DOI: http://dx.doi.org/10.26483/ijarcs.v9i2.5547

ISSN No. 0976-5697

Volume 9, No. 2, March-April 2018

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

CONTEXTUAL COMPUTATION AUGMENTATION IN MOBILE CLOUD COMPUTING

Jitender Kumar
Department of Computer Science and Engineering
DCRUST Murthal
Sonipat, India

Amita Malik
Department of Computer Science and Engineering
DCRUST Murthal
Sonipat, India

Abstract: Mobile cloud computing has emerged as a promising solution for augmenting the cognitive capabilities of the resource constraint smart mobile devices. Although a lot of efforts has been made for augmenting the computational capabilities of these hand-held smart devices, but there are significant problems which need to addressed e.g. enabling the individual mobile users to contextually tune their computing requirements and efficiently utilizing cloud resources while offering the quality of service (QoS). This paper proposes a context based computation augmentation framework to balance these competing goals. It incorporates individual contextual demands in its optimization engine to dynamically adjust its offloading decisions. In addition, it also proposes an adaptive virtual machine (VM) provisioning technique for efficient utilization of cloud resources and maintaining the QoS. We evaluated the design space of the proposed system by extending CloudSim simulator. The experimental results show that the proposed system is able to provide contextual augmentation services with QoS and low costs.

Keywords: Mobile Cloud Computing; Contextual computation augmentation; Quality of service, Virtual machines.

INTRODUCTION

In recent years, smart mobile devices with sophisticated operating systems and advanced hardware features have rejuvenated the computing world. These devices have become the primary computing entities for the day-to-day activities of users [1]. Despite the noticeable improvements in hardware and software technology, these hand-held smart devices pose certain limitations which are not going to be mitigated by making more powerful devices [2]. Due to limited processing capabilities, these devices are not suitable for carrying out compute intensive tasks.

Mobile cloud computing has been introduced to empower the processing capabilities of these devices [3]. It is a combination of two different computing paradigms namely: mobile computing and cloud computing. Mobile computing enables the users to seamlessly carry out computing tasks regardless of their locations and mobility [4]. However, cloud computing is a mode of enabling the users to access the shared pool of remote computing resources (e.g. storage, virtualized computing resources etc.) through the internet. These computing resources can be acquired in on-demand way with least management efforts [5]. Users just need to pay-per-use for the use cloud services [6].

The research community has proposed a number of efficient mobile augmentation frameworks for augmenting the cognitive capabilities of mobile devices [2, 7-14]. These frameworks have primarily focused on extensive augmentation of mobile device capabilities in terms of time and energy. However, cloud resources are available on pay-per-use basis [6]. So, there is an essential requirement for systems which can enable mobile clients to contextually acclimatize their cloud resource demands. Additionally, the current mobile device augmentations frameworks are either based on static threshold-rules [10,11] or employ minimum resource provisioning schemes [15] for efficiently utilizing the cloud resources. The static threshold-based systems are application specific [16] and require strict estimation of lower and upper thresholds. Whereas, the minimum resource provisioning schemes although proves to be resource efficient but result in higher queuing delays and higher service level agreement (SLA) violations.

This paper proposes a three tier framework for resolving the above-discussed issues. It exploits the waiting time in the queuing system as a QoS metric. The waiting time can be negotiated in the service level agreements (SLAs). The goal of the proposed framework is to satisfy the individual mobile user's contextual requirements with QoS and efficiently utilize cloud resources with lower leasing costs. The key contributions of the proposed system are: 1) it provides a mechanism for the context based application partitioning and 2) it also proposes an adaptive resource provisioning technique which also maintains the QoS. We have implemented a prototype application for Android and conducted the performance evaluation of the proposed system by extending the CloudSim [17] simulator. The experimental results of the proposed system are compared with two base line systems namely: 1) minimum resource provisioning systems [16] and 2) over-provisioning systems [9]. In all these experiments, the proposed system results in good computation speedup compared with minimum and over-provisioning based systems. Additionally, the proposed system also utilizes the cloud resources efficiently and incurs lower leasing costs while maintaining the QoS.

The remainder of this paper is organized as follows: Section 2 covers the related work, Section 3 introduces the optimization framework for the contextual augmentation of mobile devices, Section 4 presents the performance evaluation of the proposed system and section 5 discusses the conclusion.

RELATED WORK

Several approaches have been proposed for the augmentation of the cognitive capabilities of the mobile devices. Some of them focused on reducing the execution times [7-10, 13, 15], whereas others have focused on energy savings [7-9, 12-14]. Kemp et al. [7] proposed a middleware

system to improve the responsiveness and avoid the vendor lock-in. Cuervo et al. [12] introduced MAUI architecture which reduced the burden on the application developers. MAUI proposed an optimization engine for the dynamic application partitioning without developer intervention. Chun et al. [9] proposed a compiler which can automate the application partitioning process without re-designing an application from scratch. Ferber et al. [13] utilized a scaling algorithm which maintains 10% extra virtual machines in all circumstances and assumed that the mobile application just consist a single computational intensive part. Whereas, the architecture proposed by Kosta et al. [8] focuses on parallelizing the execution of mobile application by keeping more than one device clones at the cloud side.

However, all these approaches have ignored the contextual requirements of users and don't allow flexible use of cloud resources. The proposed system extends the existing frameworks by allowing the users to dynamically tune their computation speedup requirements for saving prices.

The proposed system incorporates many ideas from the previous computation augmentation systems, for instance, it incorporates programming model similar to [14]. It also delivers the expected quality of service by automatically adjusting the number of VMs according to the arriving workloads. Indifferent from the static threshold based systems, it does not require application knowledge to manually adjust the upper and lower thresholds of [10,11]. However, the resource provisioning scheme of the proposed system is also inspired from the study of Rodrigo et al. [18] which considers SLA negotiated response time as a quality of service metric. But this work considers queues with balking and requires the complete application execution at remote VMs within deadlines, whereas, in the proposed system application tasks may execute locally or on cloud VMs with no balking.

PROBLEM STAEMENT

The key objective of the proposed system is to let users to adjust their computational requirements and adaptive provisioning of VMs for efficiently utilizing these cloud resources along-with maintaining the QoS. It enables the users to contextually acclimatize according to their requirements by following mathematical formulation. Let the sample mobile application is a set of tasks K = {$C_1, C_2 \ldots C_n$}. Each task is denoted by a 4-tuple, $C_i = \left\{d_i^{in}, wl_i^{MI}, d_i^{out}, x_i\right\}$ where d_i^{in} and d_i^{out} denote input and output data of the i^{th} task. Whereas, wl_i^{MI} represents the workload corresponding to the i^{th} task. The variable x_i denotes the annotation of the component for remote VM execution ($x_i = 1$) or locally on the mobile device ($x_i = 0$). A smart mobile device is modeled as a 2-tuple $M = \left\{S_F^m, \delta_R^m\right\}$, where, S_F^m represents the clock frequency of the hand-held smart mobile device (in GHz). Whereas, δ_R^m denotes the user's arbitrarily chosen contextual requirements varying between 0 to 100 i.e. $\delta_R^m \varepsilon [0,100]$. The value $\delta_R^m \to 0$, implies that all components will be executed locally and the cloud computation cost will be 0. When $\delta_R^m \to 100$, the computation speedup will approach to optimal values similar to CloneCloud [9]. A cloud VM is represented as a 2-tuple, $VM = \left\{S_F^{Vm}, P_{LP}^{Vm}\right\}$, where, S_F^{Vm} denotes clock frequency of remote virtual machine (in GHz) and P_{LP}^{Vm} denotes the per hour price of a cloud VM. Let T_i^L , and T_i^{Vm} denotes the corresponding execution time of the i^{th} task locally and on cloud VMs, whereas, T_i^{Tr} denotes the time for sending and receiving the associated data of the i^{th} task. If the remote execution time of a task is less than its local execution time, then a task can be offloaded to the remote VMs. Let T_i^S be the difference between the local execution time and the remote execution time for a task, and V_i^R be its corresponding execution cost of i^{th} task on cloud VMs by considering per minute price of the cloud VMs and can be calculated using equation (1). Let the overall cost of cloud eligible components be represented by Z_m.

$$V_i^R = \left(T_i^{Tr} + T_i^{Vm}\right) Cost_{Min}^{Vm} \quad \ldots(1)$$

The objective of context based application partitioning process is to maximize computation speedup 'ψ', for the used specified requirement, δ_R^m, and is being formulated as:

$$\Psi = \max \sum_{i=1}^{n} T_i^S x_i \quad \ldots(2)$$

Such that

$$\sum_{i=1}^{n} V_i^R x_i \le \frac{Z_m \delta_R^m}{100} \quad \ldots(3)$$

The resulting solution $(x_1, x_2, \ldots x_n)$ is the optimized partitioning for the application. The concept is similar to CRM where users are allowed different access levels. However, the second objective of the proposed system is to maintain VM instances such that:

$$P\left\{w_q > SNT\right\} < (1 - SL/100) \quad \ldots(4)$$

Where, "*SNT*" and "*SL*" represents waiting time negotiated in the SLAs and the service level intended by the service provider, respectively. So, equation (4) represents that the probability of waiting time w_q greater than the SLA violation should be less than "*SL*" in the M/M/c queuing system. The corresponding algorithmic steps for adaptive VM provisioning are depicted in Algorithm 1. The algorithm determines how many VMs need to be provisioned at any time for the arriving workloads. It takes four arguments as input, namely: 1) mean request arrival rate (λ), 2) mean service rate (μ), 3) SLA negotiated waiting time (*SNT*), and 4) number of running servers (s).

EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

The hardware setup consist two entities: a dual core VM deployed in VMware and an HTC Android mobile device. The bubble sort application is used as the sample application for measuring the performance of the smart mobile device and VM. The execution time of the application on the smart mobile device and on the VM alone are depicted Fig. 1. As the single client-server application does not suffer any queuing delays, so, we extended CloudSim [17] simulator for analyzing the effect of queuing delays and cloud resource provisioning scheme. In the simulation model, we used one cloud datacenter with 100 hosts. The simulation parameters and corresponding values are summarized in Table 1.

Table: 1 Simulation Parameters		
Parameter Name	Value	Measuring Unit
SNT	5	Sec
SL	99	%
S_F^m	1.2	GHz
S_F^m	2.5	GHz
P_{LP}^{vm}	0.145	$

The new VM initiation takes 150 seconds to start [13]. We evaluated the proposed system in a stable data rate environment. However, the proposed system also penalizes the service provider for any delay higher than the "*SNT*". The penalty calculation model is similar to [19], and the penalties can be calculated as:

$$penalty = \sum_{t=1}^{h} \alpha + \theta \times t \quad ...(5)$$

Where "α" and "θ" are the fixed penalties and penalty rates, respectively. The penalty rate "θ" is kept equal to per minute price of the VM and value of "α" is not considered. The variable "*h*" denotes the number of SLA violations. The execution time is transformed in million instructions for CloudSim using:

$$wl_i^{MI} = T_i^{Vm} \times S_F^{Vm} \quad ... (6)$$

We compare the proposed system with two baseline systems:

Algorithm 1: Adaptive VM provisioning Scheme

Procedure: Instance_estimator(λ, μ, *SNT*, s)

1 *temp* = *s*; //running VMs
2 calculate traffic intensity "ρ"
3 if (ρ > 1)
4 then increase VMs until (ρ > 1)
5 else *s* = *s*/2;
6 while $P(w_q > SNT) > \left(1 - \frac{SL}{100}\right)$ do
7 s++;
8 end while
9 if(*s* > *temp*) then
10 start (*s* – *temp*) VMs;
11 end if
12 else
13 stop(*temp* – *s*) VMs
14 end else
15 end procedure

- **COSMOS [15]:** It is based on minimum VM provisioning approach i.e. it uses Algorithm 2. However, both algorithms differ in line# 6, the COSMOS system uses the line# 6 as $while(w_q > SNT)$.
- **OP:** It assumes that the number of VMs is provisioned according to the peak requests arrival rates, so that no SLA violation can occur like CloneCloud [9].

We evaluated the proposed system with the synthetic workload. The augmentation requests arrival follow Poisson distribution with mean 1.5 requests per minute during 8:00 AM to 4:00 PM, 1 request per minute during 4:00 PM to 12:00 AM, and 0.5 requests per minute during 12:00 AM to 8:00 AM, with overall total of 1432 application requests. All mobile applications contain 10 service tasks [12].

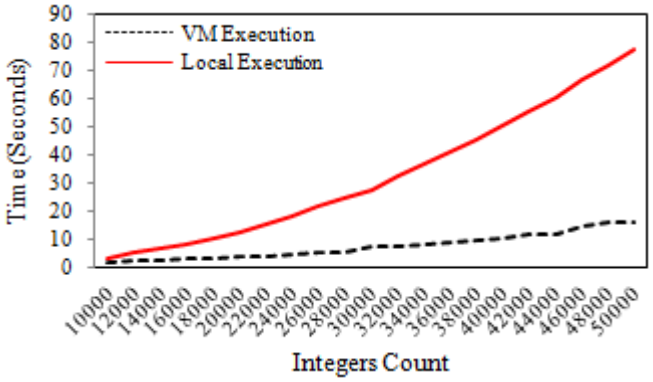

Fig. 1: Comparison of execution time of the application on local mobile device and VM.

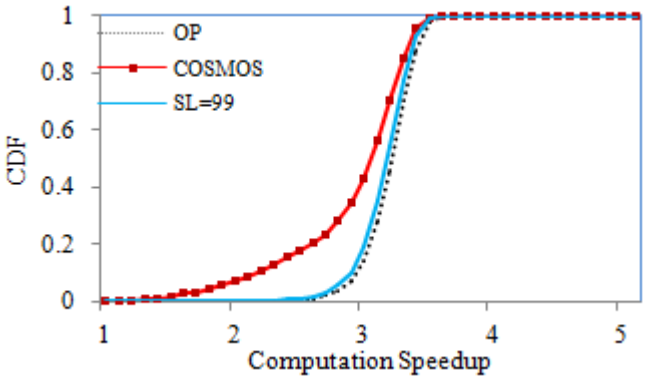

Fig. 2: Computation speedup for OP, COSMOS and the proposed system for SL=99%.

Algorithm 2: Minimum VM provisioning Scheme

Procedure: Instance_estimator(λ, μ, *SNT*, s)

16 *temp* = *s*; //running VMs
17 calculate traffic intensity "ρ"
18 if (ρ > 1)
19 then increase VMs until (ρ > 1)
20 else *s* = *s*/2;
21 while $(w_q > SNT)$ do
22 s++;
23 end while
24 if(*s* > *temp*) then
25 start (*s* – *temp*) VMs;
26 end if
27 else
28 stop(*temp* – *s*) VMs
29 end else
30 end procedure

For each scenario we measured the following metrics: Expected computation speedup, % VMs utilization rate, % SLA violations, and total monetary cost. The total monetary includes leasing costs as well as penalty costs. Expected computation speedup is the ratio of execution time of the mobile application locally and on cloud VM.

RESULTS AND DISCUSSIONS

When $\delta_R^m \to 100$, the system is equivalent to CloneCloud [9]. The expected speedup in computation for OP is 3.19X and for COSMOS is 2.9X, whereas, for the proposed system when $SL = 99\%$ the computation speedup is 3.15X (in Fig. 2). The total monetary cost of OP, COSMOS, and proposed system when $SL = 99\%$ are $24.36, $24.51, and $18.9, respectively (in Fig. 3). So, the OP system spends nearly 28.89% extra money for a mere 1.27% extra speedup. Whereas, COSMOS system spend nearly 29.69% extra money and results in 8.62% lesser speedup as compared to proposed system. Similarly, the % VMs utilization rate of OP, COSMOS and proposed system are 18.19%, 44.67%, and 24.29%, respectively (in Fig. 4). However, proposed system also suffers lesser SLA violation (0.27%) as compared to COSMOS (10.79%).

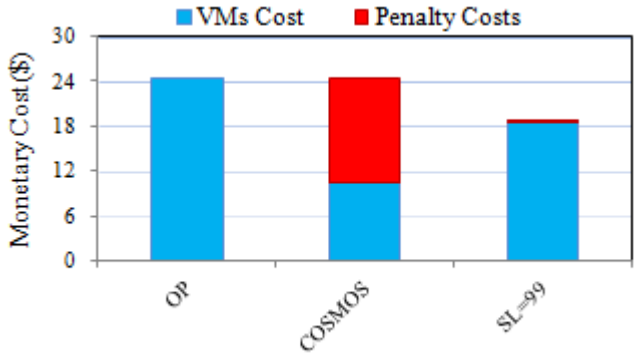

Fig. 3: Monetary costs comparison for OP, COSMOS, and the proposed system for SL=99%.

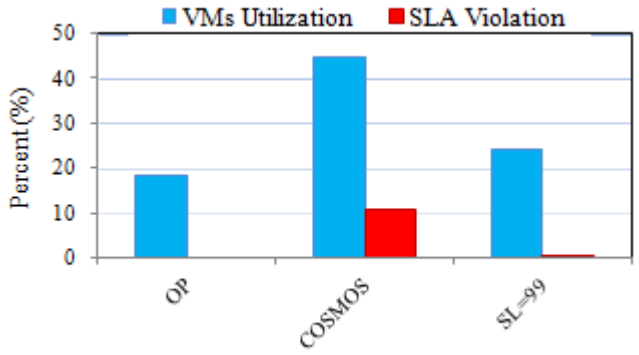

Fig. 4: Utilization and SLA violations comparisons for OP, COSMOS, and the proposed system for SL=99%.

CONCLUSION

This paper takes a step towards enabling individual mobile users to contextually acclimatize their computing demands and efficiently utilizing cloud VMs along-with maintaining the quality of service. It provides two key solutions for context based QoS provisioning: 1) it provides a mechanism for contextual requirements based partitioning of the application as cloud executable or local executable; 2) it also proposes an efficient cloud resource provisioning scheme. The experimental results show that the proposed system can provide context based services with lower monetary costs and quality.

REFERENCES

[1] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile Cloud Computing: A Survey," Future Generation Computing Systems, 2012, DOI: 10.1016/j.future.2012.05.023, pp. 85-106.

[2] M. Satyanaryan, P. Bahl, R. Caceres, and N. Davis, "The case for VM-Based Cloudlets in Mobile Computing," IEEE, Pervasive Computing, 2009, pp. 14-23.

[3] P. Bahl, R. Y. Han, L. E. Li and M. Satyanarayanan, "Advancing the State of Mobile Cloud Computing," *MCS'12*, June 25, 2012, Low Wood Bay, Lake District, UK.

[4] M. Satyanarayan, "pervasive Computing: Vision and Challenges," Personal Communications, IEEE, 8(4), pp. 10-17, 2001.

[5] P. Mell and T. Grance, "The NIST Definition of Cloud Computing (Draft),"http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

[6] M. Armburst, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," EECS Department, University of California, Berkeley, tech. Rep. UCB/EECS-2009-28, 2009.

[7] R. Kemp, N. Palmer, T. Keilmann, F.Seinstra, N. Drost, J. Maassen, and H. Bal, "eyeDentify: Multimedia Cyber Foraging from smartphones," Proceedings of 11th IEEE International Symposium on Multimedia, DOI 10.1109/ISM.2009.21, pp.392-399, 2009.

[8] S. Kosta, A. Aucinas, P. Hui, R. Mortier and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in cloud for mobile code offloading," INFOCOM, 2012 Proceedings IEEE 10.1109/INFCOM.2012.6195845 pp. 945-953.

[9] B. G. Chun, S. Ihm, P. Maiatis, M. Naik, and A. Patti. "CloneCloud: Elastic execution between mobile devices and Cloud," EuroSys, 2011, pp. 301-314.

[10] C. Mei, D. Taylor, C. Wang, A. Chandra, and J. Weissman, "Sharing-aware Cloud-based Mobile Outsourcing," In Proceedings of Fifth International Conference on Cloud Computing, DOI: 10.1109/CLOUD.2012.48, pp. 408-415, 2012.

[11] S. Bohez, E.D.Coninck, T. Verbelen, P. Simoens, and B. Dhoedt, "Enabling Component-based Mobile Cloud Computing with the AIOLOS Middleware," ARM'14 December 9, 2014, Bordeaux, France, http://dx.doi.org/10.1145/2677017.2677019.

[12] E. Cuervo, A. Balasubramanian, Dae-ki Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making Smartphoness Last Longer with Code offload," Proceedings of 8th International Conference on Mobile Systems, Applications, and Services, MobiSys'10, ACM, New York, NY, USA 2010, pp.49-62.

[13] M. Ferber, T. Rauber, M.H.C. Torres, and T. Holvoet, "Resource Allocation for Cloud-Assisted Mobile Applications," Proceedings of 5th IEEE conference on Cloud Computing, DOI 10.1109/CLOUD.2012.75, pp. 400-407.

[14] M. Shiraj, A. Gani, A. Shamim, S. Khan, and R.W. Ahmad, "Energy Efficient Computation Offloading Framework for Mobile Cloud Computing," Journal of Grid Computing, DOI 10.1007/s10723-014-9323-6, 2015.

[15] C.Shi, K. Habak, P.Pandurangan, M. Ammar, M.Naik, and E. Zegura, "COSMOS:Compuation Offloading as a Srevice for Mobile Devices," MobiHoc'14 August 11-14, 2014, Philadelphia, PA, USA, http://dx.doi.org/10.1145/2632951.2632958.

[16] T. L. Botran, J. M. Alonso, and J. A. Lozano, "A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments." Journal of Grid Computing, pp. 1-34, 2014, DOI: 10.1007/s10723-014-9314-7.

[17] Rodrigo N. Calheiros, R. Ranjan, A. Beloglazov, Cesar A.F. De Rose, and R. Buyya, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," Software: Practice and Experience(SPE), Volume 41, Number 1, pp 23-50, ISSN: 0038-0644, Wiley Press, New York, USA, January 2011.

[18] Rodrigo N. Calheiros, R. Ranjan, and R. Buyya, "Virtual Machine Provisioning Based on Analytical and QoS in Cloud Computing Environments," International Conference on Parallel Processing 2011, DOI 10.1109/ICPP.2011.17 pp 295-304.

[19] L. Wu, S.K. Garg, S. Versteeg, and R. Buyya, "SLA-Based Resource Provisioning for Hosted Software-as-a-Service Applications in Cloud Computing Environments," IEEE Transactions on Service Computing, Vol. 7, No. 3, July-September 2014.