A Multi-criteria Resource Allocation Mechanism for Mobile Clouds

Simin Ghasemi Falavarjani[1(✉)], Mohammad Ali Nematbakhsh[1], and Behrouz Shahgholi Ghahfarokhi[2]

[1] Department of Computer Engineering, University of Isfahan, Isfahan, Iran
simin.ghasemi@gmail.com, nematbakhsh@eng.ui.ac.ir
[2] Department of Information Technology Engineering, University of Isfahan, Isfahan, Iran
shahgholi@eng.ui.ac.ir

Abstract. Due to resource scarcity of mobile devices in Mobile Cloud Computing (MCC), intensive computing applications are offloaded into the cloud. There is a three-tier architecture for MCC consisting of distant cloud servers, nearby cloudlets and adjacent mobile devices. In this paper, we consider third tier. We propose an Optimal Fair Multi-criteria Resource Allocation (OFMRA) algorithm that minimizes the completion time of offloading applications along maximizing lifetime of mobile devices. Furthermore to stimulate selfish devices to participate in offloading, a virtual price based incentive mechanism is presented. The paper also designs an Offloading Mobile Cloud Framework (OMCF) which collects profile information and handles the offloading process. A prototype of the proposed method has been implemented and evaluated using a high computational load application. The results show that the proposed algorithm manages the tradeoff between optimizing completion time and energy well and improves the performance of offloading using the incentive mechanism.

Keywords: Mobile cloud computing · Offloading · Resource allocation · Multi-criteria optimization · Cooperation incentive · Fairness

1 Introduction

Recent years have witnessed an increasing popularity of mobile applications. The latest study by Juniper Research forecasts that market for cloud-based mobile applications will rise to $9.5 billion by 2014 [1]. However, some mobile devices cannot execute intensive computing applications such as image processing, speech recognition and data mining due to their resource (processing power, memory, battery life) restrictions [2]. To address this problem, it has been suggested to offload computation from mobile device to the cloud [2, 3].

A three-tier architecture is defined for MCC which consists of remote cloud servers, nearby computing servers known as cloudlets and neighboring mobile devices [2]. However, offloading to remote cloud servers causes long delay and offloading to local cloudlets limits mobility of devices [2, 4]. Therefore in this paper third tier is

A. Movaghar et al. (Eds.): CNDS 2013, CCIS 428, pp. 145–154, 2014.
DOI: 10.1007/978-3-319-10903-9_12

considered which consists of vicinal mobile devices where both service requester and service providers are mobile devices. It is also assumed that mobile devices are always connected (single hop). It is a usual case in scenarios such that all mobile devices belong to the same user or devices that are carried by a group of people on a trip or a mission e.g. in military or disaster relief scenarios [2, 5].

One of the most challenging issues in MCC is allocating appropriate resources for a given offloading service request towards satisfy QoS requirements.

Most of the previous works on resource allocation in mobile clouds have tried to decide which parts of application should be offloaded to the cloud towards optimizing time or energy. They delegate the resource allocation problem to the cloud [6, 7].

On the other hand, some researchers try to select appropriate devices among available mobile devices to allocate tasks to them optimizing specific metrics [4, 5, 8, 9]. Scavenger [8] provides a dual-profiling scheduler to assign tasks to available surrogates in order to minimize response time. In Serendipity [9], initiator mobile device allocates tasks to resource providers among mobile devices which intermittently encounter to it. In time-optimizing Serendipity, the objective is minimizing completion time while in energy-optimizing Serendipity; minimizing local energy consumption is regarded. Wei et al. [4] define a Hybrid Local Mobile Cloud Model (HLMCM) which consists of mobile devices and local cloud infrastructures and present an application scheduling algorithm which aims to maximize the profit of HLMCM while minimizing energy consumption of mobile devices. In [5] the authors consider environments which offloading is occurred among a set of mobile devices forming a Mobile Device Cloud (MDC) where nodes are highly collaborative and propose an offloading schema objects to maximize lifetime of the MDC.

However, none of the previous literatures in MCC provides a fair resource allocation method that optimizes both response time and consumed energy with respect to QoS constraints.

In this paper, an Optimal Fair Multi-criteria Resource Allocation (OFMRA) algorithm is proposed which minimizes both response time of offloading service and consumed energy of mobile devices (provides fairness via maximizing lifetime of all mobile devices) and considers deadline and budget constraints. In addition, an approach is needed to stimulate selfish mobile devices in order to participate in other's offloading process. To overcome this problem, a virtual price based incentive mechanism is presented and is considered in OFMRA. Moreover, an Offloading Mobile Cloud Framework (OMCF) is designed which gathers profile information and manages offloading process.

In summary, the main contributions of this paper are as follows: (1) proposal of OFMRA, a QoS constrained and fair resource allocation algorithm that optimizes completion time and energy consumption of offloading; (2) considering a virtual price based incentive mechanism in OFMRA to stimulate selfish mobile devices; and (3) design and implementation of OMCF to handle offloading process.

The rest of the paper is organized as follows. Section 2 explains how to stimulate mobile devices to cooperate and describes the OFMRA algorithm. In Sect. 3, Architecture of OMCF and profile estimation is stated. Implementation and evaluation of the framework is explained in Sect. 4 and the paper is concluded in Sect. 5.

2 Optimal Fair Multi-criteria Resource Allocation

In mobile cloud, when a mobile client wants to execute an intensive computing application, it requests offloading service from neighboring mobile devices. It first collects information about the application and available mobile devices (as will be described in Sect. 3) and then selects appropriate service providers among neighboring mobile devices. In this paper, it is assumed that only one mobile client can request offloading service at any time and other clients should wait until termination of current offloading process.

In the following, the price based incentive mechanism is explained which is used in OFMRA. Then QoS constrained resource allocation problem is formulated and the proposed Optimal Fair Multi-criteria Resource Allocation algorithm is discussed.

2.1 Price Based Incentive Mechanism

In offloading process, some mobile devices may be selfish and refuse to cooperate in others' offloading process in order to save themselves battery and computing resources. Therefore encouraging such selfish mobile devices is a challenging issue in mobile cloud computing. Fernando et al. [2] explain that it is essential to persuade users to collaborate and share their resources.

In order to motivate mobile devices to participate in offloading processes, we use a virtual price based incentive mechanism inspired from [10] where imaginary credits are paid to mobile devices for executing tasks. The more tasks a mobile device executes the more credits it will obtain. The mobile device can then use these credits to pay for its future offloading requests. For each device, the price of computation service is proportional to its computing power i.e. more powerful devices are more expensive.

In this paper, we assume that all mobile devices are honest and trustworthy. So the incentive mechanism is as follow: at initiation time, credits of mobile devices are initialized to a specific fee. When mobile device A requests a computing service from service provider B, the mobile device B executes the requested service and returns the results and charging information to A (e.g. time it takes for B to run the service). After that, depending on the amount of computation, remaining credit of mobile device A decreases and credit of mobile device B increases.

Next, the QoS constrained resource allocation problem considering above incentive mechanism is modeled as a multi-criteria optimization problem which minimizes completion time and energy consumption together. Then, an algorithm is proposed to solve this problem.

2.2 Problem Modeling

Suppose we have a task consisting of n parallel and independent subtasks. We assume all subtasks are the same with equal amount of computation, and m mobile devices have been discovered around the initiating mobile device. Let R_1 shows initiating mobile device and R_j represents j^{th} resource provider. t_j is the time it takes to execute a

subtask on resource provider R_j, p_j denotes price per time unit running a subtask on j^{th} mobile device, e_j is amount of energy consumed by R_j to run a subtask per second, and $E0_j$ indicates the initial energy level of device R_j. Also assume that et_j denotes the energy expenditure on device R_j to transmit a unit of data and V_{in} and V_{out} represent size of input/output of each subtask. A solution of resource allocation problem is an array b of m + 1 entries where b_j is the amount of subtasks allocated to R_j.

As mentioned in [11], due to running subtasks in parallel, execution time of the task is the maximum entry of tl_j where tl_j is the turnaround time it takes for resource provider R_j to complete b_j subtasks.

$$completeTime = \max_{j\in\{1\ldots m+1\}} tl_j, \quad tlj = bj * \left[t_j + \left(\frac{Vin + Vout}{BandWidth}\right)\right] \tag{1}$$

In virtual price based incentive modeling, a mobile device should pay off to Rj based on the amount of resources it consumes to run the subtasks on that surrogate. We use the formula proposed in [11] where the price p_j is a virtual price as described in Sect. 2.1. Thus the overall expense of task is sum of payments as below:

$$expense = \sum_{j=1}^{m} b_j * t_j * p_j \tag{2}$$

Reducing time and expense as the goal of resource allocation problem, it may leads to assigning more subtasks to devices with more computing power and lower price. Thus the energy of those devices is consumed drastically and they will drain-out of batteries while other slower devices remain intact. However in a fair resource allocation, subtasks should be assigned in a manner that the minimum residual energy between all devices be maximized. In order to provide fairness, we consider energy consumption as another criterion regarding the method of [12] which assigns subtasks in proportion to residual energy of mobile devices. That mechanism consumes lesser energy while avoids devices with low residual energy to be contributed in offloading. Consumed energy involves the amount of energy consumed to execute the subtasks and the amount of energy consumed to transfer related data. Therefore, the system energy consumption is sum of all devices' consumed energy:

$$energy = \sum_{j=1}^{m+1} \frac{consumedE_j}{E0_j - consumedE_j}, \qquad consumedE_j = b_j * \left[t_j * e_j + et_j * (Vin + Vout)\right] \tag{3}$$

Thus the multi-criteria optimization problem of resource allocation is formulated as:

Minimize *CompleteTime*
Minimize *expense*
Minimize *energy*

Subject to

$$completeTime \leq T_0, \tag{4}$$

$$expense \leq M_0, \tag{5}$$

$$E0_j - b_j * \left[t_j * e_j + et_j * (Vin + Vout)\right] \geq 20/100 * E0_j \quad for\ all\ j = 1 \ldots m+1, \tag{6}$$

$$\sum_{j=1}^{m+1} b_j = n. \tag{7}$$

Above constraints satisfy some QoS requirements and check the feasibility conditions. Equation (4) is a QoS feature to ensure that completion time does not exceed to deadline T0. Equation (5) expresses budget constraint i.e. maximal expense that initiating mobile can pay for (M0). Equation (6) avoids allocating subtask to device Rj with energy level less than 20 % of E0j, preventing mobile devices from draining out of battery. Finally Eq. (7) guarantees all subtasks to be offloaded to remote devices or be executed locally.

A Pareto solution set is achieved solving above multi-criteria optimization problem. To obtain the best solution, following weighted sum method is used to calculate total "cost" and to determine the minimum cost as the best solution:

$$Cost = w_t * completeTime + w_m * expense + w_e * 1000 * energy \tag{8}$$

Let w_t, w_m and w_e represent the weights of completion time, expense and remaining energy, respectively. These weights determine the importance of each criterion with respect to user preferences. For example, a user may consider one unit of time as valuable as one unit of price and both are as valuable as energy, then weights are set to $w_m : w_t : w_e = 1 : 1 : 1$. Weights are normalized i.e. $w_t + w_m + w_e = 1$. So in above example, $w_m = 0.33$, $w_t = 0.33$ and $w_e = 0.33$. We multiply the energy criterion by 1000, to unify the scale of energy to other criteria.

2.3 Solving Optimization Problem

It is NP-complete to find optimal solution for the above resource allocation problem. We exploit a branch and bound algorithm to solve the optimization problem (see Algorithm 1).

In lines 8–11, time, expense, energy and cost are calculated according to Eqs. (1), (2), (3) and (8), respectively. In line 13, QoS constraints in Eqs. (4), (5), (6), and (7) are checked by promising function and bound function determines bound of time and energy. To estimate bound of completion time, a greedy approach is used that iteratively chooses the surrogates with minimum completion time for each remaining subtask. In energy bound, for each remaining subtask, the destination device

with maximum residual energy is selected. If the current solution is dominated none of solutions in Pareto solution set, it is added to Pareto solution set in line 15. At the end of algorithm, the best solution with minimum cost is selected. The complexity of above algorithm is $O(n^m)$ where n is number of subtasks and m is number of neighboring mobile devices. But the number of neighboring mobile devices is limited.

Algorithm 1. OFMRA

```
1: OFMRA(n, m, profiles) // n and m are number of subtasks
   and mobile devices, respectively. Profiles represent
   task, devices and network information
2:   queue = initPriorityQueue();
3:   While !queue.empty()
4:     parentState = queue.poll();
5:     remTsk = parentState.remaingSubtasks();
6:     for t=1 to remTsk do
7:       curState = initState(parentState, t);
8:       time = comleteTime(curState, profiles);
9:       expense = serviceExpense(curState, profiles);
10:      energy = residual_Energy(curState, profiles);
11:      cost = totalCost(curState, profiles);
12:      curState.setCriter(time, energy, expense, cost);
13:      if promising(curState) & bound(curState)
14:        if remTsk+t=n & paretoSet.NonDominate(curState)
15:          paretoSet.addPareto(curState);
16:        else
17:          queue.put(curState);
18:  bestSolution = paretoSet.selectMinimumCost();
```

3 OMCF System Architecture

Offloading Mobile Cloud Framework comprises two parts: client side which requests offloading service and server side which provides services. A mobile device contains both parts. When an application starts in a mobile device, OMCF client gathers profile information to solve the task allocation problem, assigning subtasks to appropriate devices. After execution of all subtasks, OMCF client gathers partial results from OMCF servers and merges them.

Figure 1 shows high-level overview of OMCF's architecture. The client consists of three components: profiler, solver and client proxy. Profiler gathers task, device and network information.

Task profile includes execution time and input/output size of the subtasks. It is assumed that developers of an application have annotated remoteable parts (methods) and provided task partitioning information via task profile. In this paper techniques explored in [8] is used to estimate runtime and output size of each subtask based on benchmarking.

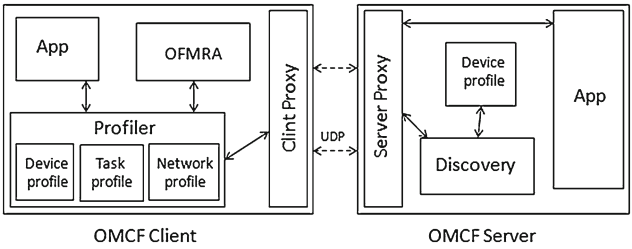


Fig. 1. High-level view of OMCF's architecture

Device profile contains execution speed and energy consumption model of each device. We use PassMark benchmark [13] and PowerBooter [14] to obtain processing power and energy consumption model of devices. Using task and device profiles, we can estimate time and consumed energy it takes to execute a subtask on every device, which is required for resource allocation.

Network profiler measures network bandwidth and latency, and monitors network connectivity. In this paper, we use the method explored in MAUI [6] i.e. 1 KB of data is sent periodically and transmission duration is measured to calculate network throughput. If network disconnection occurs, OMCF resumes offloaded subtasks and runs them locally. Collected profiles are passed to the OFMRA to find optimum solution for resource allocation problem.

On the server side, there are three components: device profiler, discovery component and server proxy. Every device intends to cooperate in offloading, introduces itself through proactive discovery protocol [15]. It broadcasts advertisement messages containing device's profile periodically. Client/Server proxies manage communication between client/server sides in local/remote mobile devices.

When offloading process ends, OMCF collects profiling information and makes a history-based profile. It uses the history-based profiles to learn and improve profile estimation for future offloading.

4 Evaluation

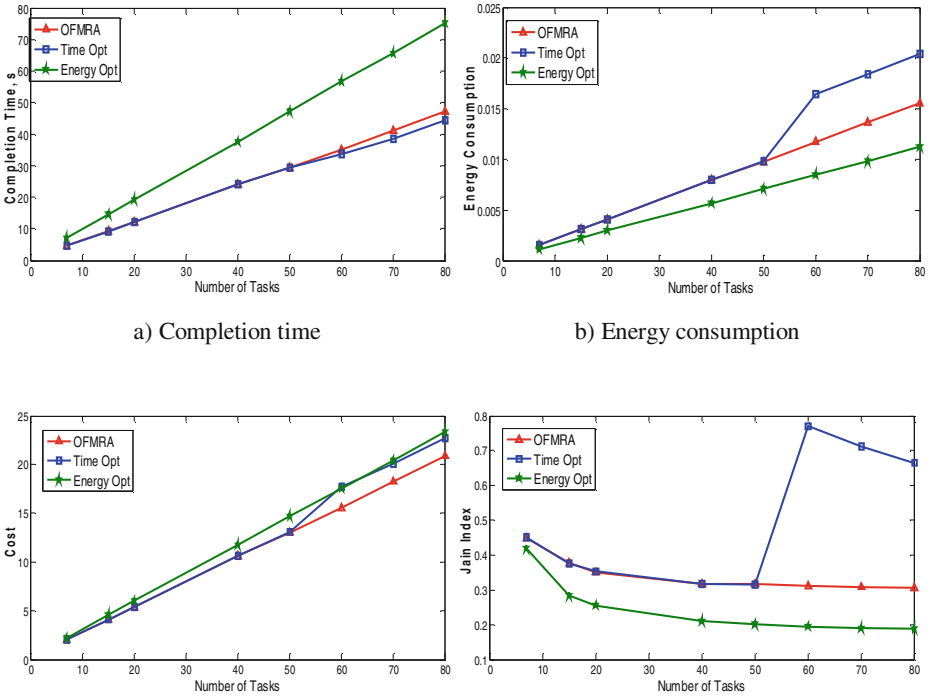
A prototype of OMCF (single-thread) has been implemented on Android and Windows OS. To evaluate performance and energy consumption of OMCF, a face detection application has been used which takes 1 MB picture as a subtask and identifies all faces in the picture.

The proposed framework has been tested on six devices: four HTC Wildfire S smartphones powered by 600 MHz ARMv6 processor and 512 MB of memory running Android 4.0.4 OS, a DELL Vostro with 2.2 GHz core 2 duo CPU and 4 GB of RAM and a Sony VAIO CB laptop with 2.4 GHz core i5 CPU and 6 GB of RAM. One of the smartphones requests for offloading service while others play the role of resource providers. The devices are connected to each other via an ad hoc network using Wi-Fi with 4 Mbps measured bandwidth.

To measure energy consumption of smartphones and laptops, PowerTour [14] and Joulmeter [16] are used, respectively. It is assumed that every device has 80 % initial energy. Price of devices is determined proportional to their computation power as presented by vector p = [0.006, 0.006, 0.006, 0.006, 0.127, 0.200]. The weights w_t, w_m and w_e are set to 0.17, 0.33 and 0.5, respectively. T_0 is considered 600 s while M_0 is 10 units of virtual money.

To demonstrate how OFMRA improves performance and energy conservation of mobile devices, we compare it with time-optimizing (time-opt) and energy-optimizing (energy-opt) algorithms in Serendipity [9] that minimize only completion time or consumed energy greedily in contrast to OFMRA that tries to minimize time and energy together.

The experimental results are as followed. Figure 2 shows the comparison of completion time, energy consumption, cost and fairness between OFMRA, time-opt and energy-opt algorithms, running face detection application on 7–80 pictures. Energy consumption is calculated using Eq. (3) while the total cost is determined from Eq. (8). As shown in Fig. 2, OFMRA stands in second place considering time or energy independently while it gives the minimum cost comparing to mentioned algorithms.



a) Completion time

b) Energy consumption

c) Cost

d) Fairness based on Jain index

Fig. 2. A comparison of (a) completion time, (b) energy consumption, (c) cost and (d) fairness (based on Jain index), between OFMRA, time-opt and energy-opt algorithms.

Energy-opt algorithm always gives subtasks to VAIO laptop which has the most battery capacity. Considering up to 50 pictures as input, both of time-opt and OFMRA algorithms give subtasks to Dell and VAIO laptops. When the number of input subtasks reaches over 50 pictures, time-opt uses other smarthphones to reduce the completion time but it yields to higher energy consumption and cost. The results illustrates that OFMRA manages the tradeoff between time and energy and tries to allocate subtasks in a way that minimize both of them. In addition, Fig. 2(d) shows Jain index based on ratio of consumed energy to initial energy of each device and demonstrates that OFMRA allocates subtasks among devices more fairly than energy-opt, however time-opt provides the best fairness.

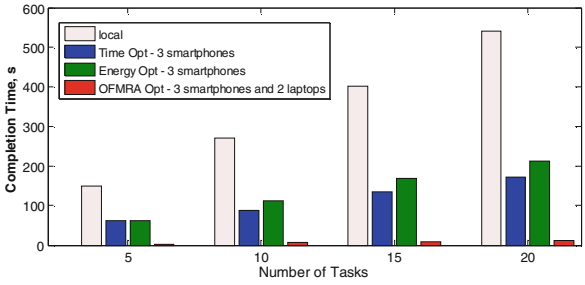


Fig. 3. The impact of virtual price based incentive on completion time.

Another experiment is also performed to show the impact of virtual price based incentive mechanism which is used in OFMRA. Figure 3 indicates completion time of 5, 10, 15 and 20 subtasks running locally or offloading to cooperative resource providers. It is assumed that two laptops are selfish and refuse to cooperate in offloading process. So time-opt and energy-opt algorithms without any incentives, offload subtasks to three smartphones which causes to take longer time. It is shown that using virtual price based incentive mechanism in OFMRA, stimulates mobile devices to participate which causes a great decrease in completion time. In addition Fig. 3 shows that offloading subtasks to remote devices improve performance comparing to local execution.

5 Conclusion

In this paper, an Optimal Fair Multi-criteria Resource Allocation algorithm is proposed which minimizes the completion time and energy consumption of offloading service, considering QoS constraints and provides fairness among mobile devices. Furthermore a virtual price based incentive mechanism is presented to stimulate selfish mobile devices to cooperate in offloading process. An Offloading Mobile Cloud Framework is also designed that enables mobile devices to offload intensive parts of their applications to the mobile cloud.

The experimental results show that in comparing to time-optimizing and energy-optimizing algorithms which lead to optimize time and energy respectively, the

proposed algorithm achieves better results considering both criteria together and manages the tradeoff between them well. In addition, using virtual price based incentive mechanism stimulates all mobile devices to participate in offloading process and improves the response time.

References

1. Perez, S.: Mobile cloud computing: $9.5 billion by 2014. http://www.readwriteweb.com/archives/mobile_cloud_computing_95_billion_by_2014.php
2. Fernando, N., Loke, S.W., Rahayu, W.: Mobile cloud computing: a survey. Future Gener. Comput. Syst. **29**(1), 84–106 (2012)
3. Satyanarayanan, M.: Pervasive computing: vision and challenges. Pers. Commun., IEEE **8** (4), 10–17 (2001)
4. Wei, X., Fan, J., Lu, Z., Ding, K.: Application scheduling in mobile cloud computing with load balancing. J. Appl. Math. **2013**, 1–13 (2013)
5. Mtibaa, A., Fahim, A., Harras, K.A., Ammar, M.H.: Towards resource sharing in mobile device clouds: power balancing across mobile devices. In: 2nd ACM SIGCOMM Workshop on Mobile Cloud Computing, pp. 51–56. ACM (2013)
6. Cuervo, E., Balasubramanian, A., et al.: MAUI: making smartphones last longer with code offload. In: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys), San Francisco, CA, USA, pp. 49–62. ACM (2010)
7. Chun, B.G., et al.: Clonecloud: elastic execution between mobile device and cloud. In: 6th Conference on Computer Systems, EuroSys 2011, pp. 301–314. ACM (2011)
8. Kristensen, M.: Scavenger: transparent development of efficient cyber foraging applications. In: IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 217–226 (2010)
9. Shi, C., Lakafosis, V., Ammar, M.H., Zegura, E.W.: Serendipity: enabling remote computing among intermittently connected mobile devices. In: 13th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 145–154 (2012)
10. Lu, R., Xiaodong L., Haojin Z., Xuemin S., Bruno P.: Pi: a practical incentive protocol for delay tolerant networks. IEEE Trans. Wirel. Commun. **9**(4), 1483–1493 (2010)
11. Wei, G., Vasilakos, A.V., Zheng, Y., Xiong, N.: A game-theoretic method of fair resource allocation for cloud computing services. J. Supercomput. **54**(2), 252–269 (2010)
12. Chang, J.H., Tassiulas, L.: Energy conserving routing in wireless ad-hoc networks. In: 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 1, pp. 22–31. IEEE (2000)
13. PassMark Benchmark. http://www.cpubenchmark.net
14. Zhang, L., et al.: Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In: 8th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, pp. 105–114. ACM (2010)
15. Kristensen, M.D.: Scavenger-mobile remote execution. DAIMI Report Series 37(587) (2008)
16. Joulmeter. http://research.microsoft.com