J Med Syst (2013) 37:9957
DOI 10.1007/s10916-013-9957-0

ORIGINAL PAPER

mHealthMon: Toward Energy-Efficient and Distributed Mobile Health Monitoring Using Parallel Offloading

Jong Hoon Ahnn · Miodrag Potkonjak

Received: 8 April 2013 / Accepted: 17 June 2013

Abstract Although mobile health monitoring where mobile sensors continuously gather, process, and update sensor readings (e.g. vital signals) from patient's sensors is emerging, little effort has been investigated in an energy-efficient management of sensor information gathering and processing. Mobile health monitoring with the focus of energy consumption may instead be holistically analyzed and systematically designed as a global solution to optimization subproblems. This paper presents an attempt to decompose the very complex mobile health monitoring system whose layer in the system corresponds to decomposed subproblems, and interfaces between them are quantified as functions of the optimization variables in order to orchestrate the subproblems. We propose a distributed and energy-saving mobile health platform, called mHealthMon where mobile users publish/access sensor data via a cloud computing-based distributed P2P overlay network. The key objective is to satisfy the mobile health monitoring application's quality of service requirements by modeling each subsystem: mobile clients with medical sensors, wireless network medium, and distributed cloud services. By simulations based on experimental data, we present the proposed system can achieve up to 10.1 times more energy-efficient and 20.2 times faster compared to a standalone mobile health monitoring application, in various mobile health monitoring scenarios applying a realistic mobility model.

Keywords Mobile health monitoring · Energy optimization · Parallel offloading · Mobile cloud computing

Introduction

Many successful health care applications based on mobile computing and communication technologies have been presented in the literature. Lv et al. [1] utilizes wireless body sensors and smart phones to monitor the wellbeing of the elderly. The key enabler is a smartphone that automatically alerts preassigned people who could be their family and friends, and call the ambulance of the emergency center. As an example of more sophisticated health monitoring systems, Chowdhury et al. proposed MediAlly, a middleware for supporting energy-efficient, long-term remote health monitoring, where sensor data is collected using physiological sensors and transported back to the middleware using a smartphone [4]. Smartphones with medical sensors attached to patients can perform publish/access medical sensor updates such as vital signals, measure personalized estimates of impact and exposure, and share patient's live health information. mHealthMon is similar in a spirit of a middleware-based system but it is fundamentally different in terms of an energy-saving paradigm.

In ArchRock and SensorBase [17], sensor data from a sensor network is aggregated at the local gateway and is published to the front-end server through which users can share the data. In SensorBase [17], back-end servers (called republishers) further process sensor data to enable sensor data searching. SensorMap [18] is a web portal service that

J. H. Ahnn (✉)
Department of Computer Science,
UCLA and Cloud Research Lab,
Samsung Information Systems America,
Los Angeles, CA 90034-4205, USA
e-mail: jhahnn@gmail.com

M. Potkonjak
Department of Computer Science, UCLA,
Los Angeles, CA 90034-4205, USA
e-mail: miodrag@cs.ucla.edu

Published online: 30 July 2013

provides mechanisms to archive and index data, process queries, and aggregate and present results on geocentric Web. mHealthMon differs from these approaches in that it focuses on large-scale participatory sensing and facilitates location-sensitive information sharing via a scalable structured P2P overlay that efficiently supports location-sensitive data publish/retrieval. Similar to other works such as GeoServ [16], mHealthMon is a two-tier mobile health monitoring platform that exploits the P2P Internet infrastructure.

The approaches from MAUI [5], CloneCloud [3], Cloudlets [7], and Zhang et al. [23] seem promising because their model incorporates a cost model for deciding best execution configuration, and they can be also adapted dynamically according to real-time conditions. The approach in [24] is similar to above, but it lacks of dynamic adaptation of the computation between mobile devices and cloud services. Prior work mostly focused on saving energy consumption on mobile devices; in contrast, mHealthMon provides analytical cost models to optimize the entire energy consumption including network and cloud at the same time.

Although many researcher have studied mobile health monitoring, little attention has been paid in modeling the system with the focus on energy saving. As depicted in Fig. 1, the overall system model may include several subsystems: mobile terminals, multiple wireless network interfaces, and cloud services. Recent research efforts have proposed several system architectures and its communication mechanism between a mobile and cloud side. However, without having concrete models in the performance of application and wireless communication medium, it is difficult to quantify the cost of operations due to dynamic nature of mobile health monitoring applications.

The key contributions are summarized in the following. We explicitly model the performance of mobile health monitoring system – mobile clients with medical sensors, wireless network medium, and cloud services – using two aspects: computation and communication cost. We propose a distributed optimized solution of complex mobile health monitoring: program partitioning , network resource allocation, and network selection problem. We propose a location-aware sensor data retrieval scheme called mHealthMon that supports geographic range queries, and a location-aware publish-subscribe scheme that enables energy-efficient multicast routing over a group of subscribed users. We prototype energy-optimized mobile health monitoring applications to prove the feasibility of our proposed techniques in various sensing scenarios applying a realistic mobility model utilizing parallel offloading.

Related work

Mobile health monitoring: Today, personal healthcare is one of emerged areas of research. Rodriguez et al. [11] have made a classification which divides the solutions into three groups. The first group records signals and takes action off-line. The second group has the feature that systems perform remote real-time processing. The last group provides local real-time processing, with taking into account the level of mobility. Our work belongs to the third category. Compared with the group two, the third group performs the local real-time monitoring in order to detect some anomalies and send alert to a control center or a hospital. Wu et al [12] proposes a wearable personal healthcare and emergency aid system called WAITER. Ziyu et al. uses wireless

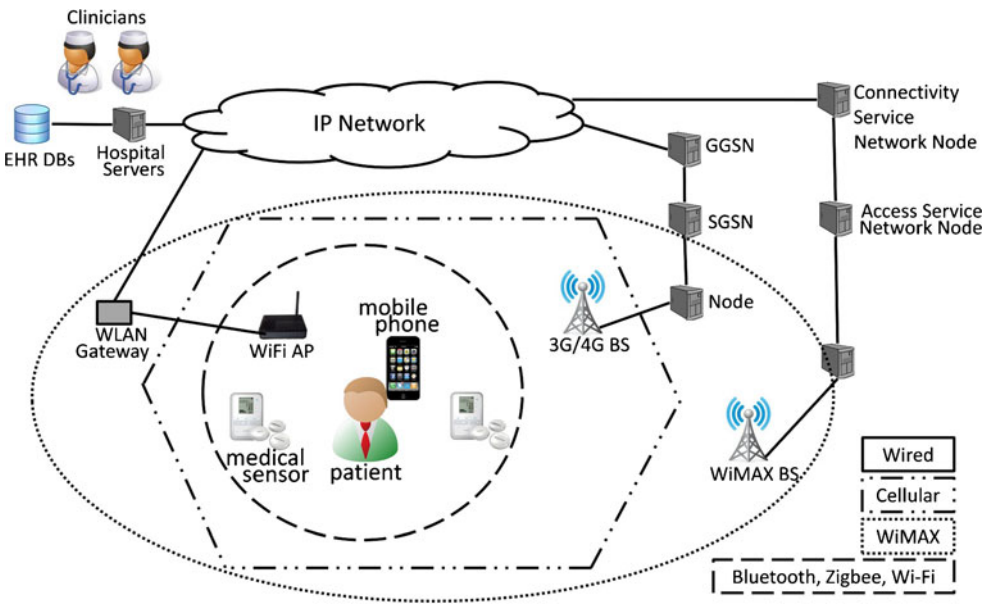

Fig. 1 A high-level overview of smartphone-based mobile health network architecture in a heterogeneous wireless network interfaces scenario

body sensors and smartphones to monitor the wellbeing of the elderly [1]. The key enabler is a smartphone that automatically alerts preassigned people who could be the old people's family and friends, and call the ambulance of the emergency center. The AMON system [8] for multi-parameter (Sp02, pulse and temperature) monitoring, the MoteCare system [9] for personalized health monitoring and For The COSMOS middleware [10] is for ubiquitous monitoring using ZigBee-based sensors. Chowdhury et al. proposed MediAlly, a middleware for supporting energy-efficient, long-term remote health monitoring, where sensor data is collected using physiological sensors and transported back to the middleware using a smartphone [4]. mHealth-Mon is similar in a spirit of a middleware-based system but it is fundamentally different in terms of an energy-saving paradigm. Prognosis [13] is a physiological data fusion model of wearable health-monitoring system for people at risk which contains decision support system and finite-state machine. It can estimate users' health status and offer corresponding alerts. Gay and Leijdekkers [14] have developed one application that can monitor the wellbeing of high risk cardiac patients using wireless sensors and smartphones.

Althgouh forementioned work is well studied, most of them relies on flat back-end systems which may not be flexible and scalable when billions of mobile devices are targetted. For instance, Samsung has been building its own big data platform for storing and processing data collected from all the products and applications such as mobile health monitoring. It is not surprising that such a big IT company targets 1 billion users. Therefore, mHealthMon provides a way to utilize hierarchical GPS-tagged data structure to build a scalable overlay on the cloud. Furthermore, no prior work supports computational offloading. We step further to support concurrent and parallel offloading to save more energy and execution time in health monitoring and its data processing.

System architecture

The main goal of mHealthMon is to provide a noble way to maximize the benefits of *parallel* code offloading upon the presence of multiple cloud machines. From a bottom up approach, consider a mobile client/terminal/user in Fig. 2. A mobile device hosts an application which may consume lots of battery time and energy. To efficiently run the application without draining its battery and at the same time saving its execution time, we consider cloud services where some part of application components can be offloaded possibly in a parallel manner. Mobile clients and cloud services are connected via several intermediate nodes such as WLAN gateways, SGSN, GGSN, connectivity service network nodes, and access service network nodes. The networked system offers heterogeneous radio access technologies (RAT)s for wireless communications between them. This means there exist several routes from a source to a destination due to multiple access medium.

Issues in the system design are follows. From a mobile client's perspective, there are two main considerations. First, a program offloading decision must be made based on analytic cost models (see section "Cloud-based Distributed Health Sensing System") before a mobile application is launched in a static setting. In a dynamic setting, the offloading decision can be reconfigured based on periodic profiling operating conditions (e.g. network data rate) even when the application is running. Second, one of wireless network technologies must be chosen before it transfers data required for offloading program pieces. From a network operator's perspective, efficient and fair resource (bandwidth) allocation must be made upon mobile clients' arrival to maximize the overall throughput of networked systems.

From a mobile client, an offloading decision can be made based on two key factors: computation cost and communication cost of a given mobile application. The computation cost is incurred when a mobile client launches an application on its mobile side, resulting in consuming energy for CPU, I/O, and other hardware resources. On the other hand, we consider the communication cost as an additional cost when some part of the application is offloaded. Thus, the communication cost can be evaluated to be acceptable only when the mobile client benefits from its offloaded computation in terms of time and energy saving.

Figure 2 presents a component-based view of the system architecture which consists of mobile terminal, wireless networks, and cloud services. The key to success in this design is to quantify the cost of offloading mobile applications. In program performance predictor, we predict the cost of computation for a given basic functional block (BFB). In this work, we define a BFB as a method or function in a program. The main challenge is the estimate varies when the input data fed to BFB changes. To overcome this issue, we model the performance of BFB with different data set using a regression theory based on a small set of profiling samples on a target machine such as mobile clients and cloud machines.

Cloud-based distributed health sensing system

Our prior work, GeoServ [16] mainly focuses on how to store in and retrieve sensor data from external storage systems, where the location-awareness is the main consideration on its data management over an overlay-based P2P

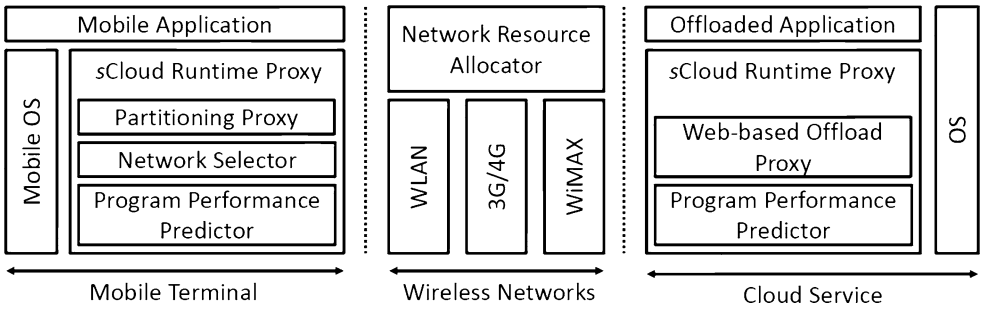

Fig. 2 An overview of mHealthMon software architecture

routing. Thus, GeoServ is a general purpose urban sensing P2P storage with no consideration of performance modeling, energy saving and optimization, and computation offloading.

As an ongoing effort of building urban sensing applications, we propose mHealthMon, a model-based "energy-efficient" system of sensor information sharing and computation offloading in urban environments. Unlike GeoServ, mHealthMon focuses on the performance/energy modeling and computational offloading by formulating cost functions (computation and communication cost). It also provides a way to optimize program execution time and energy for a given mobile application since the battery constraint is one of the biggest challenges in mobile phones. We adopted GeoServ as a "server-side" sensor data and computing resource management scheme which can be nicely integrated in our performance and energy optimization framework. Another contribution to mHealthMon is to formulate "mobile-side" several optimization formulations, which was never considered in GeoServ: a program partitioning program, network resource allocation problem, network selection problem. Therefore, mHealthMon much looks like a modern mobile cloud computing platform, focusing on enabling performance- and energy-efficient urban sensing applications.

System model

Going back to the network architecture of smartphone-based mobile health monitoring in Fig. 1, we start this section by studying the current status of urban sensing applications. The cost model of urban sensing falls in two fold. As analyzed in prior section, the cost of sensor data retrieval is a major issue in a cloud-based distributed sensor storage. Other important is the cost of offloading incurred by computational outsourcing from mobile to cloud. Several researchers have identified the fundamental challenges in urban sensing due to severe resource constraints and dynamic changes in operating conditions. There are two types of sensing applications: offline and online applications. The two dominating factors to distinguish the two are the cost of communication and that of computation. The former acts as fat mobile client that perform all the computation locally, while the latter splits its computation into local and remote parts, thus may incur additional communication cost to transfer necessary its binary and necessary data, however save the total amount of local computation. We apply a regression theory in modeling computation of mobile applications based on empirical measurement data. To model heterogeneous air interfaces such as WLAN, cellular network, and WiMAX, we apply a state-of-the-art mathematical network model based on empirical system parameters [21].

Computation model

We define a software program as a set of basic functional blocks (BFB)s, where a basic functional block corresponds to a single method or function in a program. Each BFB consists of a set of inputs as required knowledge of computation, and a set of outputs as an outcome of computation. These include both global and local variables defined in a program.

In order to model the performance of mobile applications in heterogeneous hardware environments, we apply a regression theory to derive statistical inference models, by taking a small number of samples, where each sample denotes the execution time of a BFB on a particular machine. In our regression model, a response is modeled as a weighted sum of predictor variables. By adopting statistical techniques, we then assess the effectiveness of model's predictive capability.

We suppose there are a subset of observations $\hat{\Theta}$ in a large observation space Θ for which values of response and predictor variables are known. A observed response vector is denoted by $\mathbf{y} = [y_1, ..., y_i, ..., y_\theta]$, where y_i denotes it response variable for a single observation $i \in \hat{\Theta}$ and a Φ predictor vector is denoted by $x_i = [x_i^\phi, ..., x_i^\phi]$. The corresponding set of regression coefficients is expressed by a

vector $\Gamma = [\gamma_0, ..., \gamma_\phi]$. Thus, a linear function of predictors Φ is given by,

$$f(y_i) = \Psi(x_i)\Gamma + \epsilon_i = \gamma_0 + \sum_{j=1}^{\Phi} \Psi_j(x_i^j)\gamma_j + \epsilon_i \quad (1)$$

γ_i can be seen as the expected change in y_i per unit change in the predictor variable x_i^j. An independent random error ϵ_i has mean $E(\epsilon_i) = 0$ and constant variance $Var(\epsilon_i) = \sigma^2$.

In order to determine the best fitting model, we consider least squared errors commonly used in minimizing $\Omega(\Gamma)$ the sum of squared deviations of predicted responses give by the model from observed responses.

$\Omega(\Gamma) = \sum_{i=1}^{\hat{\Theta}}(y_i - \gamma_0 - \sum_{j=1}^{\Phi} \gamma_j x_i^j)^2$ Obtaining estimates of the coefficients Γ is the goal of this approach. The correlation of response-to-predictor relationship is used in identifying the significance of the estimates. We express residuals to answer the problem of how well the model captures observed trends as, $\hat{\epsilon} = y_i - \hat{\gamma}_0 - \sum_{j=0}^{\Phi} \hat{\gamma}_j x_i^j$ Model fitting can be assessed by the F-test [20] which is a standard statistical test method using multiple correlation statistic R^2 given by $R^2 = 1 - \frac{\sum_{i=1}^{\hat{\Theta}}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{\hat{\Theta}}(y_i - \frac{1}{\hat{\Theta}}\sum_{i=1}^{\hat{\Theta}} y_i)^2}$ A larger R^2 value indicates better fits, while over-fitting if R^2 is close to 1. The over-fitting may occur when data sets are small and the number of predictors are large. A typical strategy is to set the number of predictors less than the number of observations given by $|\Phi| < \frac{\hat{\Theta}}{20}$ according to [20].

$$\hat{y}_i = E[\gamma_0 + \sum_{j=1}^{\Phi} \gamma_j x_i^j + \epsilon_i] = \gamma_0 + \sum_{j=1}^{\Phi} \gamma_j x_i^j \quad (2)$$

The Eq. 2 presents the expected value of y_i, $E[y_i]$ and its corresponding estimate $\hat{y}_i$ with $E[\epsilon_i] = 0$. We herein define a coefficient of performance η of a computer such as a mobile client and cloud server. The coefficient converts the performance in time $\hat{y}_i$ into the one in power or energy $\hat{P}_i$ on a computer j as $\hat{P}_i^j = \eta \cdot \hat{y}_i^j$, where η can be experimentally obtained.

Wireless network model

We consider multiple wireless network interfaces scenario where heterogeneous radio access technologies (RAT)s such as WIFI, WiMAX, UMTS, and GSM work together with their overlapping network coverage in a given area. According to many researchers such as [21] and [19], RATs can be largely characterized into two categories based on means to share their channels: interference constrained RATs and orthogonal RATs. In this paper, we only consider the latter.

Orthogonal RATs are network interfaces where they assume a fixed transmission power over all base stations (BS)s. In this sort of systems, time and frequency slots are thought to be their bandwidth per a base station, and their resources are assigned to mobile devices by different resource allocation techniques under various objectives such as utility, fairness, and priority. An typical example of such system is GSM/EDGE technology which is based on TDMA. The signal to interference and noise ratio (SINR) between a mobile client m and BS b can be given by

$$\nu_{m,b} = \frac{q_{m,b}\bar{P}_m}{\omega_b + \pi_b} \quad (3)$$

The thermal noise is denoted by π_b, the transmission power of BS is denoted by $\bar{P}_b$, and the intercell interference is given by ω_b for each BS b. THus, the assigned bandwidth $r_{m,b}$ to a mobile client m in BS b does not depend upon the SINR $\nu_{m,b}$. Therefore, the link rate $D_{m,b}$ of the orthogonal-based system is given by,

$$D_{m,b} = \bar{D}_{m,b} r_{m,b} \quad (4)$$

In TDMA system, $\bar{D}_{m,b}$ denotes the link rate of a unit time while it denotes the like rate of a unit frequency between a mobile client m and m in BS b. $\bar{D}_{m,b}$ can be rewritten as a function of SINR $\nu_{m,b}$ defined in Eq. 3.

Optimization problem formulation

We mainly solve different problems: a program partitioning program (P1), network resource allocation problem (P2), network selection problem (P3), and cloud resource allocation problem (P4) as depicted in Fig. 8. In this work, we mainly focus on the first three. A solution to the partitioning problem gives an optimal set of code offloading decisions in terms of computation cost and communication cost, while a solution to the network resource allocation problem gives an optimal allocation strategy toward maximizing the utility of network systems. It is obvious that solving the latter problem provides a way to choosing the best communication cost in the former problem.

Program partitioning problem

Let us consider a mobile application A and its call function graph $G = (V, E)$, where each vertex $v \in V$ denotes a method in A. An invocation of method v from one another u thereby is denoted by an edge $e = (u, v)$. We annotate each vertex with the execution time T_v of the method v and each edge with the data transfer time $T_{u \to v}$ incurred when the method v is offloaded from the method u. We reconstruct a new graph $G' = (V', E')$ from G by adding correspond-

ing offloading methods to V. The code partitioning problem based on G' can be formulated as,

$$\begin{aligned} min \quad & \textstyle\sum_{v' \in V'} T_{v'} + \sum_{e' \in E'} T_{e':u' \to v'}, \\ s.t. \quad & \frac{\sum_{v' \in V'} T_{v'} + \sum_{e' \in E'} T_{e'}}{\sum_{v \in V} T_v + \sum_{e \in E} T_e} \leq 1, \\ & T_{v'} \geq 0, T_v \geq 0, T_{e'} \geq 0, T_e \geq 0 \end{aligned} \tag{5}$$

The calculation of computation cost $T_v, T_{v'}$ depends upon the performance estimate $\hat{y}_i$ for each basic functional block (BFB) v, v'. Furthermore, the calculation of communication cost incurred due to code offload is given by,

$$T_{v'} = n_{v'} \times D_{m,b}, \tag{6}$$

where the assigned data rate is denoted by $D_{m,b}$ for a mobile client m in BS b, and the size of data to be transferred due to offloading for BFB v' is given by $n_{v'}$. We formulate further problems for how to assign the data rate to each mobile client in section "Network selection problem" and how to select one of the heterogeneous network interfaces in section "Network resource allocation problem". The problem formulated in Eq. 6 consists of a concave objective over linear constraints, and it becomes convex. Therefore, there are various convex optimization algorithms to solve it from [25].

Network resource allocation problem

We consider a utility metric as the effectiveness of allocated resources of networked systems in our optimization problem as,

$$U = \sum_m \sum_b D_{m,b} \tag{7}$$

In order to deal with fairness in resource allocation among mobile clients, the utility function with a weight variable w can construct the α proportional fairness as,

$$U = \sum_m \frac{w_m}{1-\alpha} \sum_b D_{m,b}^{1-\alpha}, \tag{8}$$

where $0 \leq \alpha < 1$. Now, we present an optimization problem as,

$$\begin{aligned} max \quad & U, \\ s.t. \quad & \textstyle\sum_m \frac{D_{m,b}}{D_{m,b}} \leq \Gamma_b, \\ & \textstyle\sum_b D_{m,b} \geq D_{min,b}, \\ & D_{m,b} \geq 0, \end{aligned} \tag{9}$$

where $D_{min,b}$ is the minimum data rate assigned to mobile clients. Our goal is for a network operator to maximize the sum of utility of all mobile users in all base stations. Note that Eq. 9 consists of a concave objective over linear constraints and thus is convex. That means there exists various algorithms to solve the problem immediately [25].

Algorithm 1 mHealthMon: an orchestrated approach to four different optimization algorithms to achieve a global optimization objective in a distributed manner. (also see Fig. 8)

Data: mobile client $m \in M$, mobile application $a \in A$, BFB $i \in a$, BS $b \in B_r$, RAT $r \in R$ cloud server $j \in J$
Result: Optimal offloading graph on optimal resources assigned: $\hat{G}'$
while *There exist jobs to be scheduled* **do**
  U = solve **P2**(D) - Eq. 9;
  $D_{m,b}$ = solve **P3**(U, D) - Eq. 6;
  ($\hat{y}$, P) = solve **P4**(A, J);
  $\hat{G}'$ = solve **P1**(m, a, $D_{m,b}$, $\hat{y}$) - Eq. 5;
end

Network selection problem

The network system model in this paper considers multiple wireless network interfaces with different radio access technologies (RAT)s such as WIFI, WiMAX, UMTS, and GSM having different capacity constraints and channel conditions. The problem we would construct is a decent network selection strategy that minimizes the expected mean cost of data transfer from a mobile client to a target offloading agent such as cloud machines. We develop a simple heuristic-based strategy $S(m, b, l) \in S$ where it takes into account current workload $l_m \in L$ for a mobile client $m \in M$ in a base station $b \in B$ which corresponds to the size of data to be transferred over the wireless network medium. The strategy $S(m, l)$ selects the best RAT which can support its workload l satisfying QoS requirements. We assume each mobile user belongs to one of K different user classes. With probability p_k, an arriving mobile user is characterized by a specific class k in the network system. Let $\bar{X}_k \in X$ denote a set of states loaded for class k specifying whether it allows the admission of a mobile user in class k to one of the RATs $r \in R$. Therefore, the strategy S(m,b,l,r) can be defined as,

$$\begin{aligned} S(m, b, l, r) := max \quad & l_m \times D_{m,b,r}, \\ & \forall l_m \in L \forall m \in M, \forall b \in B, \forall r \in R \end{aligned} \tag{10}$$

If there are several $S(m, b, l, r)$ that maximizes the performance of data transfer cost, among them the strategy chooses the one which is equivalent to the RAT having minimum current total workload.

Algorithm

We present an algorithm used in mobile health monitoring settings. We are not evaluating our optimization schemes one by one since our proposal is in nature coordinated and

orchestrated as shown in Algorithm 1. For example, when you make a decision for offloading computation, additional results coming from other two optimization problems are needed. One is network bandwidth assignment from wireless medium (P2). The other is the best wireless technology selection (P3). In the dynamic scenario, mobile clients or users request and their mobility are subject to a given mobility and traffic model rather than stochastic processes. Algorithm 1 solves P2, P3, P4, and P1 in order until the mobile cloud computing facility based on data centers ends. The network resource allocation problem P2 is for a network operator to maximize the sum of utility U of all mobile users in all base stations (Eq. 9). The following network selection problem P3 that maximizes the performance of data transfer cost, resulting in an optimal choice $D_{m,b}$ of RAT among candidate RATs (Eq. 6). The data center job allocation problem P4 minimizes the total amount of power consumption within a data center. It results in the performance estimate $\hat{y}$ for given computation to be offloaded by mobile applications, and its power consumption footprint P. Finally, the program partitioning problem collects computation cost and communication cost, and makes a decision of whether or not given computational blocks are outsourced based on optimal resources provided by P2, P3, and P4. The final offloading decision is denoted by a graph of BFBs $\hat{G}'$ (Eq. 5) with a set of optimal resources assigned.

System evaluation

Evaluating the performance consists of two parts: the performance in a mobile device and the one in a cloud machine. The former is presented in the computation cost and communication cost in five offload scenarios: local execution (L) in a mobile client, offloading with 1–5 concurrent requests (O1–O5). Note that O1 represents serial offloading similar to MAUI [5], [3], and Cloudlets [7]. Our work differs from them in that parallel and concurrent offloading is supported. For our proposed scheme, O2-O5 stands for asynchronous parallel offloading with the different number of concurrent parallel requests.

Evaluating mobile clients

This section analyzes computation cost and communication cost in time and energy when applying optimization strategies presented in this paper. We implemented a iOS-based mobile health monitoring applicaiton as shown in Figs. 3, 4, 5, 6, 7 and 8. Figure 9 compares the execution time between the mobile and the cloud by applying to a typical mobile health monitoring application. Figure 10 compares energy consumption in the same settings. As discussed, we also study how concurrent offloading requests help save time and energy in various scenarios: O1-O5. ROB presents relative offload benefits, comparing each offloading case with the non-offloading case L. We present the proposed system with the help of 5 parallel execution (O5) can perform up to 20.2 times faster and 10.1 times more energy-efficient compared to a standalone mobile health application L. We

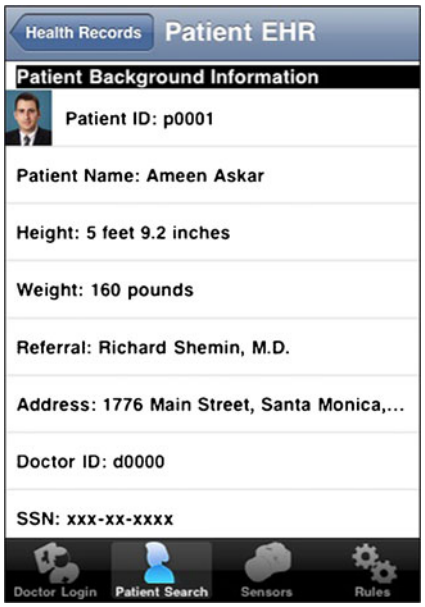

Fig. 3 We developed an iOS-based mobile health monitoring application: a screenshot for patient's EHR

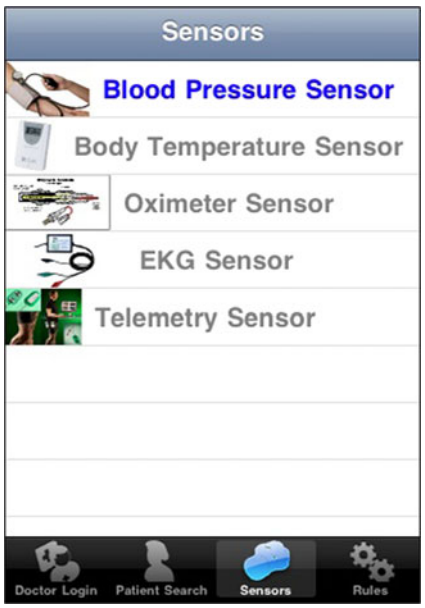

Fig. 4 A screenshot for a list of available medical sensors via Bluetooth

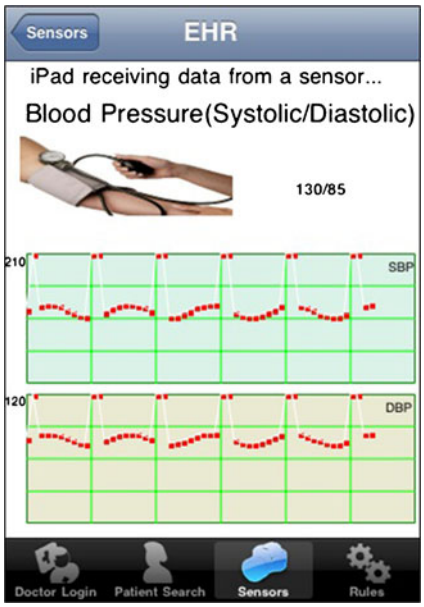

Fig. 5 A screenshot for sensor feeding (systolic/diastolic) from a blood pressure sensor

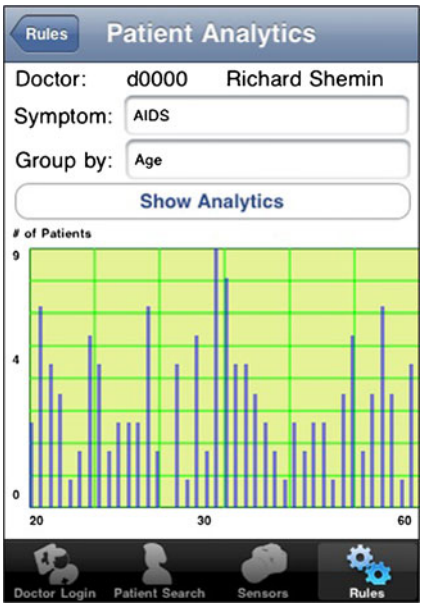

Fig. 7 A screenshot for patient's statistics

also observe that our work (O2–O5) outperforms over non-parallel offloading schemes (O1) such as MAUI [5] and CloneCloud [3], resulting in at least 2 times better both in time and energy. We observe the overall time saving and energy saving rate increases as the number of concurrent requests increases. This is done by a non-blocking (asynchronous) offload request.

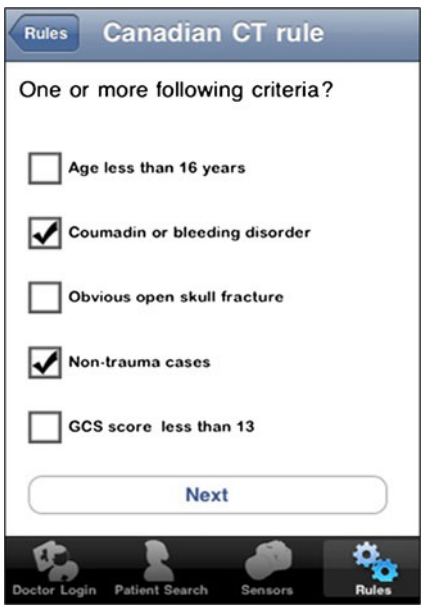

Fig. 6 A screenshot for the Canadian CT rule

Evaluating cloud servers

We implement an event-driven discrete-time simulator where each overlay hop takes a unit time. For the sake of a large-scale network simulation, our simulator does not model any queuing delay at intermediate nodes or packet loss on links. The simulator is implemented in C# and supports dynamic node generation/join/leave, load balancing, and publish-subscribe features. For system evaluation, we consider a large-scale participatory vehicular sensing scenario where mobile users in the cars participate in vehicular sensing projects (e.g., traffic information sharing and road condition monitoring). For realistic mobility generation, we use VanetMobiSim that simulates macro- and micro-mobility patterns in urban environments [22]. Macro-mobility deals with road topology/structure and traffic signs (stop signs, traffic lights, speed limits), and micro-mobility models the speed and acceleration of each mobile device.

For mobile scenarios, we use this mobility trace for the duration of 300s. We use the network area size of 12800m$\times$12800 m. The Westwood topology from Tigermap (TGR06037, Los Angeles [2]) represents the area in the vicinity of the UCLA campus. We discretize the network area into grids for the Hilbert curve-based linearization, resulting 100$\times$100 grids. Geographic range queries are made by specifying a square area (e.g., 4$\times$4 grids). Each mobile node reports sensor data to its associated overlay node every second. The size of data is set to 128 Bytes (e.g., GPS sample, timestamp, accelerometer samples). We assume that each node knows its accurate geographic coor-

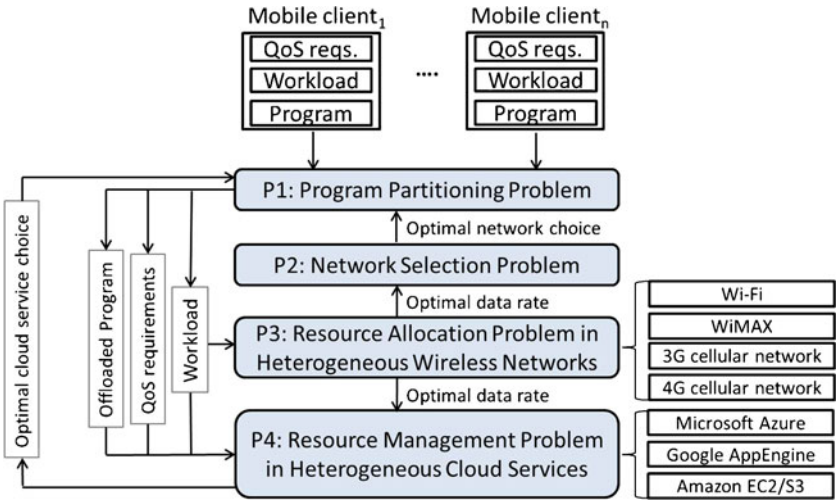

Fig. 8 A high level presentation of optimization solutions to green mobile cloud computing. Four optimization subproblems achieve a global optimization objective in a distributed fashion

dinate and thus can dynamically change their associated overlay node without any errors (e.g., no bouncing at the boundary). In GeoTable, the number of long links is set to five, as recommended in Symphony DHT [6]. Unless otherwise mentioned, for each configuration we report the average value of 30 runs. For simplicity, orthogonal-based WLAN and TDMA-based GSM are only considered. Unless specified, the network selection follows a strategy given in Equation 6 and network parameters form [19]. We assume that each BS knows its accurate geographic coordinate and thus can dynamically change their associated BS without any errors (e.g., no bouncing at the boundary). In our simulation, each mobile client randomly chooses one of four mobile applications. The duration of execution time and energy of each mobile application is given by experimental results. When one application is done, another random assignment is made automatically during our simulation period.

Large-scale simulation

Figure 11 presents the energy consumption with six different offloading scenarios in the case of 128×128 grid cells. We assume each cell has only one base station. In our presentation, a boxplot shows min, 25 percentile, median, 75 percentile, and max; an empty rounded dot inside of each box presents median. As a ground truth, we first show local execution of mobile health applications with no offloading capability, resulting in no energy saving. The mobility is generated by the VanetMobisim simulator. By applying our proposed offloading technique with concurrent request capability, we see clear improvement of time and energy saving in a static viewpoint. From O2 to O5 scenarios, we apply parallelism with the different number of concurrent requests, resulting in slightly 2× more energy saving compared to local execution only. This means applying our optimization techniques saves lots of energy by utilizing cloud.

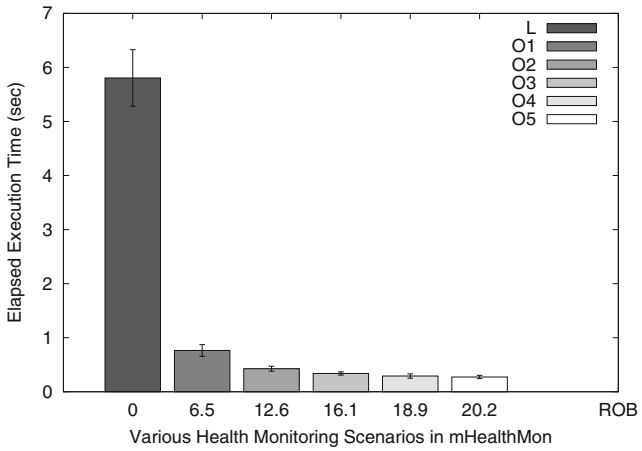

Fig. 9 The average execution time of mHealthMon with various offloading scenarios are measured and presented with 95 percent confidence intervals

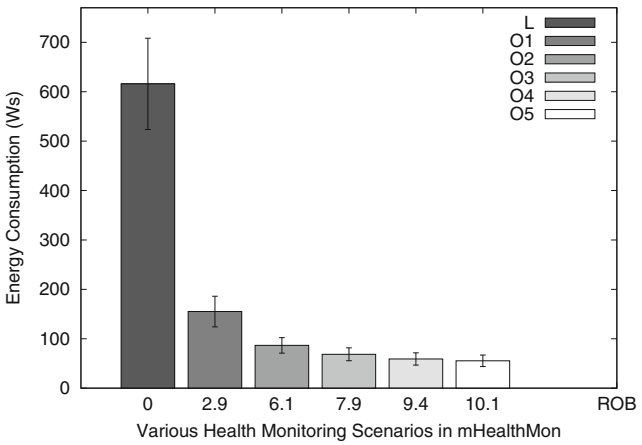

Fig. 10 The average energy consumption of mHealthMon with various offloading scenarios are measured and presented with 95 percent confidence intervals

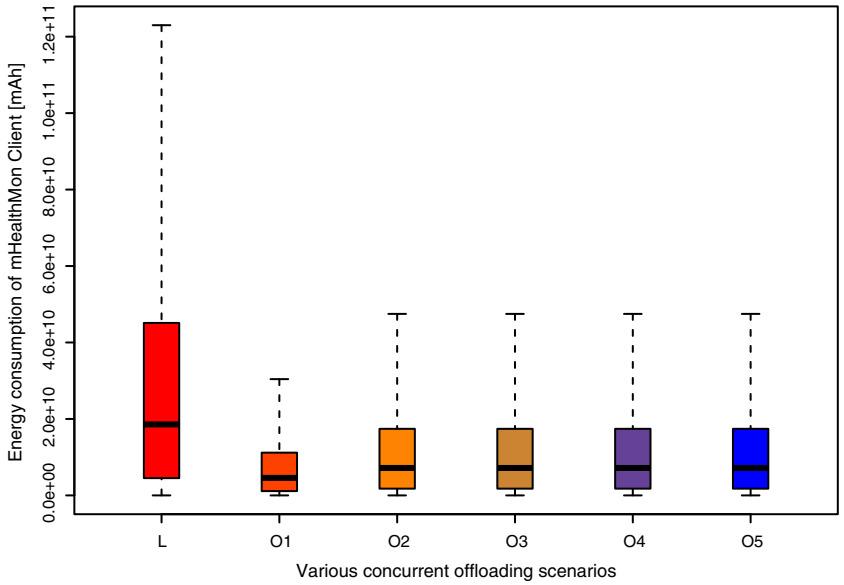

Fig. 11 Energy consumption [Watts] for the simulation period is presented. The total amount of energy consumed by each cell is averaged

Conclusion

This paper presents an attempt to decompose the very complex mobile health monitoring system whose layer in the system corresponds to a decomposed subproblem, and interfaces between them are quantified as functions of the optimization variables in order to orchestrate the subproblems. By simulations, we showed the proposed system can perform up to 10.1 times more energy-efficient and 20.2 times faster compared to a standalone mobile health application. We also compared our work with non-parallel offloading schemes such as MAUI [5] and CloneCloud [3], resulting in at least 2 times better both in time and energy. For the future work, we plan to study energy issues in resource allocation of cloud services. Particularly, the cooling down machines in data center has been a major research topic. By modeling air flow in data center, we may generate a better power management and job scheduling strategy.

Acknowledgments This work was in part funded by Samsung Advanced Institute of Technology under grant numbers 290112465.

References

1. Lv, Z., Xia, F., Wu, G., Yao, L., and Chen, Z., iCare: A Mobile Health Monitoring System for the Elderly. In: *GreenCom*, pp. 699–705, 2010.
2. U.S. Census Bureau. TIGER. Available at www.census.gov/geo/www/tiger.
3. Chun, B-G., Ihm, S., Maniatis, P., Naik, M., and Patti, A., CloneCloud: Elastic Execution between Mobile Device and Cloud. *EuroSys*, pp. 301–314, 2011.
4. Chowdhury, A. R., and Falchuk, B., MediAlIy: A Provenance-Aware Remote Health Monitoring Middleware. In: *PerCom*, pp. 125–134, 2010.
5. Cuervoy, E., Balasubramanianz, A., Cho, D-K., Wolmanx, A., Saroiux, S., Chandrax, R., and Bahl, P., MAUI: Making Smartphones Last Longer with Code Offload. In: *MobiSys*, pp. 49–62, 2010.
6. Manku, G. S., Bawa, M., and Raghavan, P., Symphony: Distributed Hashing in a Small World. *USITS*, Vol 4, pp. 10–10, 2003.
7. Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N., The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Computing*, 8(4):14–23, 2009.
8. Lukowicz, P., AMON: A Wearable Medical Computer for High Risk Patients. In: *ISWC*, pp. 133–134, 2002.
9. Lubrin, E., Lawrence, E., and Navarro, K. E., Motecare: an adaptive smart ban health monitoring system. In: *BioMed*, pp. 60–67, 2006.
10. Kim, Y. B., Kim, M., and Lee, Y., COSMOS: a middleware platform for sensor networks and a u-healthcare service. In: *SAC*, pp. 512–513, 2008.
11. Rodriguez, J., Goni, A., and Illarramendi, A., Real-Time Classification of ECGs on a PDA, IEEE Transactions on Information Technology. In: *Biomedicine*, Vol. 9, pp. 23–34, 2005.
12. Wu, W., Cao, J., and Zheng, Y., WAITER: A wearable personal healthcare and emergency aid system. In: *PerCom*, pp. 680–685, 2008.
13. Pantelopoulos, A., and Bourbakis, N. G., Prognosis - A Wearable Health Monitoring System for People at Risk: Methodology and Modeling. In T-ITB, 14(3):613–21, 2010.
14. Gay, V., Leijdekkers, P., and Barin, E., A Mobile Rehabilitation Application for the Remote Monitoring of Cardiac Patients after a Heart Attack or a Coronary Bypass Surgery. In PETRA, No. 21, 2009.
15. Grossetete, P., ArchRock Energy Optimizer: a case study on IP WSN data for energy and environmental monitoring. In DMSN, No. 2, 2009.
16. Ahnn, J. H., Lee, U., and Moon, H. J., GeoServ: A Distributed Urban Sensing Platform. *CCGRID*, pp. 164–173, 2011.
17. Reddy, S., Chen, G., Fulkerson, B., Kim, S. J., Park, U., Yau, N., Cho, J., Sensor-Internet Share and Search – Enabling Collaboration of Citizen Scientists. In: *DSI*, pp. 11–16, 2007.

18. Nath, S., Liu, J., and Zhao, F., SensorMap for Wide-Area Sensor Webs. In *IEEE Computer Magazine*, 40(7):90–93, 2007.
19. Buhler, J., and Wunder, G., An optimization framework for heterogeneous access management. *WCNC*, pp. 2525–2530, 2009.
20. Harrell, F., Regression modeling strategies. *Springer*, 2001.
21. Blau, I., Wunder, G., Karla, I., and Sigle, R., Decentralized Utility Maximization in Heterogeneous Multicell Scenario with Interface Limited Orthogonal Air Interfaces. *EURASIP*, No. 2, 2009.
22. Härri, J., Fiore, M., and Fethi, F., VanetMobiSim: Generating Realistic Mobility Patterns. *VANET*, pp. 96–97, 2006.
23. Zhang, X., Jeong, S., Kunjithapatham, A., and Simon Gibbs. Towards an Elastic Application Model for Augmenting Computing Capabilities of Mobile Platforms. In *Mobileware*, 16(3):270–284, 2010.
24. Giurgiu, I., Riva, O., Juric, D., Krivulev, I., and Alonso, G., Calling the Cloud: Enabling Mobile Phones as Interfaces to Cloud Applications. In *Middleware*, pp. 83-102, 2009.
25. Boyd, S., and Vandenberghe, L., Convex Optimization. *Cambridge University Press*, New York, 2004.