HMCC: A Hybrid Mobile Cloud Computing Framework Exploiting Heterogeneous Resources

Zohreh Sanaei[1], Saeid Abolfazli[1], Abdullah Gani[2], Min Chen[3]
YTL Communications and Xchanging, Malaysia[1]
Center for Mobile Cloud Computing, University of Malaya, Malaysia[2]
Embedded and Pervasive Computing Lab, School of Computer Science and Technology,
Huazhong University of Science and Technology, China[3]
sanaei, abolfazli, abdullahgani, minchen@ieee.org

***Abstract*—Hybrid Mobile Cloud Computing (HMCC) refers to a Mobile Computation Outsourcing (MCO) model that exploits hybrid granular cloud-based resources composed of coarse-, medium-, and fine-grained resources interconnected by wireless and wired networks to augment mobile devices. Leveraging single type of granules for augmentation (i.e., vertically heterogeneous) has its own deficiencies of low proximity or/and scalability that leads to communication or/and computation latency. Therefore, responsiveness and energy efficiency of cloud-connected Compute-intensive Mobile Applications (CiMA) are degraded. In this paper, we aim to enhance energy-time efficiency of executing CiMA using HMCC. Performance evaluation results show significant gains, 80%-96% round-trip time and 83%-96% energy saving when executing CiMA using HMCC.**

I. Introduction

Recently, Mobile Computation Outsourcing (MCO) [1], [2] as a software mobile augmentation technique is emerged as a hot research topic with aim at augmenting capability of mobile devices by leveraging Granular Cloud-based Resources (GCRs) composed of Coarse-, Medium-, and Fine-grained Resources (CgRs, MgRs, and FgRs) interconnecting by wireless and wired networks, offering Mobile Cloud Computing (MCC) era. MCC technologies are emerging to overcome the computing, storage, and power limitation of mobile devices [3] toward rich mobile applications [2].

Considering a set of GCRs, we categorize them into coarse, medium, and fine resources, each with unique attributes of scalability and proximity, in particular. The GCR triple-set forms a pool of multi-level hybrid granular resources (see figure 1) populated with granules of relatively similar scalability, proximity, and multiplicity or other similar features. In this study, we consider scalability and proximity only. Scalability is ability to scale up to provide computing requirements and proximity indicates that how proximate is a resource to the mobile user (network distance).

CgRs are Virtual Machine (VM) instances of giant clouds, which feature high scalability and low proximity (giant clouds are very few in number located far from most of users) that leads to communication latency. MgRs are wall-connected cloudlets[4] featuring medium scalability and proximity that leads to communication and computation latency. FgRs are wirelessly connected battery-operating computing devices, including smartphones and tablets with high proximity and low scalability originating computation latency. Such communication and computation latencies negatively impact on time and energy consumption of CiMA leading to performance degradation problem. Although several MCC research efforts studied in [2] aimed to alleviate resource-deficiency of mobile devices using single type of GCRs, the prototype performance of HMCC utilizing heterogeneous cloud-based granules is an open research area.

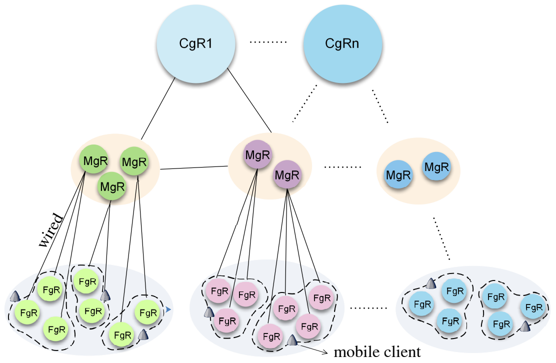


Fig. 1: Abstract view of Hierarchical granular cloud-based resources

We contribute by classifying MCO approaches, proposing a HMCC framework based on Horizontally Heterogeneous MCO (H^{2}MCO) with triple heterogeneous granules, prototyping the proposed framework based on RESTful Service-Oriented Architecture (SOA), and evaluating the HMCC performance. The remainder of paper is organized as follows. Section II presents our HMCC framework and design concerns regarding its light weight characteristic to achieve CiMA execution efficiency for resource-constraint mobile devices. Section III describes prototype implementation of the HMCC and its performance evaluation by benchmarking and statistical modelling. Result of HMCC performance evaluation and validation is explained in Section IV and paper is concluded in Section V.

A. Related Works and Background

The idea of augmenting capability of mobile devices by outsourcing computation-intensive task to the surrounded servers, clusters, or grids is not new. All are aiming to achieve efficient mobile application execution towards rich mobile applications. Clouds are another advanced outsourcing resources that could strongly facilitate mobile computation augmentation from different perspectives of storage, computation, safety, and data availability at anytime, anywhere. Cloud-based mobile augmentation is the-state-of-the-art MCO "that leverages cloud computing technologies and principles to increase, enhance, and optimize computing capabilities of mobile devices by executing resource-intensive mobile application components in the resource-rich cloud-based resources" [2].Recently, several MCO approaches emerged for MCC that we classify them into two categories of (i) vertically and (ii) horizontally heterogeneous, based on employed architectures that utilize granular cloud-based resources.

1) Vertically Heterogeneous Mobile Computation Outsourcing (VHMCO): It is referred to MCO approaches that use vertically heterogeneous cloud-based resources with only one type of resource granularity. For instance, a single type of cloud VM instances is utilized. Researchers in recent works [5], [4], [6] have leveraged three main types of vertically heterogeneous cloud-based resources, including CgRs, MgRs, and FgRs that can perform heterogeneous computations. Therefore, we classify existing works in three classes of coarse, medium, and fine granular approaches described below.

- ***Coarse Granular:*** In typical CMA approaches [5], [7], [8], VM-based distant clouds are leveraged as CgRs. The advantages of using CgRs in MCC (i.e., giant clouds) is high availability, scalability, elasticity, reliability, and security that make them suitable resources for extremely intensive computational task. However, leveraging these resources from mobile devices is encumbered by long WAN latency [4], [9]. Due to low multiplicity, CgRs are often far from users and accessing them via WAN through Internet is time-, energy-, and money-intensive leading to CiMA execution deficiency. MCO architectures using CgRs are more applicable for substantially computation-intensive tasks which are not time-sensitive so that communication latency does not impact much on user experience. However, this architecture is not advisable for data-intensive and communication-intensive applications [7]. Amazon EC2 and Google App Engine are two examples of coarse-grained resource providers.
- ***Medium Granular:*** MgRs are resources located in nearer location to mobile users compared to the CgRs, but are not as near as FgRs. They feature medium scalability and elasticity more than FgRs but less than CgRs. Number of MgRs is more than CgRs, they are located in more geographical regions, and their computing powers is medium. Cloudlet [4] is an effort that exploited computing power of VM-based proximate public immobile computers to perform intensive computation with less communication latency. However, using only MgRs in Cloudlets causes lack of mobility, limited availability, and secure threats, especially due to decentralized architecture [1], [2]. The advantages of using MgRs are lower WAN latency and higher multiplicity compared to other granules.Usually MgRs are suitable for moderate computation-intensive tasks with moderate time sensitivity.
- ***Fine Granular:*** FgRs are located in mobile user proximity and feature high proximity but low scalability and elasticity (e.g., smartphones, tablets, and laptops). Multiplicity of FgRs is significantly higher than CgRs and is expected to grow in the presence of insatiably popularity of mobile devices. They are located almost everywhere, but their computing powers is very limited. Researchers [8], [6] use nearby smartphones as remote FgRs for outsourcing in MCC. The advantages of using FgRs are their high multiplicity and proximity building a scalable cloud of proximate mobile devices with low WAN latency. Their high proximity to mobile users make them suitable resources for extremely time- and delay-sensitive small interactive applications.

2) Horizontally Heterogeneous Mobile Computation Outsourcing (H^{2}MCO): Unlike VHMCO that one type of resource granules is selected, in H^{2}MCO, resources are composed of multiple granules. For instance, in a H^{2}MCO solution, VM instances of Amazon EC2, cloudlets, and mobile devices are used which are placed in triple-level layers with varied functional and non-functional characteristics. Mobile applications built based on H^{2}MCO can take the benefits of all three classes of granularities with a communication-computation trade-off to gain better performance.

- ***Hybrid:*** Hybrid MCO is a feasible MCO architecture in which computing resources are combination of CgRs, MgRs, and FgRs. Hybrid resources are classified in category of H^{2}MCO since these resources in contrast to VHMCO, use multi-tier heterogeneous granules. MapCloud [10] is an effort using 2-tiered resources and offloading solution to reduce the time and price cost of offloading computation-intensive tasks to remote outsources. Finally, hybrid resources are aimed to overcome limited scalability and proximity capabilities of VHMCO which resulted communication and computation latencies to reduce response time and energy consumption of CiMA. Therefore, to advance cloud computing platform for mobile devices, there is a need for HMCC composed of varied granular resources.

II. Hybrid MCC (HMCC) Design

HMCC framework and its significance for augmenting computing capabilities of mobile devices are presented below.

A. Framework

The framework features three major building blocks (mobile service requester, system arbitrator, and horizontally heterogeneous service provider) depicted in Fig. 2. Service providers' resources are horizontally heterogeneous hybrid granular computing entities that feature varied scalability and proximity levels. Thus, efficiency of service delivery for mobile users with varied quality requirements is improved. We classify the resources based on three proximity levels of low as country-level, medium as city-level, and fine as cell-level, and also, three scalability levels of low, medium, and high. These resources can be customized for resource consumers of international and national chained companies, universities, and hospitals with varied proximity and geographical distribution.

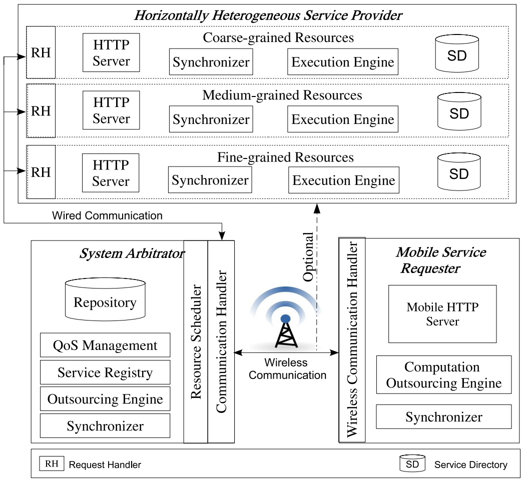


Fig. 2: Schematic presentation of our framework

Also, the framework has flexibility of directly/indirectly connecting users to GCRs upon user preference. Some of the key features of HMCC framework are briefed below.

• **Lightweight computation-intensive Mobile Application:** 'Lightweight' refers to the low temporal and energy overheads of utilizing heterogeneous GCRs. Employing SOA in designing the framework reduces application development burden, cloud services maintenance, and application portability to varied platforms such as Android and iOS. The service-based development generates loosely coupling of resource-intensive services with the rest of application. So, overhead of identifying, partitioning, and offloading application to remote resources are omitted and hence the application execution originates less temporal and energy costs.

• **Portability** is a significant feature of HMCC which is inherited from SOA. The framework is both vertically and horizontally platform-independent [11]. The former ensures that applications built based on this framework can be executed on various version of a certain operating systems, such as Android. The latter means that applications built based on this framework can be executed on, and ported to varied mobile operating systems without considerable reconfiguration and modification.

• **Centralized Architecture:** Considering implications of using ad-hoc and peer-to-peer architectures, utilizing centralized architecture remarkably improves the complexity and management of MCO. Transferring the complexity of interacting with varied service providers to arbitrator improves management, complexity, and efficiency by omiting the overhead from the mobile service requester; otherwise the overhead of service provider discovery by mobile service requester could neutralize the MCO performance gain.

• **Seamless Execution:** Deploying arbitrator functionalities in the MNO, as a central trustworthy entity, which manages the entire outsourcing process facilitates seamlessness. The arbitrator tracks all cloud-based resources, connected clients, and mediates the entire outsourcing platform to ensure remote execution of jobs seamlessly takes place with low overhead.

• **Enriched User Interaction:** We employed asynchronous communication technology where mobile-cloud communications take place in the background without freezing the application or mobile client device. Utilizing asynchronous communication omits interaction distraction and remarkably enhances user interaction experience. While outsourcing process is executing in the background, mobile user can fully utilize the features of the device and applications with no distraction.

III. Performance Evaluation

The performance of framework is evaluated via benchmarking experiments on real testbed, using 30 synthetic workloads. Benchmarking results are validated via statistical modelling. Application Round-Trip Time (RTT (ms)) and Energy Consumption (EC (mJ)) are two metrics in evaluating the performance of the MCC outsourcing systems. Power Tutor 1.4 and auto logger are used for energy and time profiling, respectively.

A. Benchmarking

We used HTC Nexus One smartphone with 1 GHz CPU and 512 MB RAM running Android OS and Cisco Linksys WRT54G for wireless communication. The CgR is a compute optimized c1.xlarge VM instance of the Amazon EC2 located in Singapore with 8 vCPU featuring 20 elastic computing unit, 7 GB RAM and high performance I/O. The MgR is a desktop with Intel quad core 3.3 GHz CPU and 8 GB RAM. The FgR is a Dell Laptop XPS14z featuring quad core 2.5 GHz CPU and 4GB RAM. The prototyped application is composed of three computation-intensive tasks, namely factorial, x power y, and prime generator services.

B. Statistical Modelling

Techniques used to provide statistical modelling of analysing RTT of application (computing and communication time) are discussed in two modes of local and hybrid. We produce the statistical model using independent replication model to train the regression model. The statistical model is validated using split-sample model. We use partial dataset to build the model and the rest to validate the model. The statistical model is applied to the subsets of main dataset and the results are used to validate the model.

1) Local Mode:

• *Local Round-Trip Time ($LRTT$):* is the total time consumed to execute all computation-intensive services in mobile devices. Therefore, for each workload i, the $LRTT$ is,

$$LRTT_i = RTT_{fc_i} + RTT_{pw_i} + RTT_{pr_i} \tag{1}$$

where RTT_{fc}, RTT_{pw}, and RTT_{pr} are execution time of factorial, power, and prime services, respectively. Therefore, first we formulate expected time for each of them. These three algorithms have different time complexity. So, to characterize growth rate of different functions according to their workloads, the Big O notation is used. Therefore, a unique equation for each function identifies upper and lower bounds along with constant and coefficient values related to specific device that runs the algorithm. To evaluate the execution time and energy consumption of the mobile application, we leverage observation-based prediction method using supervised regression as the most common approach. In the following we present the result of regression analysis and RTT equation for each services.

Factorial's Algorithm: The factorial algorithm is classified as quadratic algorithm (Factorial complexity is $O(f^2)$) and its round-trip time Equation (RTT_{fc}) is a quadratic equation as

$$RTT_{fc_i} = Af_i^2 + Bf_i + C \tag{2}$$

where, f is the workload and RTT_{fc} is the total time in (ms) to execute factorial service on mobile devices. To identify the coefficients and constant values, we analyze the growth rate of execution time and its correlation with workloads using curve estimation regression. The summary results of regression and Anova test presented in Table I advocate that the quadratic regression model is as accurate as %99 and is fitted into the expected equation and time complexity.

TABLE I: The factorial's regression model result in local mode

Model Summary			Parameter Estimates		
R Square	F	Sig.	Constant	b1	b2
0 .99	1776.96	0.00	1013.97	-0.98	7.09E-4

The $R\ Square$, F and $Sig.$ values in the Table I show significant direct correlations between the workloads and their corresponding execution times. So, by replacing the coefficient A, B with $b2, b1$, respectively and the constant value of C in Equation (2) with constant value from Table I, we have,

$$RTT_{fc_i} = 7.09E - 4f_i^2 - 0.978f_i + 1013.97 \tag{3}$$

In order to validate the devised model for factorial, we perform split-sample validation model that successfully demonstrates validity of the devised model based on the results of the analysis reported in Table II. The results show strong correlations between the factorial workloads and the execution time in all the three cases. Since the difference between the R^2 values of split samples and full sample are less than 5%, it can be concluded that the proposed model remains valid. For the sake of brevity for two other algorithms, we only bring the generated models.

TABLE II: Factorial's Split-sample validation results in local mode

Metrics	split = 1.00 (Selected)	split = 0.00 (Selected)	Non-split Sample
R	1	1	1
R^2	1	1	1
Adjusted R^2	1	1	1
df	15	13	29

Power's Algorithm: Power algorithm has the time complexity of $O(p^3)$. Thus, the RTT of power function (RTT_{pw}) is modelled as a cubic equation. The summary results of regression advocate that the cubic regression model is as accurate as %99 and is fitted into the following equation.

$$RTT_{pw_i} = 3.52E{-}9p_i^3{-}1.38E{-}4p_i^2{+}2.080p_i{-}6.98E3 \tag{4}$$

Prime's Algorithm: Analyzing this algorithm results the time complexity of $O(r)$. Here, we use linear estimation regression to identify the correlation and constant values. The summary results of linear regression validate that the model is as accurate as %98 and is fitted into the linear equation.

$$RTT_{pr_i} = (5.39E - 3)r_i + 111.25 \tag{5}$$

Finally, the total RTT by substituting the equations will be,

$$\begin{aligned} LRTT_i = {} & (7.09E - 4f_i^2 - 0.978f_i) \\ & + (3.52E - 9p_i^3 - 1.38E - 4p_i^2 + 2.080p_i) \\ & + (5.39E - 3r_i) + 5.96E3 \end{aligned} \tag{6}$$

• *Local Energy Consumption (LEC):* The major energy consumer in local execution mode is CPU and LCD. In this study we consider only CPU usage which is completely dependent to processing time of services running on mobile devices. Similar to time, the mean energy consumed by CPU per millisecond (TEC_{cpu}) is modelled by linear regression analysis. The summary results show that the regression model is as accurate as %99, and CPU power consumption per millisecond estimation is 0.34 (mW). So,

$$LEC_i = LRTT_i \times 0.344 \tag{7}$$

2) Hybrid Mode:

• *Hybrid Round-Trip Time (HRTT):* The $HRTT$ for executing the CiMA is calculated via considering the arbitration, communication overhead, and maximum time needs to process three services of factorial, power, and prime in remote resources. In hybrid mode, mobile device calls arbitrator and sends along the name of desired services asking to find appropriate resources for remote execution; arbitrator use its scheduler component to find the IP address(es) of the appropriate service providers able to perform mobile device task; one found, the scheduler return results to the arbitrator. Then, arbitrator makes asynchronous calls to the identified remote resources along with data asking for execution. Once the results are completed and received to the arbitrator, it forwards them to the mobile device. Therefore, for calculation of HRTT we need to calculate the Arbitrator Time (AT), Remote Execution Time (RET), and the total Communication Time (CT). So,

$$HRTT_i = AT_i + RET_i + CT_i \tag{8}$$

whereas, the scheduling time is independent of workloads, then we consider only one mean value for all workloads. Also, its value highly depends on the wireless network and performance of computing device that hosted the system arbitrator. Therefore, we replace AT_i with AT. In order to identify the mean value of system arbitrator, we measured the arbitrator delays. The calculated mean value is 1041.39 (ms). For the sake of simplicity, we round the delay to 1041 (ms) per call. Also, as the remote resources are multilayer, it is beneficial to perform asynchronous calls from the system arbitrator, so that remote executions perform in parallel. Therefore, RET_i equals to the execution time of the longest service. In this study, the maximum execution time between all services belongs to power function. So, $RET_i = RET_{pw_i}$.

Here, we follow the same approach used in local mode and leverage observation-based prediction method using supervised regression model. Also, our prototype is computation-intensive so the communication volume in our experiment is fixed and does not grow by workload increase. Hence, there is no correlation between the CT_i and workloads. This delay depends only on the quality of communication medium; the medium is fixed for all workloads and executions. In order to estimate its value, we observed communication delay of our dataset and calculated its mean value as $134.41(ms)$ that is rounded to $134(ms)$. Hence, we have:

$$\begin{aligned} HRTT_i &= 1041 + (-2.59E-011)p_i^3 + (1.79E-006)p_i^2 \\ &+ 17.56 + 134 = (-2.59E-011)p_i^3 \\ &+ (1.79E-006)p_i^2 + 1192.56 \end{aligned} \tag{9}$$

• *Hybrid Energy Consumption (HEC):* To evaluate HEC, we only consider energy usage by CPU and WiFi data transmission without LCD consideration. Whereas, the tested application was focused on intensive computation so the uplink-bytes and downlink-bytes are always low (at most 1163 Bytes) in this study. Thus, the WiFi state is low and based on the literature and our experiment, the particular mobile device used in this experiment consumes 0.034 (mW/ms) power as long as is connected to WiFi. Then, $HEC_i = TEC_{cpu_i} + TEC_{w_i}$ where, The TEC_{cpu} and TEC_w are the total energy consumption by CPU and WiFi, respectively.

By replacing, $TEC_{w_i} = HRTT_i \times 0.034$ and $TEC_{cpu_i} = (HRTT_i \times 0.328) - 94.10$ we have:

$$\begin{aligned} HEC_i &= (HRTT_i \times 0.328 - 94.10) + (HRTT_i \times 0.034) \\ &= HRTT_i \times 0.362 - 94.10 \end{aligned} \tag{10}$$

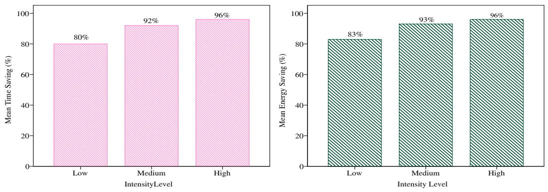
IV. Results and Discussion

Results are synthesized for local and hybrid in three intensity levels and the evaluation results are validated via statistical modelling.

A. Round-Trip Time (RTT)

Based on measured results the hybrid execution mode can reduce execution time of computation-intensive mobile application in average of 80%, and 92% to 96%, in low, medium, and high intensity levels, respectively, compare with local execution mode (see Fig. 3a).

In local mode, the mean RTT is 6370.85 (ms) ($\simeq$ 6s). With increase of workloads in medium and high intensity levels, the RTT significantly grows. The low-medium and low-high RTT differences are as high as 12250.10 (ms) ($\simeq$ 12s) and 35524.09 (ms) ($\simeq$ 36s). By contrast, in hybrid mode when the workload is low, the mean RTT is 1284.35 (ms) ($\simeq$ 1s). Unlike local execution mode, in hybrid mode increase in workload intensity, causes insignificant rise in RTT values. Low-medium and low-high RTT differences are as low as 138.50 (ms) ($\simeq$ 0.1s) and 580.21 (ms) ($\simeq$ 0.6s) in second and third workloads intensity categories, receptively. Fig. 4 compares the measured RTT for 30 workloads in local and hybrid modes. It is noticable that despite of scheduling engine in arbitrator and data transmission through wireless communication, increase in workload intensity in hybrid mode has low impact on execution time growth of the CMA, using high computing processors with large storage and memory. So, HMCC is a time-efficient platform for CMA compared with local application execution. The higher compute-intensity of workloads, the greater is the time saving in hybrid mode. Fig. 5 depicts average computation and communication time of intensive services on coarse-, medium-, and fine-grained resources. Bars and their segmented areas clearly demonstrate computation-communication trade-off when leveraging heterogeneous granular resources for low, medium, and high intensity workloads. The highest communication overhead belongs to distant resource stated at the coarse level, while the rest of resources have very low communication overhead. By distributing computation-intensive tasks to different resources we can significantly reduce the WAN latency in all three services. However, computing time of services running on medium- and fine-grained resources slightly is increased compared to the coarse-grained resources. Therefore, leveraging heterogeneous GCRs in HMCC can enhance responsiveness toward optimal execution of computation-intensive applications on mobile devices.



(a) Time saving average (b) Energy saving average

Fig. 3: Hybrid mode

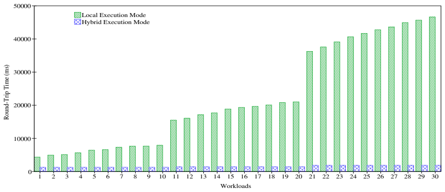


Fig. 4: RTT comparison: Local vs Hybrid

B. Energy Consumption (EC)

HMCC can save energy in average of 83% to 96% in three intensity levels. Fig. 3b indicates that energy saving is increased about 13% from low to high workloads. In local mode, mean EC for the low workloads is 2180.96 mJ ($\simeq$ 2J), while with increase of workloads, the EC raises 4120.59 mJ ($\simeq$ 4J) and 12106.07 mJ ($\simeq$ 12J) in medium and high intensity level, receptively. By contrast, in hybrid mode when the workload is low, the mean EC is 360.88 mJ ($\simeq$ 0.4J $\ll$ 2J), while with increase of workloads, the EC raises only

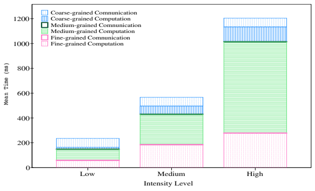


Fig. 5: Comp-comm trade-off

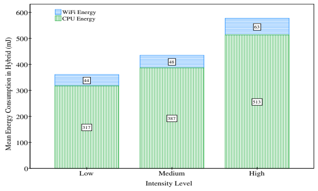


Fig. 6: EC mean:WiFi and CPU usage

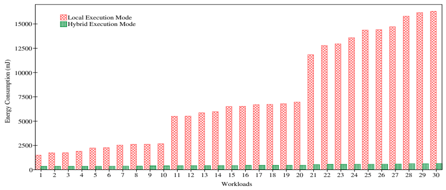


Fig. 7: EC comparison: Local mode vs Hybrid mode

74.01 mJ ($\simeq$ 0.07J $\ll$ 4J) and 215.84 mJ ($\simeq$ 0.2J $\ll$ 12J) in second and third workloads' category, receptively. It means that, the EC of hybrid in all intensity levels is less than 1 J which has significant difference with local mode where energy values vary from 2 J to 14 J. Therefore, these results unveil the advantage of utilizing H^{2}MCC for CMA.

Fig. 7 demonstrates the average results of measured energy consumption in local and hybrid modes. Although in hybrid mode, some energy are dissipated for communication, the high processing power of servers and distributing jobs between heterogeneous service providers noticeably prevent battery power dissipation in mobile devices. Fig. 6 depicts the average energy consumed by WiFi and CPU in our tested.

C. Discussion

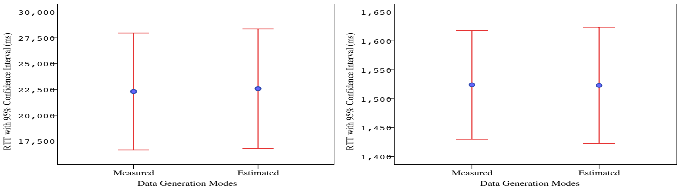
Analytical comparision validates the performance gains of HMCC. The Error bar charts 8 and 9 related to local and hybrid RTT and EC generated by these two modelling with 95% Confidence Interval (CI). As shown, the CI range of benchmarking and statistical modelling has significant overlap that proves a negligible difference in RTT (Figs. 8a and 8b) as well as EC (Figs. 9a and 9b). The small differences in mean values and CI testifies validity of performance evaluation.

V. Conclusions

In this paper, proposed HMCC framework aims to leverage heterogeneous resources with varied scalability and proximity that can be allocated to varied computation- and delay-sensitive tasks while performing a communication-computation trade-off. The evaluation results advocate efficient computation outsourcing for computation-intensive mobile applications with 80%-96% responsiveness and 83%-96% energy efficiency. The results show that the more computation-intensive applications, the more energy-time efficient application of HMCC compared with local mode.

Acknowledgment

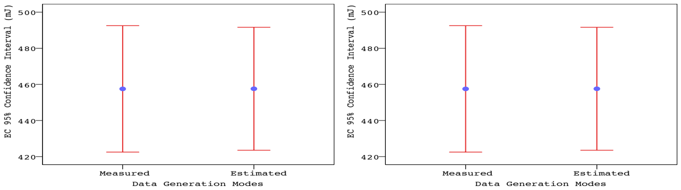
This work was funded by the Malaysian Ministry of Higher Education under the University of Malaya High Impact Research Grant UM.C/HIR/MOHE/FCSIT/03. Prof. Min Chen's work was supported in part by China National Natural Science Foundation under Grants 61300224, the International Science and Technology Collaboration Program (2014DFT10070) funded by China Ministry of Science and Technology (MOST), Hubei Provincial Key Project under grant 2013CFA051, and Program for New Century Excellent Talents in University (NCET).



(a) Local Mode

(b) Hybrid Mode

Fig. 8: RTT mean with 95% CI



(a) Local Mode

(b) Hybrid Mode

Fig. 9: Mean EC with 95% CI

References

[1] M. Sharifi, S. Kafaie, and O. Kashefi, "A Survey and Taxonomy of Cyber Foraging of Mobile Devices," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 1232–1243, 2011.

[2] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-Based Augmentation for Mobile Devices: Motivation, Taxonomies, and Open Challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 337–368, 2014.

[3] V. Leung, M. Chen, M. Guizani, and B. Vucetic, "Cloud-Assisted Mobile Computing and Pervasive Services," *IEEE Network*, vol. 27, no. 5, pp. 4–5, 2013.

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[5] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: making smartphones last longer with code offload," in *ACM MobiSys'10*, CA, USA, 2010, pp. 49–62.

[6] S. Abolfazli, Z. Sanaei, A. Gani, F. Xia, and W. M. Lin, "RMCC: A Restful mobile cloud computing framework for exploiting adjacent Service-based Mobile Cloudlets," in *IEEE CloudCom'14*, 2014.

[7] K. Kumar and Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[8] E. E. Marinelli, "Hyrax: Cloud computing on mobile devices using MapReduce," Master Thesis, Carnegie Mellon University, 2009.

[9] S. Abolfazli, Z. Sanaei, M. Alizadeh, A. Gani, and F. Xia, "An experimental analysis on cloud-based mobile augmentation in mobile cloud computing," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 1, pp. 146–154, Feb. 2014.

[10] M. R. Rahimi, N. Venkatasubramanian, S. Mehrotra, and A. V. Vasilakos, "MAPCloud: Mobile Applications on an Elastic and Scalable 2-Tier Cloud Architecture," in *Proc. IEEE/ACM UCC'12*, Chicago, USA, 2012, pp. 83–90.

[11] Z. Sanaei, S. Abolfazli, A. Gani, and R. K. Buyya, "Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 369–392, 2014.