Cloudlet-based Mobile Offloading Systems: a Performance Analysis

Huaming Wu, Katinka Wolter
Department of Mathematics and Computer Science
Free University of Berlin, Berlin, Germany
{huaming.wu, katinka.wolter}@fu-berlin.de

Alessandro Grazioli
Information Engineering Department
Universit‘a degli Studi di Parma, Parma, Italy
alessandro.grazioli81@gmail.com

***Abstract*—Offloading is an effective method to migrate programs from mobile devices to cloud, but it critically depends on network and cloud conditions. We suggest that the mobile device does not directly communicate with a distant cloud, but instead, with a nearby cloudlet acting as an intermediate node. Such a cloudlet-based offloading system is modeled and analyzed with respect to the state transition process with failures. In addition, we evaluated the failure and repair time, and four types of execution time as well. The numerical results reveal that in environments characterized by higher reachability of cloudlet and cloud, longer connection time or even larger speedup factor F, this scheme benefits from reduced execution time.**

***Index Terms*—offloading system; cloud computing; cloudlet; network bandwidth; performance analysis**

I. INTRODUCTION

Offloading a program from mobile devices to cloud is becoming increasingly attractive to reduce execution time and extend battery life. Apple's Siri and iCloud [1] are two remarkable examples. However, cloud offloading critically depends on a reliable end-to-end communication and on the availability of the cloud. In addition, it suffers from high network access latencies and low network bandwidth. Mahadev [2] proposed a vm-based cloudlet for the infrastructure setup of mobile systems. Instead, we want to investigate how effective and efficient they are and what factors influence their performance. With this purpose, we introduce a mathematical model and analyze cloudlet-based offloading systems with failures, considering application execution time and failure recovery time.

II. SYSTEM OVERVIEW

A. Problems concerning direct offloading systems

Network condition: Different network types have a large impact on communication time, cost and energy. 3G provides a near-ubiquitous coverage, but it consumes more energy than WiFi because of latencies, and is sensitive to location [3].

Cloud condition: Offloading is difficult in locations such as the interior of a tunnel or subway, where the low network bandwidth prevents cloud applications from working properly. In addition, distant cloud dependence could lead to severe problems when service outages occur.

B. Overview of cloudlet-based offloading systems

Rather than relying on a distant cloud, the resource poverty of a mobile device can be addressed by using a nearby resource-rich cloudlet via a wireless LAN. A cloudlet is a trusted, resource-rich computer which is well-connected to the internet and is available for use by nearby mobile devices [2]. As shown in Fig.1, cloudlets are dispersed and located close to mobile devices while cloud is generally far. At runtime, the app discovers a nearby cloudlet and offloads a computation-intensive program to it. The mobile device does not need to communicate with the distant cloud, but only with the cloudlet. This model decreases latency and lowers battery consumption by using WiFi instead of broadband wireless technology.

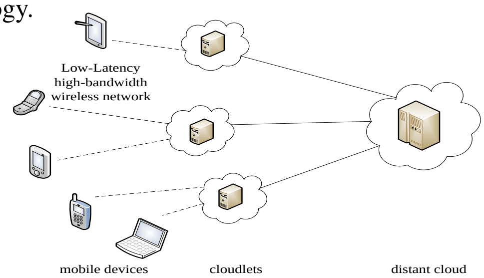

Fig. 1. Architecture of cloudlet-based offloading systems

Because wireless LAN bandwidth is remarkably higher than the bandwidth provided by radio access on a mobile device [3], we choose a path connecting the mobile device to a nearby cloudlet and then to a remote cloud. As depicted in Fig.2, D is the communicated data and B is the bandwidth between the mobile device and the cloud. Likewise, B_1 is the bandwidth between the mobile device and cloudlet, which generally uses gigabit internal connectivity and a high-bandwidth wireless LAN. B_2 is the bandwidth between the cloudlet and cloud, which is usually based on broadband technology. Generally, $B \leq B_1$ and $B \leq B_2$. Direct offloading saves execution time only if $T_{\mathrm{m}} > T_{\mathrm{s}} + \frac{D}{B}$, $T_{\mathrm{m}} = FT_{\mathrm{s}}$; T_{m} and T_{s} are the execution times on the mobile device and cloud, respectively; and the speedup $F \geq 1$ indicates how powerful the cloud is by comparison with the mobile device. Similarly, the cloudlet-based offloading saves time when $T_{\mathrm{m}} > T_{\mathrm{s}} + \frac{D}{B_1} + \frac{D}{B_2}$. Therefore, the cloudlet-based offloading model performs better than direct offloading approach when $\frac{1}{B} > \frac{1}{B_1} + \frac{1}{B_2}$.

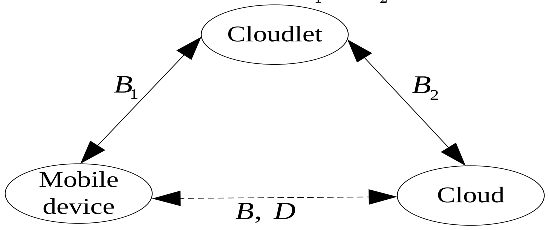

Fig. 2. Model of cloudlet-based offloading systems

III. PERFORMANCE ANALYSIS

A. Ideal cloudlet-based offloading systems

For ideal offloading systems, no failure occurs. The pure program execution time is $T_{\mathrm{p}}(n) = (1-\sigma) \cdot T_{\mathrm{m}}(n) + \frac{\sigma \cdot T_{\mathrm{m}}(n)}{F}$ [5], here σ is the proportion between the sub-tasks performed by the cloud and the mobile device; $0 \leq \sigma \leq 1$, $(1-\sigma) \cdot T_{\mathrm{m}}(n)$ and $\frac{\sigma \cdot T_{\mathrm{m}}(n)}{F}$ represent the execution time spent on the mobile device and cloud, respectively; n is the number of sub-tasks. The total execution time is $T_{\mathrm{OPT}}(n) = T_{\mathrm{p}}(n) + T_{\mathrm{c}}(n)$, $T_{\mathrm{c}}(n) = \frac{D}{B_1} + \frac{D}{B_2}$ is the total communication time.

B. Cloudlet-based offloading systems with failures

There are four states depicted in Fig.3. When offloading starts, the execution state changes from S_{NE} to S_{OE1}. If the remote cloud is available, it changes to S_{OE2}. Once the executions of all offloaded components are successfully completed, the execution state changes back from S_{OE2} to S_{NE}. However, failures may occur in all states as [5]:

1) S_{NE}: running out of battery, abnormal shutdown.

2) S_{OE1}: wireless link failures, cloudlet shutdowns or becomes unreachable due to mobile device's movement.

3) S_{OE2}: cloud unavailable, cloud outages or becomes unreachable due to cloudlet's failures.

4) S_{FR}: nested failures may also happen.

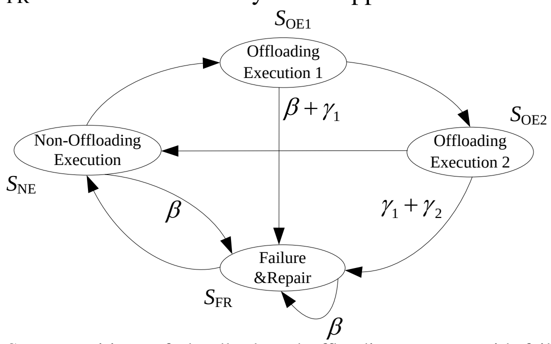

Fig. 3. State transitions of cloudlet-based offloading systems with failures

Offloading completes when the execution period elapses without failure. Independent failures caused by the mobile device, cloudlet and cloud are modeled as non-homogenous Poisson Processes with rates β, γ_1 and γ_2, respectively. According to Fig.3, the failure rate $\lambda(t)$ is defined as:

$$\lambda(t) = \begin{cases} \beta & \text{in state } S_{\mathrm{NE}} \text{ and } S_{\mathrm{FR}} \\ \beta+\gamma_1 & \text{in state } S_{\mathrm{OE1}} \\ \gamma_1+\gamma_2 & \text{in state } S_{\mathrm{OE2}} \end{cases} \tag{1}$$

The time period R is required to complete a repair in the presence of nested failures. The expectation of the failure repair time is $E(R^*) = \frac{1}{\beta}\left(\frac{1}{E[\mathrm{e}^{-\beta R}]} - 1\right)$ [5]. In presence of failures, the program execution time is calculated as

$$\begin{aligned} E[T_{\mathrm{FT}}(n)] \;=\; & \frac{1}{2}\Big[(1-\alpha_1)\cdot E[T_{\mathrm{OE1}}(n)] + \alpha_1 E[T_{\mathrm{NE/FR}}(n)] + \\ & (1-\alpha_2)\cdot E[T_{\mathrm{OE2}}(n)] + \alpha_2 E[T_{\mathrm{NE/FR}}(n)]\Big] \end{aligned} \tag{2}$$

where $E[T_{\mathrm{NE/FR}}(n)] = \left(\frac{1}{\beta} + E[R^*]\right)\left(\mathrm{e}^{\beta E[T_{\mathrm{OPT}}(n)]} - 1\right)$ is the time spent in state S_{NE} and S_{FR}, $E[T_{\mathrm{OE1}}(n)] = \left(\frac{1}{\beta+\gamma_1} + E[R^*]\right)\left[\mathrm{e}^{(\beta+\gamma_1)E[T_{\mathrm{OPT}}(n)]} - 1\right]$ in state S_{OE1} and $E[T_{\mathrm{OE2}}(n)] = \left(\frac{1}{\gamma_1+\gamma_2} + E[R^*]\right)\left[\mathrm{e}^{(\gamma_1+\gamma_2)E[T_{\mathrm{OPT}}(n)]} - 1\right]$ in state S_{OE2}; α_1, α_2 are the probabilities of unreachability of cloudlet and cloud, respectively.

IV. Numerical results and analysis

If the program offloads, through a cloudlet, a number of sub-tasks to the cloud and the remaining ones are executed locally; we have execution time $T(n) = T_{\mathrm{m}}[(1-\sigma)n] + T_{\mathrm{FT}}(\sigma n)$, where $T_{\mathrm{m}}[(1-\sigma)n]$ and $T_{\mathrm{FT}}(\sigma n)$ are the times spent on the mobile device and cloud.

The parameters are set as follows: $E(R) = 10$, $n = 100$, $\sigma = 0.9$, $\beta = 10^{-3}$, $\gamma_1 = 10^{-4}$, $\gamma_2 = 10^{-5}$, $\alpha_1 = 0.1$ and $\alpha_2 = 0.2$. The average execution time for each sub-task on the mobile device is 5 second and $T_c = 0.3n$.

As shown in Fig.4, when F increases, the execution time decreases except for $T_{\mathrm{m}}(n)$, which is horizontal at 500s.

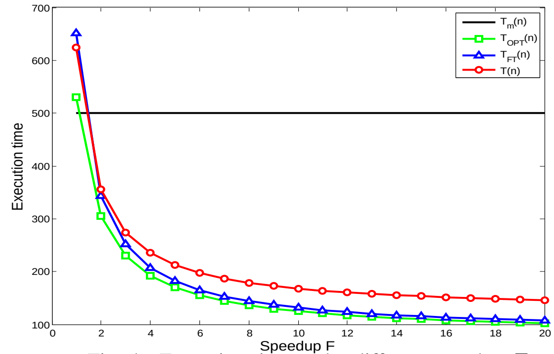

Fig. 4. Execution time under different speedup F

$T_{\mathrm{FT}}(n)$ and $T(n)$ produce a remarkably longer time than $T_{\mathrm{m}}(n)$ when $F < 2$, due to offloading and failure repairs. With larger F, the cloud can save execution time. However, time spent on offloading operation and failure handling will increase the total execution time, especially in higher cloudlet or cloud unreachability environments.

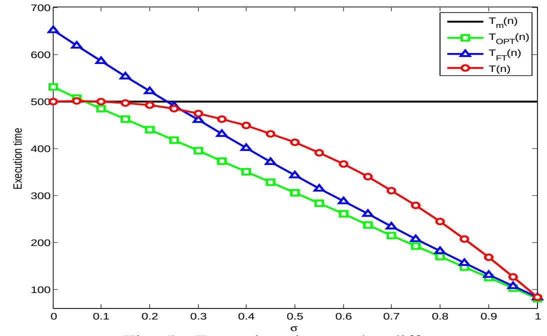

Fig. 5. Execution time under different σ

As shown in Fig.5, $F = 10$, $T(n)$ reduces to $T_{\mathrm{m}}(n)$ when $\sigma = 0$ and $T(n)$ approaches $T_{\mathrm{FT}}(n)$ when $\sigma = 1$. Both $T_{\mathrm{FT}}(n)$ and $T_{\mathrm{OPT}}(n)$ decrease along with the increase of σ, while $T(n)$ first decreases slightly and afterwards more rapidly with the increase of σ. $T_{\mathrm{FT}}(n)$ is larger than $T_{\mathrm{m}}(n)$ when $\sigma < 0.25$ due to offloading and failure recoveries. The larger σ is, the more execution time the cloudlet-based offload system saves.

V. Conclusion

In this short paper, we proposed an analytical model for cloudlet-based offloading systems, where the mobile device communicates with a nearby cloudlet instead of a remote cloud during the entire offloading process. In the environments characterized by high cloudlet or cloud unreachability, long disconnection time or even small speedup factor, this scheme will not benefit from reduced application execution time. The analysis results provide useful guidance for the design of efficient offloading systems.

References

[1] D. Niu, "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless Communications*, vol. 3, 2013.

[2] 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, pp. 14-23, October 2009.

[3] B. G. Chun and P. Maniatis, "Dynamically partitioning applications between weak devices and clouds," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing and Services: Social Networks and Beyond (MCS)*, no. 7, June 2010.

[4] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: can offloading computation save energy?" *IEEE Computer*, vol. 43, no. 4, pp. 51-56, April 2010.

[5] S. Ou, Y. Wu, K. Yang and B. Zhou, "Performance analysis of fault-Tolerant offloading systems for pervasice services in mobile wireless environments," in *Proceedings of the 2008 IEEE International Conference on Communications (ICC'08)*, pp. 1856-1860, May 2008.