Toward Sensor-Cloud Integration as a Service: Optimizing Three-tier Communication in Cloud-integrated Sensor Networks

Dũng H. Phan and Junichi Suzuki
Department of Computer Science
University of Massachusetts, Boston
Boston, MA, 02125-3393, USA
{phdung, jxs}@cs.umb.edu

Shingo Omura and Katsuya Oba
OGIS International, Inc.
San Mateo, CA 94402, USA
{omura, oba}@ogis-international.com

ABSTRACT

This paper proposes a cloud-integrated sensor networking architecture, called Sensor-Cloud Integration Platform as a Service (SC-iPaaS), which hosts virtualized sensors in clouds and operates physical sensors through their virtual counterparts. SC-iPaaS performs push-pull hybrid communication between three layers: cloud, edge and sensor layers. This paper formulates an optimization problem for SC-iPaaS to seek the optimal data transmission rates for individual nodes and examines evolutionary optimization with respect to multiple conflicting objectives subject to given constraints. Simulation results show that multiobjective analysis is critical in configuring and operating three-tier push-pull hybrid communication in SC-iPaaS.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Design, Algorithms

Keywords

Wireless sensor networks, cloud computing, multiobjective optimization, evolutionary algorithms

1. INTRODUCTION

The advances in minuscule sensor devices, mobile computing and cloud computing offer tremendous opportunities to seamlessly integrate the physical world and the cyber space. The notion of cyber-physical systems (CPSs) is aimed at a computing paradigm to sense, understand, intervene in, control and/or predict physical phenomena, events and processes. Cloud-integrated CPS (CCPS) refers to virtually representing physical system components (e.g., sensors, actuators, robots and other devices) in clouds, accessing (e.g., monitoring, actuating and navigating) those physical components through their virtual representations, and processing/managing the sheer amount of data collected from physical components in clouds in a scalable, on-demand, efficient and/or reliable manner.

This paper proposes a CCPS architecture, called Sensor-Cloud Integration Platform as a Service (SC-iPaaS), and examines communication optimization in SC-iPaaS. SC-iPaaS is a three-tier architecture that seamlessly integrates the *sensor*, *edge* and *cloud* layers. The sensor layer consists of sensor nodes embedded in the physical environment. The edge layer consists of sink nodes that collect sensor data from sensor nodes in the physical environment. The cloud layer consists of cloud computing environments that host *virtual sensors*, which are virtualized counterparts (or software counterparts) of physical sensors in the sensor layer. Virtual sensors collect sensor data from sink nodes in the edge layer and store those data for future use. Clouds also host cloud applications, which obtain sensor data from virtual sensors and aid users to monitor physical phenomena, events and processes in the physical environment.

SC-iPaaS performs *push-pull hybrid communication* between its layers. Individual sensor nodes periodically transmit (or push) sensor data to sink nodes, which in turn forward (or push) incoming sensor data periodically to virtual sensors. When a virtual sensor does not have sensor data that a cloud application requires, it obtains (or pulls) that data from a sink node or a sensor node. This push-pull communication scheme is intended to make as much sensor data as possible available for cloud applications by taking advantage of push communication while allowing virtual sensors to pull any missing data anytime in an on-demand manner.

In order to properly operate three-tier push-pull communication in SC-iPaaS, it is important to optimize various parameters, namely data transmission rate of each sensor node and sink node. Therefore, this paper formulates a communication optimization problem in SC-iPaaS. It is to seek the optimal data transmission rate for each sensor node and sink node with respect to multiple optimization objectives such as sensor data yield (sensor data availability) for cloud applications, bandwidth consumption between the cloud layer and the edge layer and energy consumption of sensor nodes in the sensor layer. This paper heuristically approach this

BODYNETS 2013, September 30-October 02, Boston, United States

DOI 10.4108/icst.bodynets.2013.253639

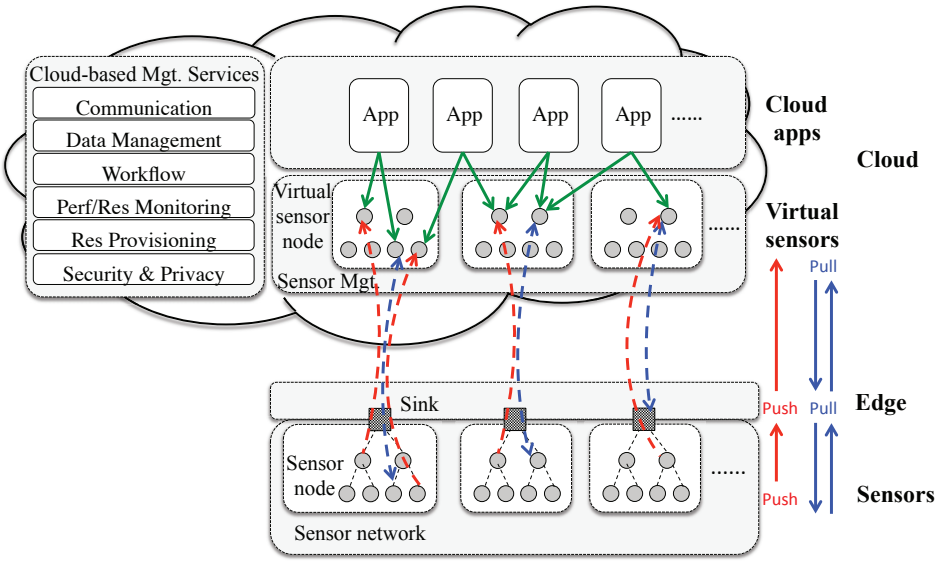


Figure 1: A Push-Pull Hybrid Communication Architecture for Cloud-integrated Sensor Networks

multiobjective problem with an evolutionary optimization algorithm. Simulations are carried out to examine Pareto-optimal communication configurations (parameter settings) against data request patterns placed by cloud applications. Simulation results show that multiobjective analysis is critical in configuring and operating three-tier push-pull hybrid communication in cloud-integrated sensor networks.

2. SC-IPAAS AND ITS APPLICATIONS

Figure 1 shows an architectural overview of SC-iPaaS. SC-iPaaS consists of the following three tiers.

Sensor Layer: operates one or more wireless networks of stationary sensor nodes embedded in the physical environment. Each network is assumed to be heterogeneous; it has different types of sensor devices such as air temperature sensors, humidity sensors and barometric pressure sensors. Sensor nodes are battery-operated or solar-powered; they have limited energy supplies. In each sensor network, nodes form a particular topology (e.g., tree, star or mesh topology). In Figure 1, sensor networks use a tree topology. Nodes periodically read sensors and transmit (or push) sensor data to a special node, called sink node, on a hop-by-hop manner through a given network topology. Different sensor nodes have different data transmission rates.

Edge Layer: is a collection of sink nodes, each of which participates in a certain sensor network and receives sensor data periodically from individual nodes in the network. Each sink node stores incoming sensor data in its memory space and then transmits (or pushes) them periodically to the cloud layer. It maintains the mappings between physical sensors and virtual sensors. In other words, it knows the origins and destinations of sensor data. Different sink nodes have different data transmission rates. A sink node's data transmission rate can be different from the ones of sensor nodes in the same network. Note that each sensor network operates one or more sink nodes. In Figure 1, one sink node is operated in each sensor network. Depending on application domains, sink nodes may have limited energy supplies through batteries and solar panels or they may have infinite energy supplies.

In addition to pushing sensor data to a virtual sensor, each sink node receives a "pull" request from a virtual sensor when the virtual sensor does not have data that a cloud application(s) requires (Figure 1). If the sink node has the requested data in its memory, it returns that data. Otherwise, it issues a pull request to a sensor node that is responsible for the requested data. Upon receiving the pull request, the sensor node reads a sensor and returns sensor data.

Cloud Layer: operates on one or more clouds to host end-user applications and management services for the applications. Applications are operated on virtual machines in clouds. Users are assumed to place continuous sensor data queries on virtual sensors via cloud applications in order to monitor the physical environment. If a virtual sensor already has data that an application queries, it returns that data. If a query does not match, the virtual sensor issues a pull request and sends it to a sink node. Each query is assumed to have a relative time window within which an application requires particular sensor data. While push communication carries out one-way upstream travel of sensor data, pull communication incurs a round trip for requesting sensor data and receiving that data (Figure 1).

Cloud-based management services offer common functionalities to implement and operate applications (Figure 1). This paper focuses on the following two services.

- *Sensor manager*: virtualizes physical heterogeneous sensors in a unified way by abstracting away their low-level operational details. Cloud applications always access physical sensors through virtual sensors; for example, collecting sensor data with a pull request and sending control signals (e.g., turning on/off sensors and setting data transmission rates).
- *Communication manager*: is responsible for push-pull hybrid communication between different layers. It is assumed to operate on top of certain publish/subscribe communication middleware such as TinyDDS [4]. A

key component in this manger is *communication optimizer*, which this paper focuses on to seek the optimal data transmission rates for sensor and sink nodes with respect to multiple optimization objectives.

While SC-iPaaS can be effectively applicable to a wide range of applications, this paper discusses two specific example applications. The first example is a physiological monitoring application in pervasive healthcare for inpatients and homebound patients [10, 17]. This application assumes per-patient wireless networks of in-body and/or on-body sensors for, for example, heart rate, blood pressure, oxygen saturation, body temperature, respiratory rate, blood coagulation and galvanic skin response. Those sensors are wirelessly connected to a dedicated per-patient device or a patient's computer (e.g., cell phone, tablet machine or laptop computer) that serves as a sink node. Real-time physiological sensor data are periodically pushed to virtual sensors in clouds so that clinicians, hospital nurses and visiting nurses can share the data for clinical observation and intervention. When an anomaly is found in physiological data, clinical staff may pull extra data in a higher temporal resolution to better understand a patient's medical condition. Given a sufficient amount of data, they may perform clinical interventions (for inpatients) or dispatch visiting nurses or ambulances (for homebound patients).

The second example application of SC-iPaaS is in-situ environmental monitoring in public beaches [9,16,18]. A sensor network is deployed in each beach area with water quality sensors and other environmental sensors (e.g., bacteria, pH, dissolved oxygen, water temperature, humidity, barometric pressure, wind speed, wind direction and rain fall sensors). A sink node(s) in each beach area periodically receive sensor data from individual sensor nodes and pushes them to virtual sensors. Users of cloud applications include local government officials for public health, environmental quality, recreation and tourism. Cloud applications are intended to indicate and localize episodic changes in, for example, water quality, tidal level and climate at beaches. When certain environmental properties are detected as predictors of unhealthy or dangerous conditions, applications allow local government officials to pull extra sensor data in a higher spatiotemporal resolution so that they can be better informed to decide beach closures, signal warnings to the general public and dispatch public safety officials to beaches.

3. COMMUNICATION OPTIMIZATION IN SC-IPAAS: PROBLEM STATEMENT

This section describes an optimization problem to seek the optimal data transmission rates for sensor and sink nodes in SC-iPaaS. The following notations are used to state the optimization problem.

- $S = \{s_1, s_2, ..., s_i, ..., s_M\}$ is the set of M sensor nodes in sensor networks. ν_{s_i} denotes the data transmission rate for the i-th sensor node (s_i) to push sensor data to its corresponding sink node. The rate is measured as the number of sensor data transmitted per a unit time. d_i indicates the size of single sensor data that s_i generates and transmits to a sink node. h_i denotes the shortest logical distance (i.e., hop count) from s_i to its corresponding sink node.
- ν_{e_i} denotes the data transmission rate for a sink node to push sensor data receiving from the i-th sensor node (s_i). ν_{e_i} and ν_{s_i} are not necessarily equal.
- W indicates a relative time window that SC-iPaaS considers to monitor sensor data requests from cloud applications and compute its communication performance with respect to optimization objectives.
- $R_i = \{r_{i1}, r_{i2}, ..., r_{ij}, ..., r_{i|R_i|}\}$ is the set of all sensor data requests that cloud applications issue to the virtual counterpart of s_i (s'_i) during the time period of W in the past. r_{ij} denotes the j-th request to the i-th virtual sensor s'_i. Each request is characterized by its time stamp (t_{ij}) and time window (w_{ij}). It requests all sensor data available in the time interval $[t_{ij} - w_{ij}, t_{ij}]$. If s'_i has at least one data in $[t_{ij} - w_{ij}, t_{ij}]$, it returns those data to a cloud application. Otherwise, it issues a pull request to a sink node.
- $R^e_i \subset R_i$ is the set of sensor data requests for which the virtual sensor s'_i has no data. This means that $|R^e_i|$ indicates the number of pull requests that s'_i issues to a sink node. In other words, $R_i \setminus R^e_i$ indicates the requests that s'_i can fulfill.
- $R^s_i \subset R^e_i \subset R_i$ is the set of sensor data requests for which the sink node for s_i do not have data. This means that $|R^s_i|$ indicates the number of pull requests that the sink node issues to s_i. In other words, $R^e_i \setminus R^s_i$ indicates the requests that the sink node can fulfill.

This paper considers three optimization objectives: bandwidth consumption between the edge and cloud layers (f_B), energy consumption of physical sensors (f_E) and data yield for cloud applications (f_D). The first two objectives are to be minimized while the third is to be maximized.

The bandwidth consumption objective (f_B) is defined as the total amount of data transmitted per a unit time between the edge and cloud layers. This objective impacts the payment for bandwidth consumption based on a cloud operator's pay-per-use billing scheme. It also impacts the lifetime of sink nodes if they are battery-operated or solar-powered. f_B is computed as follows.

$$f_B = \sum_{i=1}^{M}(\nu_{ei} \times d_i) + \frac{1}{W}\sum_{i=1}^{M}\sum_{r_{ij} \in R^e_i}(\phi_{ij} \times d_i + d_r) \qquad (1)$$

The first and second terms indicate the bandwidth consumption by one-way push communication from the edge layer to the cloud layer and two-way pull communication between the cloud and edge layers, respectively. ϕ_{ij} denotes the number of sensor data that the request r_{ij} can collect in the time interval $[t_{ij} - w_{ij}, t_{ij}]$. d_r indicates the size of a single pull request from the cloud layer to the edge layer. It is constant for all sensor nodes.

The energy consumption objective (f_E) is defined as the total amount of energy that sensor nodes consume for data transmissions during the time period of W. This objective impacts the lifetime of sensor nodes and sensor networks. It is computed as follows.

$$f_E = \sum_{i=1}^{M}(h_i \times e_t \times d_i \times \nu_{si} \times W) + \sum_{i=1}^{M}\sum_{r_{ij} \in R^e_i}\Big(h_i \times e_t \times (d_i + d'_r)\Big) \qquad (2)$$

The first and second terms indicate the energy consumption by one-way push communication from the sensor layer to the edge layer and two-way pull communication between the edge layer and the sensor layer, respectively. e_t denotes the amount of energy that a unit amount of data consumes to travel from a sensor node to its neighboring node. d'_r denotes the size of a single pull request from the edge layer to the sensor layer. e_t and d_r are constant for all sensor nodes.

The data yield objective (f_Y) is defined as the total amount of data that cloud applications gather for their users. This objective impacts the informedness and situation awareness for application users. It is computed as follows.

$$f_Y = \sum_{i=1}^{M} \sum_{r_{ij} \in R_i} \phi_{ij} \tag{3}$$

In SC-iPaaS, optimization objectives conflict with each other. For example, the data yield objective conflicts with the other two objectives. Maximizing data yield means increasing data transmission rates for sensor and sink nodes. This increases bandwidth consumption and energy consumption. Similarly, the energy consumption objective conflicts with the data yield objective. Minimizing energy consumption means reducing data transmission rates for sensor nodes. This can reduce data yield. Given these conflicting objectives, this paper seeks the optimal trade-off (i.e., Pareto-optimal) configurations for data transmission rates in SC-iPaaS.

SC-iPaaS considers two constraints in its optimization process. The first constraint (C_E) is the upper limit for energy consumption (f_E):

$$f_E < C_E \tag{4}$$

The constraint violation in energy consumption (g_E) is computed as follows where $I_E = 1$ if $f_E > C_E$; otherwise $I_E = 0$.

$$g_E = I_E \times (f_E - C_E) \tag{5}$$

The second constraint (C_Y) is the lower limit for data yield (f_Y):

$$f_Y > C_Y \tag{6}$$

The constraint violation in data yield (g_Y) is computed as follows where $I_Y = 1$ if $f_Y < C_Y$; otherwise $I_Y = 0$.

$$g_Y = I_Y \times (C_Y - f_Y) \tag{7}$$

4. COMMUNICATION OPTIMIZER IN SC-IPAAS

SC-iPaaS leverages an evolutionary multiobjective optimization algorithm (EMOA) for its communication optimizer. The algorithm iteratively evolves the population of solution candidates, called *individuals*, through several operators (e.g., crossover, mutation and selection operators) toward the Pareto-optimal solutions in the objective space.

The EMOA in SC-iPaaS is intended to search Pareto-optimal solutions that are equally distributed in the objective space because there exits no single optimal solution under conflicting objectives but rather a set of alternative solutions of equivalent quality. Therefore, it can produce both *extreme* data transmission configurations (e.g., the one exhibiting high data yield and high energy consumption) and *balanced* configurations (e.g., the one exhibiting intermediate data yield and energy consumption) at the same time. Given a set of heuristically-approximated Pareto-optimal solutions, an SC-iPaaS operator can examine the trade-offs among them and make a well-informed decision to choose one of them, as the data transmission configuration, according to his/her preferences and priorities. For example, an SC-iPaaS operator can examine how he/she can data yield for energy consumption and determine a particular data transmission configuration that achieves a desirable/comfortable balance of data yield and energy consumption.

Algorithm 1 Optimization Process in SC-iPaaS

```
1: g = 0;
2: P_g = Randomly generated N individuals;
3: while g < MAX-GENERATION do
4:    O_g = ∅;
5:    while |O_g| < N do
6:       p_1 = tournament(P_g)
7:       p_2 = tournament(P_g)
8:       if random() ≤ P_c then
9:          {o_1, o_2} = crossover(p_1, p_2)
10:      end if
11:      if (random() ≤ P_m) then
12:         o_1 = mutation(o_1)
13:      end if
14:      if random() ≤ P_m then
15:         o_2 = mutation(o_2)
16:      end if
17:      O_g = {o_1, o_2} ∪ O_g
18:   end while
19:   R_g = P_g ∪ O_g
20:   F = sortByDominationRanking(R_g)
21:   P_{g+1} = {∅}
22:   i = 1
23:   while |P_{g+1}| + |F_i| ≤ N do
24:      P_{g+1} = P_{g+1} ∪ F_i
25:      i = i + 1
26:   end while
27:   sortByCrowdingDistance(F_i)
28:   P_{g+1} = P_{g+1} ∪ F_i[1 : (N − |P_{g+1}|)]
29:   g = g + 1
30: end while
```

In order to seek Pareto optimality, the notion of *dominance* plays an important role [20]. An individual i is said to dominate an individual j (denoted by $i \succ j$) if both of the following conditions are hold.

- i's objective values are superior than, or equal to, j's in all objectives.
- i's objective values are superior than j's in at least one objectives.

In SC-iPaaS, each individual represents a particular data transmission configuration, which is a set of data transmission rates for all sensor and sink nodes (Figure 2).

ν_{e1}	ν_{e2}	ν_{e3}	....	ν_{eM-1}	ν_{eM}
ν_{s1}	ν_{s2}	ν_{s3}	....	ν_{sM-1}	ν_{sM}

Figure 2: Individual Representation

Algorithm 1 shows the evolutionary optimization process in SC-iPaaS. It follows the algorithmic structure in NSGA-

II [6]. At the *0*-th generation, N individuals are randomly generated as the initial population $\mathcal{P}_0$ (Line 2). Each of them has randomly-selected data transmission rates for sensor and sink nodes.

In each generation (g), two parent individuals (p_1 and p_2) are selected from the current population $\mathcal{P}_g$ with binary tournaments (Lines 6 and 7). A binary tournament randomly takes two individuals from $\mathcal{P}_g$, compares them based on the notion of *constrained dominance*, and chooses a superior one as a parent. Given the notion of constrained dominance, an individual i is said to *constrained-dominate* an individual j, if any of the following conditions is hold:

- i is feasible j is not.
- i and j are both feasible, and i dominates j in the objective space.
- Both i and j are infeasible, but i dominates j in the constraint space.
- Both i and j are infeasible and both of the following conditions are hold
  1. i's constraint violation are less than, or equal to, j's in all constraints.
  2. i's constraint violation are less than j's in at least one constraint.

With the crossover rate P_c, two parents reproduce two offspring through crossover (Lines 8 to 10). Then, mutation occurs on each offspring (Lines 11 to 16). It assigns a new randomly-chosen data transmission rate to each node with the mutation rate P_m. The binary tournament, crossover and mutation operators are executed repeatedly on $\mathcal{P}_g$ to reproduce N offspring. The offspring ($\mathcal{O}_g$) are combined with the parent population $\mathcal{P}_g$ to form $\mathcal{R}_g$ (Line 19).

Environmental selection follows reproduction. Best N individuals are selected from $2N$ individuals in $\mathcal{R}_g$ as the next generation population ($\mathcal{P}_{g+1}$). First, the individuals in $\mathcal{R}_g$ are ranked based on the constrained dominance relationships among them. Non-dominated individuals are on the first rank. The i-th rank consists of the individuals dominated only by the individuals on the $(i-1)$-th rank. Ranked individuals are stored in $\mathcal{F}$ (Line 20). $\mathcal{F}_i$ contains the i-th rank individuals.

Then, the individuals in $\mathcal{F}$ move to $\mathcal{P}_{g+1}$ on a rank by rank basis, starting with $\mathcal{F}_1$ (Lines 23 to 26). If the number of individuals in $\mathcal{P}_{g+1} \cup \mathcal{F}_i$ is less than N, $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. Otherwise, a subset of $\mathcal{F}_i$ moves to $\mathcal{P}_{g+1}$. The subset is selected based on the crowding distance metric, which measures the distribution (or diversity) of individuals in the objective space [6] (Lines 27 and 28). The metric computes the distance between two closest neighbors of an individual in each objective and sums up the distances associated with all objectives. A higher crowding distance means that an individual in question is more distant from its neighboring individuals in the objective space. In Line 27, the individuals in $\mathcal{F}_i$ are sorted from the one with the highest crowding distance to the one with the lowest crowding distance. The individuals with higher crowding distance measures have higher chances to be selected to $\mathcal{P}_{g+1}$ (Line 28).

5. SIMULATION EVALUATION

This section evaluates communication optimization in SC-iPaaS through simulations.

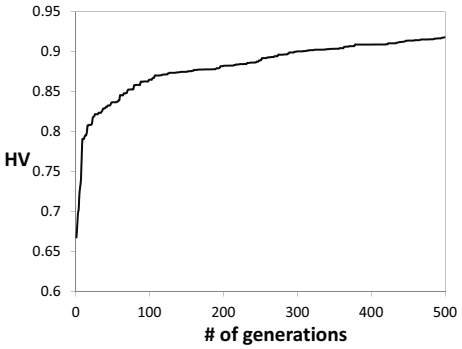


Figure 3: Changes in Hypervolume over Generations

5.1 Simulation Configurations

Simulations are configured with a set of parameters shown in Table 1. A network of 50 sensor nodes is simulated to connect to a cloud through a sink node. Cloud applications issue 100,000 sensor data requests a day. Those requests are uniformly distributed over 50 virtual sensors. Table 2 shows three types of sensors used in simulations. Half of 50 sensors are temperature sensors. The size of each temperature data is 15 bytes. The time window for each temperature data request is randomly generated following a normal distribution with the mean of 300 seconds and the standard deviation of 60 seconds.

Table 1: Simulation Configurations

Parameter	Value	Parameter	Value
Total # of sensors (M)	50	Population size (N)	100
Total # of data requests	100,000	Crossover rate (P_c)	0.9
Simulation time (W)	1 day	Mutation rate (P_m)	0.1

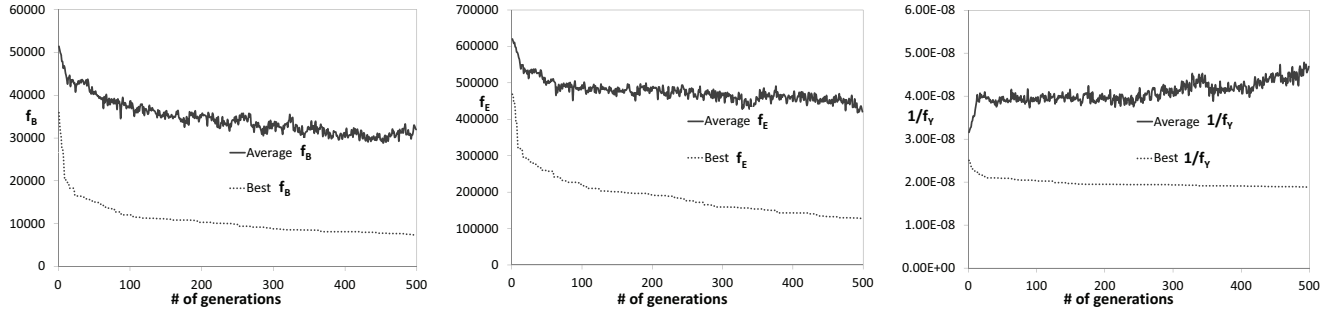
Table 2: Configurations for Sensors and Sensor Data Requests

Sensor type	Quantity	Data size (d_i) (Bytes)	Request time window (w_{ij}) (Seconds)
Temperature	25	15	$\mathcal{N}(300, 60^2)$
Acceleration	15	100	$\mathcal{N}(600, 120^2)$
Sound	10	128	$\mathcal{N}(1800, 360^2)$

5.2 Simulation Results

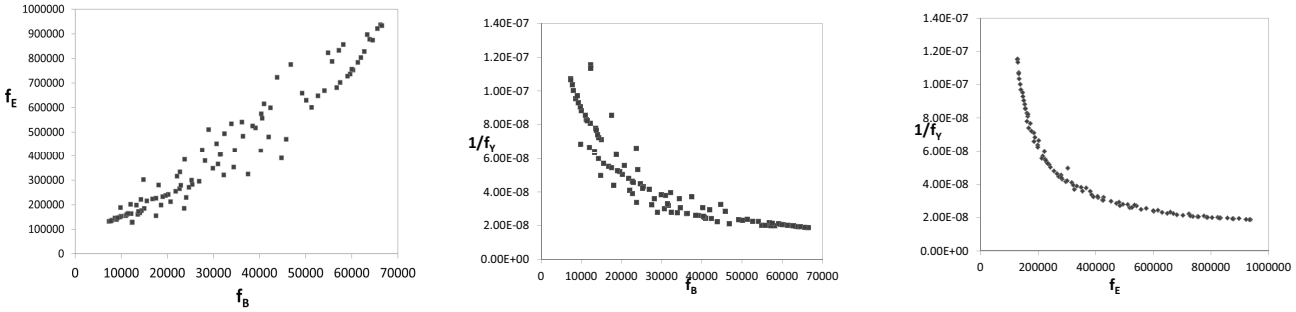
Figure 3 shows how individuals evolve as the number of generations grows when no constraints are specified ($C_E = \infty$ and $C_Y = 0$). It uses the hypervolume (HV) metric [24]. HV measures the union of volumes that non-dominated individuals dominates in the objective space. Thus, HV quantifies the optimality and diversity of non-dominated individuals. A higher HV indicates that non-dominated individuals are closer to the Pareto-optimal front and more diverse in the objective space. As shown in Figure 3, SC-iPaaS rapidly increases its hypervolume measure in the first 10 generations and converges around the 400th generation. At the last generation, all individuals are non-dominated in the population.

Figure 4 shows how the average and the best objective values in the population change over generations when no constraints are specified. As the number of generations grow,



(a) Bandwidth Consumption (f_B) (b) Energy Consumption (f_E) (c) Data Yield (f_Y)

Figure 4: Average Objective Values over Generations



(a) Bandwidth Consumption and Energy Consumption (b) Data Yield and Bandwidth Consumption (c) Data Yield and Energy Consumption

Figure 5: Two-dimensional Objective Spaces

the average objective values of bandwidth consumption and energy consumption decrease while the average data yield increases. However, the best objective value clearly decreases over generations in every objective. Figures 3 and 4 verify that SC-iPaaS allows individuals to efficiently evolve and improve their quality and diversity

Figure 5 shows two-dimensional objective spaces, each of which plots individuals obtained at the last generation. (No constraints are specified.) Figure 5(a) illustrates that bandwidth consumption and energy consumption correlate. Figure 5(b) illustrates that data yield and bandwidth consumption conflict with each other. According to Figure 5(c), data yield and energy consumption conflict as well. SC-iPaaS successfully reveals the relationships among optimization objectives and clearly exhibits the trade-offs among non-dominated individuals

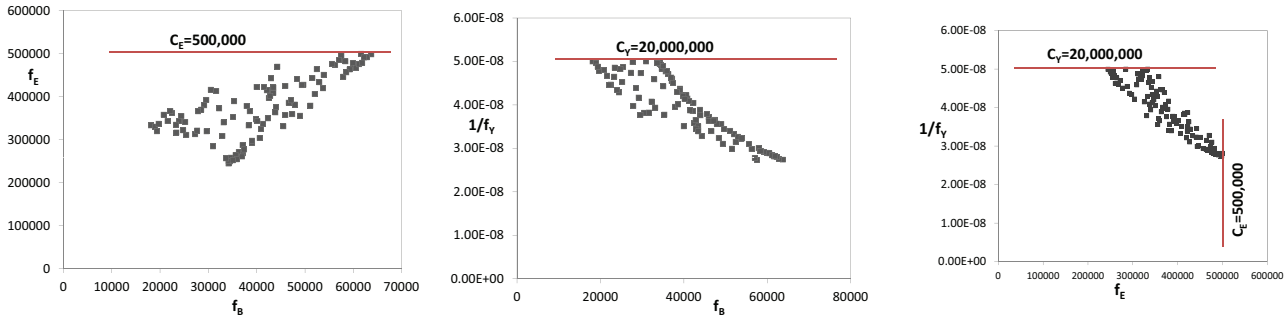
Figure 6 shows two-dimensional objective spaces, each of which plots individuals obtained at the last generation when constraints are specified for data yield and energy consumption ($C_E = 500,000$ and $C_Y = 20,000,000$). At the last generation, all individuals are feasible; their objective values are below given constraint values. In comparison with Figure 5, Figure 6 demonstrates that SC-iPaaS effectively optimizes data transmission configurations subject to given constraints.

Figure 7 shows three-dimensional objective spaces that plot individuals obtained at the last generation with and without constraints. At the last generation, all individuals are non-dominated with each other in both cases. When constraints are given, all individuals are feasible at the last generation. As Figure 7 illustrates, SC-iPaaS evolves individuals in different ways with and without constraints and successfully obtain feasible individuals when constraints are specified.

6. RELATED WORK

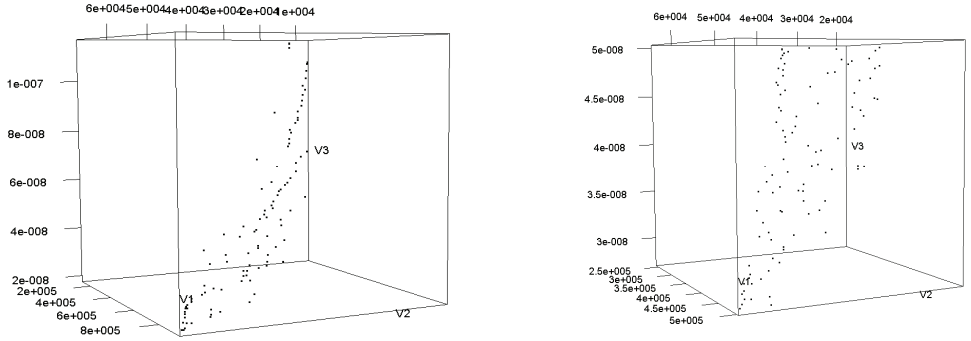
Various architectures and research tools have been proposed for cloud-integrated sensor networks [1, 3–5, 7, 8, 11, 19, 23]. Hassan et al. [11], Aberer et al. [1], Shneidman et al. [8,19] and Boonma et al. [3,4] assume three-tier architectures similar to SC-iPaaS and investigate publish/subscribe communication between the edge layer to the cloud layer. Their focus is placed on push communication. In contrast, SC-iPaaS investigates push-pull hybrid communication between the sensor layer and the cloud layer through the edge layer. Fortino et al. study a three-tier architecture to integrate body area networks with clouds [5, 7]. Yuriyama et al. propose a two-tier architecture that consists of the sensor and cloud layers [23]. The architectures proposed by Fortino et al. and Yuriyama et al. are similar to SC-iPaaS in that they leverage the notion of virtual sensors. However, they do not consider push-pull (nor publish/subscribe) communication. All the above-mentioned work do not consider communication optimization as SC-iPaaS does.

Push-pull hybrid communication has been studied well in sensor networks [2, 12–15, 21]. However, few attempts have been made to investigate it between the edge and cloud



(a) Bandwidth Consumption and Energy Consumption (b) Data Yield and Bandwidth Consumption (c) Data Yield and Energy Consumption

Figure 6: Two-dimensional Objective Space (Data Yield and Energy Consumption) with Two Constraints



(a) Without Constraints ($C_E = \infty$ and $C_Y = 0$) (b) With Constraints ($C_E = 500,000$ and $C_Y = 20,000,000$)

Figure 7: Three-dimensional Objective Spaces

layers in the context of cloud-integrated sensor networks. Unlike existing relevant work, this paper formulates an optimization problem with cloud-specific optimization objectives as well as the ones in sensor networks and examine sensor-to-cloud communication optimization.

Xu et al. propose a three-tier architecture called CEB (Cloud, Edge and Beneath), which is similar to SC-iPaaS, and optimize data transmission rates between layers [22]. CEB runs two optimization algorithms collaboratively: OPT-1 and OPT-2, which optimize data transmission rates between the cloud and edge layers and between the edge and sensor layers, respectively. Optimization is carried out on a sensor node by sensor node basis with respect to a single objective: energy consumption of sensor nodes. In contrast, SC-iPaaS runs a single optimization algorithm for the entire group of sensor nodes and sink nodes simultaneously with respect to multiple conflicting objectives. SC-iPaaS assumes sensor data requests with time windows to heterogeneous sensor networks while CEB assumes requests without time windows to homogeneous networks.

7. CONCLUSIONS

This paper proposes a cloud-integrated sensor networking architecture, called SC-iPaaS, which hosts virtualized sensors in clouds and operates physical sensors through their virtual counterparts. SC-iPaaS performs push-pull hybrid communication between three layers: cloud, edge and sensor layers. This paper formulates an optimization problem for SC-iPaaS to seek the optimal data transmission configurations and approaches the problem with an evolutionary multiobjective optimization algorithm. SC-iPaaS successfully optimizes data transmission configurations with respect to multiple objectives (data yield, bandwidth consumption and energy consumption) subject to given constraints. It also reveals the relationships among objectives and clearly exhibits the trade-offs among different data transmission configurations.

8. REFERENCES

[1] K. Aberer, M. Hauswirth, and A. Salehi. Infrastructure for data processing in large-scale interconnected sensor networks. In *Proc. the 8th IEEE Int'l Conference on Mobile Data Management*, 2007.

[2] P. Boonma, Q. Han, and J. Suzuki. Leveraging biologically-inspired mobile agents supporting composite needs of reliability and timeliness in sensor applications. In *Proc. IEEE Int'l Conf. on Frontiers in the Convergence of Biosci. and Info. Tech.*, 2007.

[3] P. Boonma and J. Suzuki. Toward interoperable publish/subscribe communication between wireless

sensor networks and access networks. In *Proc. IEEE Int'l Workshop on Information Retrieval in Sensor Networks*, 2009.

[4] P. Boonma and J. Suzuki. TinyDDS: An interoperable and configurable publish/subscribe middleware for wireless sensor networks. In A. Hinze and A. Buchmann, editors, *Principles and Apps. of Dist. Event-Based Systems*, chapter 9. IGI Global, 2010.

[5] A. Cuzzocrea, G. Fortino, and O. Rana. Managing data and processes in cloud-enabled large-scale sensor networks: State-of-the-art and future research directions.

[6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization:NSGA-II. In *Proc. of Int'l Conference on Parallel Problem Solving from Nature*, 2001.

[7] G. Fortino, M. Pathan, and G. D. Fatta. BodyCloud: Integration of cloud computing and body sensor networks. In *Proc. IEEE Int'l Conference on Cloud Computing Technology and Science*, 2012.

[8] M. Gaynor, M. Welsh, S. Moulton, A. Rowan, E. LaCombe, and J. Wynne. Integrating wireless sensor networks with the grid. *IEEE Internet Computing*, July/August 2004.

[9] J. Goldman, N. Ramanathan, R. Ambrose, D. A. Caron, D. Estrin, J. C. Fisher, R. Gilbert, M. H. Hansen, T. C. Harmon, J. Jay, W. J. Kaiser, G. S. Sukhatme, and Y.-C. Tai. Distributed sensing systems for water quality assessment and management. Technical report, Woodrow Wilson Int'l Center for Scholars, 2007.

[10] Y. Hao and R. Foster. Wireless body sensor networks for health-monitoring applications. *Physiological Measurement*, 29(11), 2008.

[11] M. M. Hassan, B. Song, and E.-N. Huh. A framework of sensor-cloud integration opportunities and challenges. In *Proc. the 3rd ACM Int'l Conference on Ubiquitous Info. Mgt. and Comm.*, 2009.

[12] S. Kapadia and B. Krishnamachari. Comparative analysis of push-pull query strategies for wireless sensor networks. In *Proc. International Conference on Distributed Computing in Sensor Systems*, 2006.

[13] W. C. Lee, M. Wu, J. Xu, and X. Tang. Monitoring top-k query in wireless sensor networks. In *Proc. IEEE International Conference on Data Engineering*, 2006.

[14] M. Li, D. Ganesan, and P. Shenoy. PRESTO: Feedback-driven data management in sensor networks. In *Proc. USENIX Symposium on Networked Systems Design and Implementation*, 2006.

[15] W. Liu, Y. Zhang, W. Lou, and Y. Fang. Managing wireless sensor networks with supply chain strategy. In *Proc. Int'l Conference on Quality of Service in Heterogeneous Wired/Wireless Networks*, 2004.

[16] B. O'Flynn, F. Martinez-Catala, S. Harte, C. O'Mathuna, J. Cleary, C. Slater, F. Regan, D. Diamond, and H. Murphy. SmartCoast: A wireless sensor network for water quality monitoring. In *Proc. IEEE Conference on Local Computer Networks*, 2007.

[17] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers. A review of wearable sensors and systems with application in rehabilitation. *Journal of Neuroengineering and Rehabilitation*, 9(21), 2012.

[18] L. A. Seders, C. A. Shea, M. D. Lemmon, P. A. Maurice, and J. W. Talley. LakeNet: An integrated sensor network for environmental sensing in lakes. *Environmental Engineering Science*, 24(2), 2007.

[19] J. Shneidman, P. Pietzuch, J. Ledlie, M. Roussopoulos, M. Seltzer, and M. Welsh. Hourglass: An infrastructure for connecting sensor networks and applications. Technical report, Harvard University, TR-21-04, 2004.

[20] N. Srinivas and K. Deb. Multiobjective function optimization using nondominated sorting genetic algorithms. *Evol. Computat.*, 2(3), 1995.

[21] H. Wada, P. Boonma, and J. Suzuki. Chronus: A spatiotemporal macroprogramming language for autonomic wireless sensor networks. In N. Agoulmine, editor, *Autonomic Network Mgt. Principles: From Concepts to Applications*, chapter 8. Elsevier, 2010.

[22] Y. Xu, S. Helal, M. Thai, and M. Scmalz. Optimizing push/pull envelopes for energy-efficient cloud-sensor systems. In *Proc. the 14th ACM Int'l Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2011.

[23] M. Yuriyama and T. Kushida. Sensor-cloud infrastructure - physical sensor management with virtualized sensors on cloud computing. In *Proc. the 13th Int'l Conf. on Network-Based Info. Sys.*, 2010.

[24] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms: A comparative study. In *Proc. Int'l Conf. on Parallel Problem Solving from Nature*, 1998.