by analyzing the human mobility for an observation period. Concerning FMECs, we exploit the temporal dimension by considering places of our region remaining mostly active during the whole 24 hours: these are spatial locations not subject stable along the time.

on the targeted MEC node and forwards to the lower-layer Elijah platform the request for the creation of a VM/container to execute the task. Finally, the HEC middleware returns to the mobile device the reference to transparently invoke the specified VM/container for future direct VM/container-to-device interactions.

By focusing on our HEC middleware (currently based on the Android OS) at mobile devices, first, we distinguish between regular mobile devices (capable of working only as service clients) and powerful ones (which may be promoted dynamically to host virtualized functions and to serve as M^2EC nodes). In our current implementation, the latter are statically identified based on their device hardware/software profiles (at the moment high-end tablets or laptops that are locally paired with smartphones, typically belonging to the same human owner, acting as MCS sensors). The evolution trend of mobile/embedded devices is such that the potential set of mobile nodes that can be promoted to M^2EC at runtime is ever increasing; this is the case for experiments showing that even limited RaspberryPI nodes can adequately host OpenStack++ middleware [14]; on these nodes we assume that our HEC middleware is already installed before starting service provisioning.

From the MCS application perspective, the tasks that have already been implemented and experimented for execution at HEC nodes are:
- Video analysis/processing for face recognition that can trigger online alarms
- Analytics on all or fused monitoring indicators over geographical areas of highest interest and density (data fusion, history-based processing of temporal series,etc.).

FMEC and M2EC Selection and Human-Enabled VM/Container Migration

Our architecture considers both spatial and temporal dimensions for selecting FMECs and M^2ECs by analyzing the human mobility for an observation period. Concerning FMECs, we exploit the temporal dimension by considering places of our region remaining mostly *active* during the whole 24 hours: these are spatial locations not subject stable along the time. For this purpose, we use the DBSCAN algorithm, which clusters together the positions of the users [15]. Points lying within the distance ε are considered part of the same cluster, a new cluster is detected only if it contains at least γ different points. DBSCAN results in K distinct clusters; we purge them by filtering out those clusters reporting a low number of different users, specifically those clusters composed of positions reported by only a few users. After such a filtering process, we only considered $k \leq K$ clusters.

Concerning M^2EC, we consider the temporal dimension by selecting those places of our region becoming *active* only during specific time intervals. In fact, our goal is to dynamically (re)configure our cloud architecture according to the natural rhythm of a city. For this purpose, we analyze the human mobility during some temporal frames characterizing the typical phases of a working day. For each of the frames, we cluster together the user positions with a process similar to the one described for the FMEC. Also in this case, DBSCAN results in H clusters, of which we keep the top $h \leq H$.

The next step is moving data from FMEC to M^2EC and vice versa. For this purpose we consider humans (i.e., *couriers*), and the mobile devices they carry in their pockets equipped with our HEC middleware, as the primary actors that can be involved in the loop. In particular, we assume that mobile devices provide different kinds of network interfaces (short, medium, and broadband) as well as storage capacity to enable store-and-forward of VM/containers between FMEC and M^2ECs, as well as to replicate them across users concurrently joining the same M^2EC so as to increase the stability, defined as the capability to maintain the VM/container active in the M^2EC. For the selection of couriers, we keep track of user mobility and prefer those users that have more repetitive and predictable behavior: the more a user commutes from an FMEC to an M^2EC, the more he/she is a good courier candidate.

Finally, since not all the FMECs are connected to all the M^2ECs during the 24 hours, we consider the possibility of saving bandwidth in the cloud-to-FMEC direction and consequently storage resources at FMECs. With a closer view to details, HEC implements a load balancing function that exploits mobility traces and past connectivity history (between FMECs and M^2EC) to select which VMs/containers to move from the cloud to the FMECs by applying a locality principle that preloads at the FMEC only those ones that are more likely to be stored and forwarded by a courier toward an M^2EC.

EXPERIMENTAL RESULTS

Our experiments are based on the ParticipAct dataset [6], which implements an MCS platform. The ParticipAct dataset includes data collected from about 170 students in the Emilia Romagna region of Italy by including their mobility traces. In particular, ParticipAct users are equipped with Android smartphones running an app that tracks their locations using the Google location application programming interfaces (APIs) and a fixed sampling interval of 2.5 minutes. All the data collected are stored in the ParticipAct backend, which provides some basic analytics features, such as filtering out location samples with low accuracy and interpolation of user locations between consecutive samples. The data collection covers about two years overall, and in this work we consider a period of one month, from March to April 2014, during the spring 2014 semester to ensure that most of the participants move in the city.

Of course, the dynamics of a city change according to the vacation periods of the year (i.e., working vs. vacation days) and of the week (i.e., working days, often with slightly different behaviors on Mondays–Fridays, vs. weekends); often these dynamics are predictable to some extent, and we could have adapted all initial HEC configurations according to that. However, to demonstrate the applicability of the proposed approach, in the following we consider a baseline worsening scenario where we statically decided (based on a pragmatic analysis of the mobility traces) to use as the HEC configuration parameters (i.e., k = 6 and h = 6) average values calculated over all the days in the considered period. We refer to a baseline scenario where HEC parameters are not

finely tuned according to the vacation/weekly dynamics of the targeted system. This scenario also allows us to show the strengths of the adopted approach.

IDENTIFYING FMECS AND M²ECS

In this first set of experimental results, we analyze the mobility traces with the objective of spotting places in which to deploy FMEC (denoted by E) as well as M²EC (denoted by P) according to dynamics of human mobility. Both the MECs (fixed and mobile) have been identified by using an observation period of three weeks. Specifically, we use the first three weeks of the dataset (March 1 to March 23, 2014) to learn how people in ParticipAct move and therefore identify places to consider for possible deployments of FMECs and M²EC. Figure 2 shows six FMECs selected as well as the centroid of each of the clusters marked as a red circle ($E1$ to $E6$).

Concerning M²EC, we analyze those places of our sensing region becoming active only during specific time intervals. To this purpose, we analyze the mobility traces by slicing the dataset into 5 temporal frames: 9 a.m. to 1 p.m., 1 p.m. to 5 p.m., 5 p.m. to 8 p.m., 8 a.m. to 12 a.m., and 12 a.m. to 9 a.m. For each of these time intervals, we obtain six clusters reported in Fig. 3 (the M²EC centroids are not reported for better readability of the figure).

ASSESSING HEC CONNECTIVITY, STABILITY, AND LOAD BALANCING

This second set of experiments, different from the experimental results presented in the previous subsection, is based on the fourth week of the dataset (March 24 to March 30, 2014) to validate the MEC selection previously done with respect to the mobility of the fourth week of our dataset. We aim to validate the effectiveness of our HEC implementation; we start by showing some performance measures to quantify the *connectivity* and *stability* metrics as previously defined, and then assessing the performance improvements obtainable via FMEC load balancing.

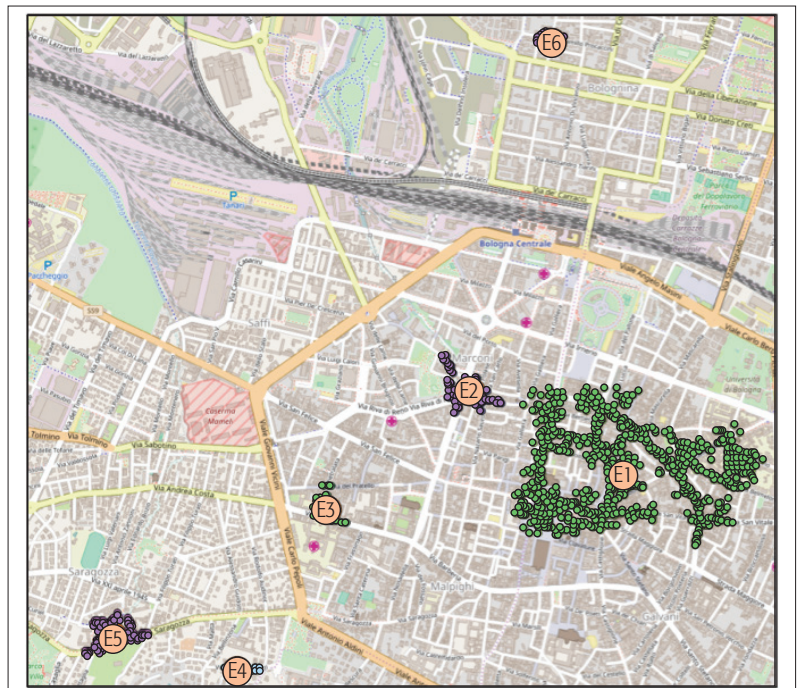Connectivity measures the robustness of links

Figure 2. MECs identified as DBSCAN clusters.

connecting FMEC and M²EC. A link between two MECs exists if users move from/to an FMEC toward an M²EC. The robust links are those characterized by a high number of users visiting FMEC and M²EC during the same time interval, while weak links are those MECs sporadically connected by a few users. Connectivity is an interesting marker, useful to understand and predict the capability of the HEC architecture to remain connected over time. Indeed, if during a specific time interval a pair of FMEC and M²EC remains disconnected, the data stored on the FMEC cannot be moved to the M²EC, giving rise to a sort of temporal disconnection of the cloud.

We measure the connectivity of the MECs previously identified as a temporal graph, which is a graph evolving over time. Figure 4 shows, with a
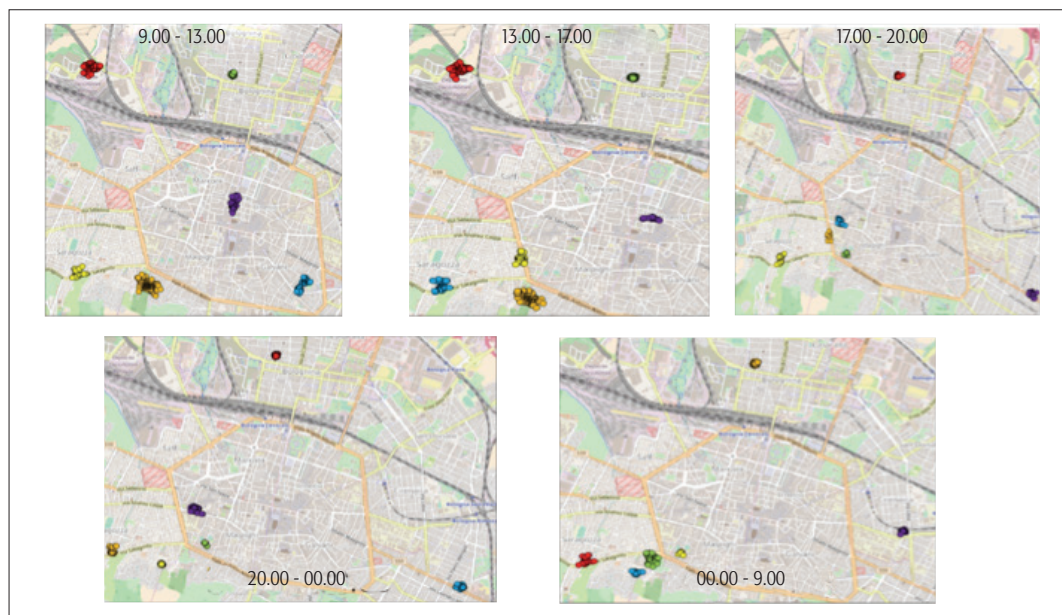

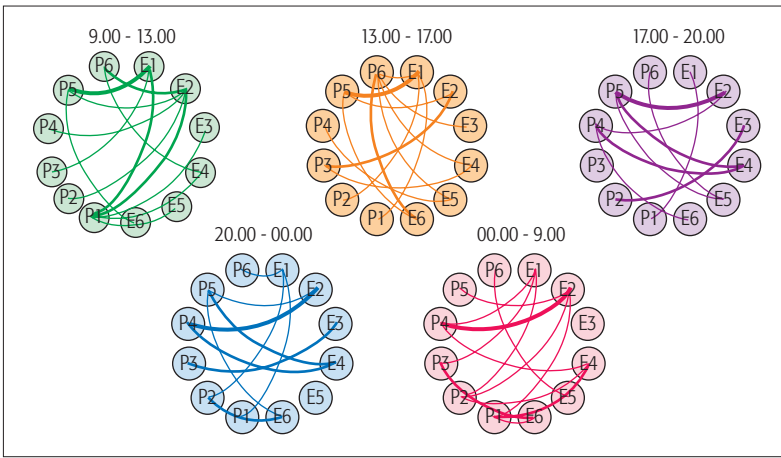**Figure 3.** M²EC identified as DBSCAN clusters.

Figure 4. Measure of connectivity as a temporal graph.

circular layout, that FMECs (E1 to E6) are always the same, while M²ECs (P1 to P6) may change for each of the five intervals, as shown in Fig. 3. The weight of the links connecting FMEC and M²EC (represented by different thicknesses of lines in Fig. 4) is proportional to the number of users whose position has been reported in both MECs during the same interval.

For example, E1 and P5 during 9 a.m.–1 p.m. are well connected, while the link connecting E1 and P3 during the same interval is weak. A first consideration is that it is possible to identify some dominant FMECs, namely those FMECs remaining well connected during the whole 24 hours. Examples of such FMECs are E1, E2, E4 and E6. These are FMECs deployed in core areas of the city, representing a sort of backbone of our architecture. Second, we observe that during the 24 hours all the M²EC (P points on Fig. 4) are connected with at least one FMEC. Finally, we note that connectivity among MECs is significantly robust also during evening and night intervals (8 p.m.–12 a.m. and 12 a.m.–9 a.m.), ensuring the possibility of also moving VMs/containers from/to FMECs when user mobility tends to reduce.

The second metric we analyze refers to the stability of our cloud architecture, which measures the capability of M²ECs to locally maintain those data previously moved from an FMEC. In particular, we aim to measure if, once data are moved from/to a M²EC, such data remain on the MEC for a while. For this purpose, we consider that users joining an M²EC can exchange data directly by exploiting short, medium, and high range network interfaces; specifically, we consider three kinds of direct mode communications: Bluetooth (up to 25 m), Wi-Fi Direct (up to 150 m) and LTE Direct (up to 500 m). Given our five time intervals, we measure the percentage of time during which at least two users join the same M²EC. Figure 5 shows, for each M²EC, a triplet of histograms of different colors: orange represents stability via Bluetooth interface, green stability via Wi-Fi Direct, and blue the stability obtained via LTE Direct interfaces; the black line on the left, instead, reports the 100 percent scale.

We observe that stability is preserved for each of the M²ECs during the time intervals. There are cases in which the stability can be preserved only by means of high-range interfaces, such as the M²EC deployed on the north region of Fig. 5 at interval 5 p.m.–8 p.m.: in this case, users use LTE Direct connections in order to enable local replication. In contrast, there are situations in which stability is guaranteed for all the network interfaces, such as the M²EC deployed in the south region of Fig. 5 during the interval 8 p.m.–12 a.m.

The last metric we present refers to HEC load balancing across FMECs, described earlier. In particular, we measure the benefits of balancing the load by focusing on the FMEC-to-M²EC direction and analyzing, for each of the six FMECs, how many M²ECs they are connected to via at least one courier. More specifically, given the FMEC E_i ($i = 1$ to 6) we measure the percentage p of M²ECs for which there is no need to pre-cache VMs/containers since no connections from E_i are available to such M²ECs; this percentage ranges from 43 percent for E_2 up to 93 percent for E5, and therefore we consider it a promising strategy
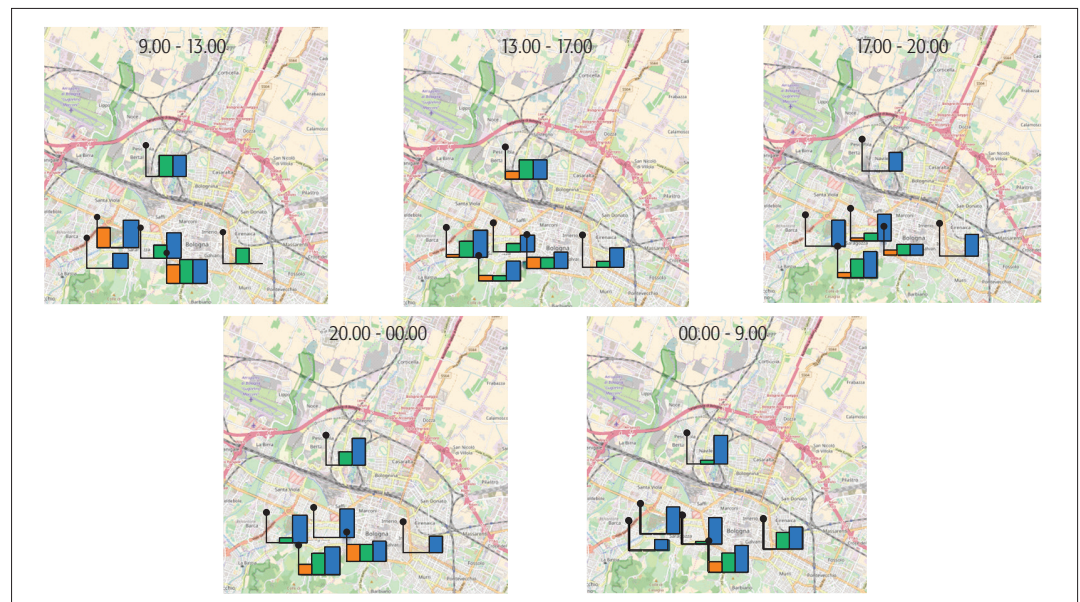

Figure 5. Stability of the M²ECs with different communication modes.

to obtain significant performance improvements by avoiding unnecessary waste of bandwidth and storage resources at FMECs.

Let us conclude by remarking that experimental results presented in this section are based on the analysis of the ParticipAct mobility dataset, but from our experience on mobility datasets (i.e., [7]) we believe that the same methodology can be adopted in other experiments following the same reproducible steps to cluster user positions along time, and then on spotting those locations being activated during specific time windows or remaining active for long times.

CONCLUSIONS

In this article, we present HEC, a new model to ease the provisioning and extend the coverage of traditional MEC solutions by bringing together the best of the MEC and MCS research areas. The cornerstone of our proposal lies in the ability to leverage human social and mobility effects to broaden MEC coverage through the impromptu formation of M^2ECs and to make it more efficient and aware of the pulse of the city by dynamically tailoring the HEC hierarchical architecture to it. Moreover, results obtained from our ParticipAct living lab confirm that clustering techniques can be employed to effectively plan both FMEC and M^2EC placement.

These encouraging results are pushing us to further investigate and refine our HEC model, and we are currently exploring various main areas. We are working to enable self-adaptable finetuning of our HEC middleware to the different dynamics and variations of the city pulse, and we are investigating how to increase the connectivity between FMEC and M^2EC also considering some additional user profiling information for the courier selection.

> We are working to enable the self-adaptable fine tuning of our HEC middleware to the different dynamics and variations of the city pulse, and we are investigating how to increase the connectivity between FMEC and M^2EC considering also some additional user profiling information for the courier selection.

REFERENCES

[1] S. Wang et al., "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications," IEEE Access, vol. 5, Mar. 2017, pp. 6757–79.
[2] S. K. Datta et al., "oneM2M Architecture Based IoT framework for Mobile Crowd Sensing in Smart Cities," Proc. 2016 Euro. Conf. Networks and Commun., 2016, pp. 168–73.
[3] K. M. S. Huq et al., "Enhanced C-RAN Using D2D Network," IEEE Commun. Mag., vol. 55, no. 3, Mar. 2017, pp. 100–07.
[4] T. X. Tran et al., "Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges," IEEE Commun. Mag., vol. 55, no. 4, Apr. 2017, pp. 54–61.
[5] L. Pappalardo et al., "Returners and Explorers Dichotomy in Human Mobility," Nature Commun. vol. 6, Article 8166, 2015.
[6] G. Cardone et al., "The ParticipAct Mobile Crowd Sensing Living Lab: The Testbed for Smart Cities," IEEE Commun. Mag., vol. 52 , no. 10, Oct. 2014, pp. 78–85.
[7] M. Girolami, S. Chessa, and A. Caruso, "On Service Discovery in Mobile Social Networks: Survey and Perspectives," Computer Networks, vol. 88, 2015, pp. 51–71.
[8] K. Kaur et al., "Container-as-a-Service at the Edge: Trade-off between Energy Efficiency and Service Availability at Fog Nano Data Centers," IEEE Wireless Commun., vol. 24, no. 3, 2017, pp. 48-56.
[9] K. Ha et al., "Adaptive VM Handoff Across Cloudlets," tech. rep. CMU-CS-15-113, CMU School of Comp. Sci., 2015.
[10] E. Saurez et al., "Incremental Deployment and Migration of Geo-Distributed Situation Awareness Applications in the Fog," Proc. ACM Int'l. Conf. Distrib.and Event-Based Systems, 2016, pp. 258–69.
[11] M. Felemban, S. Basalamah, and A. Ghafoor, "A Distributed Cloud Architecture for Mobile Multimedia Services," 2013.
[12] A. Ravi and S. K. Peddoju, "Handoff Strategy for Improving Energy Efficiency and Cloud Service Availability for Mobile Devices," Wireless Personal Commun., vol. 81, no. 1, 2015, pp. 101–32.
[13] M.G.R. Alam et al., "Multi-Agent and Reinforcement Learning Based Code Offloading in Mobile Fog," Proc. Int'l. Conf. Info. Networking, 2016.
[14] P. Bellavista, M. Solimando, and A. Zanni, "A Migration-Enhanced Edge Computing Support for Mobile Services in Hostile Environments — Lessons Learnt from Platform Implementation and Deployment," to appear, Proc. Int'l. Wireless Commun. and Mobile Computing Conf., 2017.
[15] M. Ester et al., "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc. Int'l. Conf. Knowledge Discovery, AAAI Press, 1996, pp. 226–31.

BIOGRAPHIES

PAOLO BELLAVISTA [SM] (paolo.bellavista@unibo.it) is an associate professor at the University of Bologna, Italy. His research activities span from mobile agent-based middleware and pervasive wireless computing to location/context-aware services and vehicular sensor networks, from big data adaptive stream processing to adaptive multimedia. He is the Editor in Chief of MDPI Computers, and he serves on the Editorial Boards of IEEE TNSM, IEEE TSC, Elsevier PMC, Elsevier JNCA, Elsevier JSA, Springer JNSM, and Springer WINET.

STEFANO CHESSA (stefano.chessa@unipi.it) graduated from the University of Pisa, Italy, from which he received a Ph.D. degree in computer science in 1999. Since 2015 he has been an associate professor at the Computer Science Department of the University of Pisa. He has also been a research associate of the ISTI-CNR Institute since 2003. His current research interests are in the areas of the Internet of Things, smart environments, ambient assisted living, activity recognition, and indoor localization.

LUCA FOSCHINI [M] (luca.foschini@unibo.it) graduated from the University of Bologna, from which he received a Ph.D. degree in computer engineering in 2007. He is now an assistant professor of computer engineering at the University of Bologna. His interests include distributed systems and solutions for system and service management, management of cloud computing, context data distribution platforms for smart city scenarios, context-aware session control, mobile edge computing, and mobile crowdsensing and crowdsourcing.

LEO GIOIA (leo.gioia@studio.unibo.it) graduated from the University of Bologna, from which he received a B.Sc. degree in 2015. Currently, he is studying for a second cycle degree. His interests include distributed file systems, cloud computing, big data processing, and storage system support for container applications.

MICHELE GIROLAMI (michele.girolami@isti.cnr.it) graduated from the University of Pisa, from which he received a Ph.D. degree in computer science in 2015. Currently, he is a research staff member at CNR-ISTI with the Wireless Network Laboratory. His research interests include data forwarding and service discovery in mobile social networks, crowdsensing techniques, and context-aware middleware for smart environments.