MVR: an Architecture for Computation Offloading in Mobile Edge Computing

Xiaojuan Wei[1], Shangguang Wang[1*], Ao Zhou[1], Jinliang Xu[1], Sen Su[1], Sathish Kumar[2], Fangchun Yang[1]
[1]State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications, Beijing, China
[2]Department of Computer Science and Information Systems
Coastal Carolina University, South Carolina, USA
{weixjmm, sgwang, aozhou, jlxu, susen}@bupt.edu.cn; skumar@coastal.edu; fcyang@bupt.edu.cn

***Abstract*—As communication and sensing capabilities of mobile devices increase, mobile applications are becoming increasingly complex. The ability of computation offloading, which is one of the main features of mobile edge computing gains relevance as a technique to improve the battery lifetime of mobile devices and increase the performance of applications. In this paper, we describe the offloading system model and present an innovative architecture, called "MVR", contributing to computation offloading in mobile edge computing.**

***Keywords*—Mobile Edge Computing, Edge Cloud, Computation Offloading, Architecture, Internet of Things.**

I. Introduction

Mobile devices are becoming increasingly popular. Cisco Visual Networking Index – Global Mobile Data Traffic Forecast [1] was updated in February 3, 2016, which indicates that monthly global mobile data traffic will be 30.6 exabytes by 2020, up from 3.7 exabytes per month at the end of 2015. There will be 50 billion connected devices by 2020, contributing to the vision of Internet of Things. However, this unparalleled growth is not matched the improvement on mobile devices' batteries, whose lifetime is not improving at the same pace [2]. Given the tremendous increase in the usage of the mobile devices, solving the energy impediment is one of the mobile industry's foremost challenge. There are many researches that are conducted from different views to study the issue of energy consumption [3], [4], [5], [6]. While programming optimization [7] and vulnerability detection [8] of energy consumption in application software view and components optimization [9] in hardware view enhance the devices energy efficiency, these methods only alleviate the excessive energy consumption based on the device itself. Terminal and cloud fusion provide a completely different technical direction for energy consumption optimization. This cloud-terminal fusion concept provide available resources in the cloud to support the operation of the terminal application. In other words, this federal technology sees the cloud as an extension of the terminal by offloading application computing from the terminal to the cloud. Mobile Edge Computing (MEC) as a new paradigm provides IT and cloud computing capabilities within the Radio Access Network (RAN) in close proximity to mobile subscribers [10], [11], [12]. This is indeed a possible way to overcome the obstacle of the battery and to enable the mobile devices, whenever possible and convenient, to offload their most energy-consuming (computation-intensive) tasks to MEC environment. This concept will reduce a long application execution time on mobile devices that results in reduction of power consumption. But, how can we implement this promising project in background of MEC?

We can describe this work as a "3WH" issue, including: why to offload, what to offload, when to offload and how to offload. In this paper, we will describe the offloading system model and present an innovate architecture named as MVR for computation offloading under the MEC.

II. Offloading in MVR

MEC is a federation of mobile devices and Edge Cloud to support computation offloading. Offloading computation does not necessarily imply transferring all the program execution to the Edge Cloud. In general, a program is first divided into modules, some of them need to be run locally (on the mobile device). For instance, all modules controlling the input/output peripherals. For the rest of the modules, a decision has to be taken on what is more appropriate to offload. So, how to divide the offloading program and how to make a decision?

We present MVR, an innovative architecture through fine-grained offloading execution to Edge Cloud platform. MVR enable the use of virtual resources (VRs) in Edge Cloud to alleviate the resource burden and reduce energy consumption of the mobile device itself and improve the application performance. Conceptually, MVR automatically transforms a single-machine execution (on a mobile device) into a distributed execution (on the mobile device and Edge Cloud).

A. MVR system model

In single-machine execution model, as shown in the Fig. 1, the procedure (one task of the application) is executed sequentially in the mobile device. While in distributed execution model, the procedure in the mobile device is divided into five tasklets, including three non-offloaded tasklets (marked as "0" in the procedure) and two tasklets that were offloaded to Edge Cloud (marked as "1" in the procedure). Edge Cloud has a variety of VRs (VMs or containers or others), the role of VRs is responsible for the execution of the "1" tasklets.

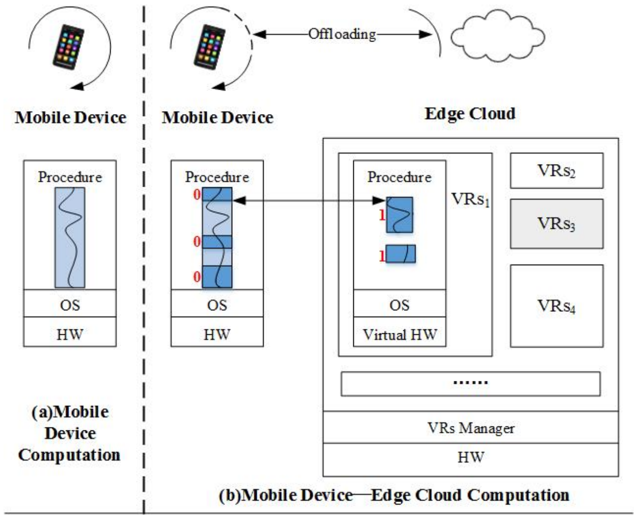

Fig. 1. MVR system model.

B. *High-level view of the MVR architecture*

As mentioned above, Edge Cloud have a variety of VRs and all execution would be in these VRs. Fig. 2 shows a high-level view of the MVR architecture.

On the Mobile Device, the MVR runtime consists of four components:

- **Proxy Monitor**: responsible for the discovery of resourceful proxy, i.e. Edge Cloud;
- **Decision Engine**: combines with static analysis and dynamic analysis to support offloading decision. Static Analyzer is responsible for identifying methods that can be offloaded to the Edge Cloud based on a set of limitation factor: the tasklet requiring local resources must be executed on mobile device, the tasklet sharing local state must be executed on one same mobile device, and avoiding the appearance of nested offloading. Dynamic Analyzer analyze the cost and benefit of computation offloading about the current cyber conditions of dynamic changes such as bandwidth;
- **Tasklet Distributor**: marks the tasklet as "0" or "1", in other words, it distributes the computational tasklet such that is executed on the mobile device or VRs in Edge Cloud;
- **Executor**: executes the procedure and interacts with Edge Cloud.

On the other side, each VRs in Edge Cloud have six main components:

- **Registration Manager**: responsible for the registering and authenticating of the mobile devices that are requesting services;
- **Predictor**: predicts the arrival of the mobile user and execution time of tasklet. Some researches have shown that the user's movement abide by a certain law [13]. We name the offloading task regularly as "Very Important Person (VIP)" and provide resources in advance for these "VIP" through authentication and prediction for further improving application performance;
- **Partition Analyzer**: analyzes the current network conditions of Edge Cloud and construct offloading overhead function combine with the Decision Engine of the mobile device;
- **Decision Solver**: provides a decision method about the computational tasklets for a minimum energy computation or a short execution time and decides the "0" tasklet to stay locally and the parts of "1" to offload to Edge Cloud;
- **Virtual Resource Controller**: the main component of Edge Cloud are virtual resources (VRs). Each of these VRs is managed by a certain VRs Controller (VRC) in the Edge Cloud. When the offloaded computational tasklets arrives, the required resources are allocated through Virtual Resource Controller. It is possible that one tasklet may need multiple VRs, so the Virtual Resource Controller need to aggregate the execution results and feed back to the mobile device;
- **Executor**: similar to mobile device, Executor executes the procedure and fulfill interaction during execution.

In addition to the above components, there are also some basic factors for system operation: operating system (OS), hardware (HW), and Virtual HW in Edge Cloud.

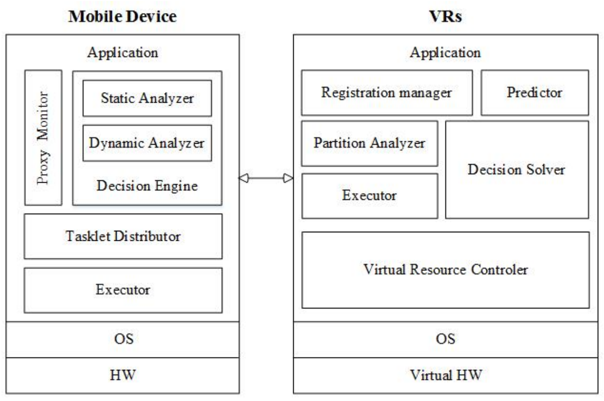

Fig. 2. The MVR architecture of the prototype.

C. *The process of MVR offloading*

Characteristics of applications that can benefit from Edge Cloud are typically time-critical and require a very low computing and communication latency in an environment with limited bandwidth, and limited computing power. For example applications such as Linpack (computation-intensive), 3D Car Racing (interaction-intensive) and Chess (computation & interaction-intensive).

This section describes an application of MVR in augmented reality (AR). As mentioned in [14], an AR algorithm according greyscale video frame to show the 3D objects was split six tasklets: VideoSource, Renderer, Tracker, Mapper, Relocalizer and Object Recognizer. The relationship of these tasklets is showed in Fig. 3. The VideoSource fetches video frames what are rendered by the Renderer and calculated by the Tracker to know the camera pose by 2D image features and 3D feature points whose map is generated and updated by Mapper. Object Recognizer tries to locate known objects and Relocalizer tries to relocate the position when no feature points are found.

Following this example, a mobile device running the AR application seeks resourceful Edge Cloud, the Edge Cloud identifies and predicts the mobile device and classifies as new user, ordinary user or "VIP". According to Decision Engine and Decision Solver, a completed AR algorithm is divided into six tasklets. In this AR example, the Tasklets of VideoSource and the Renderer have to be executed on the mobile device, as they access device specific hardware. So, these tasklets would be marked as "0". While the Tracker analyses video frames and calculates the camera pose and the Relocalizer that relocate the camera position should be marked as "1". Each tasklet will be assigned to a mobile device or Edge Cloud by Tasklet Distributor. The Virtual Resource Controller in Edge Cloud begins to allocate matched resources by user's identity and task's requirements for the tasklet that is offloaded to the Edge Cloud. When each tasklet is executed, the certain VRC in Edge Cloud will collect the results back to the mobile device. Mobile device aggregates the results according to the relationship of the tasklets and performs the next step until the application is completed.

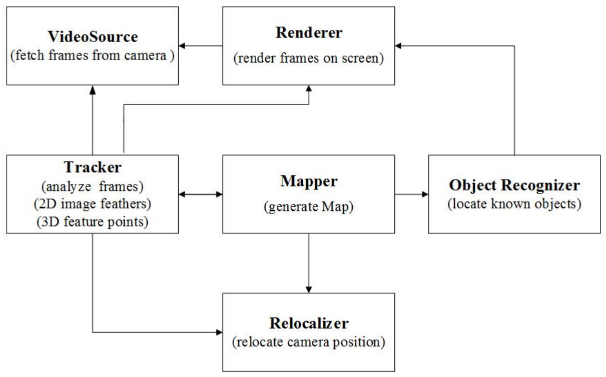

Fig. 3. The relation of the AR algorithm tasklets.

III. Related Work

There have been many related attempts to make mobile devices use remote cloud or collaborative execution to improve performance and energy consumption. Some researches have been focused on how to effectively offload the tasks and get the results back promptly. In this section, some representative related works are introduced.

MAUI [15] maximize the potential for energy savings while minimizing the burden on the programmer by using a combination of code portability, programming reflection, serialization, type safety, and network costs. In MAUI, applications are pre-built and executed on HTC smart phone and MAUI servers through a dual-core desktop running Windows 7 with the .NET Framework v3.5. It was observed that MAUI was able to offload mobile game components to servers in the remote cloud can save 27% of energy consumption for computer games and 45% for the chess game. CloneCloud [16] distributes the computational effort of the resource-intensive parts of an execution to a more powerful clone created on a cloud and synchronized periodically or on-demand with mobile devices. The offloaded task can be either as a whole to be executed in the clone or executed on both entities by partitioned into pieces. A related limitation is that CloneCloud does not virtualize access to native resources that are not virtualized already and are not available on the clone. It may be pursued in conjunction with thread-granularity migration. MAUI enables mobile applications to reduce energy consumption through automated offloading. CloneCloud can minimize either energy consumption or execution time of applications by automatically identifying compute-intensive parts. COSMOS [17] takes the next important step further to bridge the gap between offloading demands and the availability of cloud computing resources. COSMOS can perform risk-based offload decisions to overcome the uncertainties caused by variable network connectivity. COSMOS may minimize the cost of power consumption and reduces the cost of leasing cloud resources per request to the provider.

Offloading to the remote cloud is not always a solution, because of the high WAN latencies, especially for applications with real-time constraints such as augmented reality. Therefore the possible way is to bring computational resources closer to the mobile user, such as collaboration and alliance of mobile devices or cloudlet, as shown in Fig. 4.

To solve the high WAN delay caused by offloading to the remote cloud, Cloudlets [18] bring abundant resources closer to the users, only one hop distance between mobile device and Cloudlet. The infrastructure of Cloudlets is decentralized and widely-dispersed which is self-managing, and requiring little more than power, Internet connectivity, and access control for setup. Simply, the cloudlet is a predefined cloud in proximity that consists of some static stations and is generally installed in public domains, but with no guarantee of availability near a mobile device. Karim Habak [19] considers how a collection of co-located devices can be orchestrated to provide a cloud service at the edge. The proposed femtocloud system provides a dynamic, self-configuring and multi-device mobile cloud for a cluster of mobile devices. The femtocloud system mainly contains two parts: an extremely stable and deliberately configured controller and many highly mobile and unpredictable devices, i.e. a compute cluster responsible for performing the computation. Spontaneous proximity cloud (SPC) [20] drops the high delay between the mobile device and the cloud by a set of neighboring mobile devices in a collaborative manner. Another approach with dynamic voltage and frequency scaling for energy minimization is presented in [21], which can determine the tasks to be offloaded onto the cloud and mapping the remaining local tasks onto other potentially heterogeneous cores in the same mobile device.

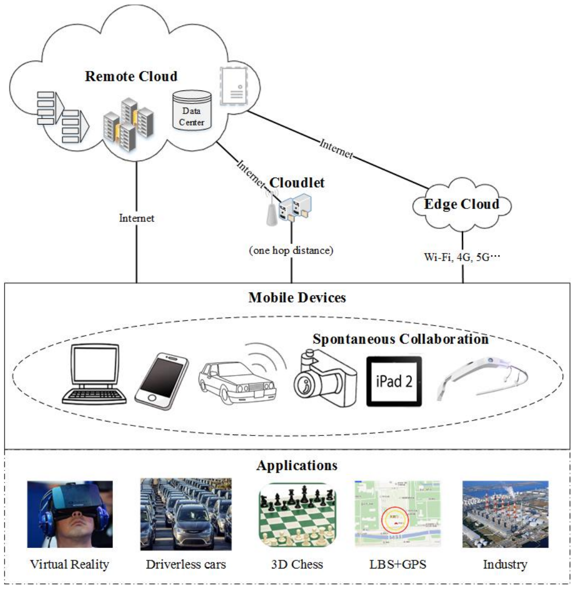

Fig. 4. The application model of computation offloading.

IV. Conclusion

In this paper, we present MVR architecture for computation offloading in Mobile Edge Computing to creat a wonderful bridge between computation/interaction-intensive applications and the resourceful Edge Cloud. With the advent of new technology such as 5G, there are many challenging work awaiting us to solve. For example, there is a lack of understanding under a multi-dynamic environment about the joint optimization problem involving leasing cost, runtime, energy consumption, delay time and so on. Can we design a mechanism to create and maintain a stable "Mobile Federation"? How effective is such an approach in a setting with multiple clouds and multiple service providers? The security level of mobile users can change from location to location frequently and quickly. We predict that the use of privacy services for mobile users in MEC environment will increase the computation offloading complexity. In our future work, we will continue to explore on the problem of computation offloading.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No. 61472047)

References

[1] C. VNI, "Cisco visual networking index: Global mobile data traffic forecast update, 20152020," *URL: http://www. cisco. com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_ paper_c11-520862. pdf*.

[2] M. R. Palacin, "Recent advances in rechargeable battery materials: a chemists perspective," *Chemical Society Reviews*, vol. 38, no. 9, pp. 2565–2575, 2009.

[3] A. Zhou, S. Wang, B. Cheng, Z. Zheng, F. Yang, R. Chang, M. Lyu, and R. Buyya, "Cloud service reliability enhancement via virtual machine placement optimization," *IEEE Transactions on Services Computing*, 2016.

[4] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Infocom, 2012 Proceedings IEEE*, pp. 945–953, IEEE, 2012.

[5] B. P. Rimal, D. P. Van, and M. Maier, "Mobile-edge computing vs. centralized cloud computing in fiber-wireless access networks," in *Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on*, pp. 991–996, IEEE, 2016.

[6] J. Xu, A. Zhou, S. Wang, Q. Sun, J. Li, and F. Yang, "Machine status prediction for dynamic and heterogenous cloud environment," in *Cluster Computing (CLUSTER), 2016 IEEE International Conference on*, pp. 136–137, IEEE, 2016.

[7] D. Li and W. G. Halfond, "An investigation into energy-saving programming practices for android smartphone app development," in *Proceedings of the 3rd International Workshop on Green and Sustainable Software*, pp. 46–53, ACM, 2014.

[8] A. Banerjee, L. K. Chong, S. Chattopadhyay, and A. Roychoudhury, "Detecting energy bugs and hotspots in mobile apps," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 588–598, ACM, 2014.

[9] K. H. Kim, A. W. Min, D. Gupta, and P. Mohapatra, "Improving energy efficiency of wi-fi sensing on smartphones," vol. 34, no. 17, pp. 2930–2938, 2011.

[10] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, *et al.*, "Mobile-edge computing introductory technical white paper," *White Paper, Mobile-edge Computing (MEC) industry initiative*, 2014.

[11] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computinga key technology towards 5g," *ETSI White Paper*, vol. 11, 2015.

[12] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *IEEE International Conference on Intelligent Systems and Control*, 2016.

[13] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation offloading for service workflow in mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3317–3329, 2015.

[14] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, "Cloudlets: Bringing the cloud to the mobile user," in *Proceedings of the third ACM workshop on Mobile cloud computing and services*, pp. 29–36, ACM, 2012.

[15] 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*, pp. 49–62, ACM, 2010.

[16] 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*, pp. 301–314, ACM, 2011.

[17] C. Shi, K. Habak, P. Pandurangan, M. Ammar, M. Naik, and E. Zegura, "Cosmos: computation offloading as a service for mobile devices," in *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, pp. 287–296, ACM, 2014.

[18] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE pervasive Computing*, vol. 8, no. 4, 2009.

[19] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," in *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pp. 9–16, IEEE, 2015.

[20] R. Golchay, F. L. Mouël, J. Ponge, and N. Stouls, "Spontaneous proximity clouds: Making mobile devices to collaborate for resource and data sharing," *arXiv preprint arXiv:1612.02468*, 2016.

[21] X. Lin, Y. Wang, Q. Xie, and M. Pedram, "Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 175–186, 2015.