www.ietdl.org

Published in IET Wireless Sensor Systems
Received on 15th June 2013
Revised on 29th August 2013
Accepted on 1st October 2013
doi: 10.1049/iet-wss.2013.0073

ISSN 2043-6386

Optimal gateway selection in sensor–cloud framework for health monitoring

Sudip Misra[1], *Samaresh Bera*[1], *Ayan Mondal*[1], *Reena Tirkey*[1], *Han-Chieh Chao*[2], *Samiran Chattopadhyay*[3]

[1]*School of Information Technology, Indian Institute of Technology, Kharagpur 721302, India*
[2]*Department of Computer Science and Information Engineering, National Ilan University, I-Lan 260, Taiwan*
[3]*Department of Information Technology, Jadavpur University, Salt Lake, Kolkata 700098, India*
E-mail: hcc@niu.edu.tw

Abstract: Sensor–cloud computing is envisioned to be one of the key enabling technologies for remote health monitoring. Integration of sensed data into cloud applications in sensor–cloud will help in real-time monitoring of patients over geographically distributed locations. In this study, the authors study the optimal gateway selection problem in sensor–cloud framework for real-time patient monitoring system by using a zero-sum game model. In the proposed model, a gateway acts as the first player, and chooses the strategy based on the available bandwidth, whereas a user request acts as the second player, and follows the strategy chosen by the first player. The authors evaluate the execution time for selecting the optimal gateway through which the sensed data will be fetched to the cloud platform. In addition, the authors show how user requests are serviced by the gateways to access data from cloud platform optimally. The authors also show that by using the proposed approach, the execution time decreases, thereby helping in forming a reliable, efficient and real-time architecture for health monitoring.

1 Introduction

A sensor–cloud architecture conceptually integrates cloud infrastructure with sensor networks, thereby enabling real-time monitoring of data-intensive applications (such as healthcare) that are typically spread over geographically distributed locations [1]. Sensor networks are popularly used for deploying health-related applications such as monitoring patients with blood sugar, blood pressure and sleep activity pattern monitoring (SAPM) [2, 3]. In such an application, the health centre takes necessary decisions according to the sensed data from the patients. It is a difficult task to monitor the health status remotely, when a patient moves randomly. Hence, an efficient computing mechanism is necessary to monitor the health status of the patients when they are mobile. The data-intensive, time-varying requirements of the sensor networks can benefit from the intricate integration of the computational and storage resources offered by the cloud computing applications for big-data processing [4]. Thus, sensor–cloud platforms are increasingly become popular in healthcare.

Standalone sensor networks have some inherent challenges [2, 5, 6], which can be undertaken by sensor–cloud infrastructures.

1. *Data management*: One of the major problems of sensor networks is data management. Storing and accessing sensed data from sensor networks is efficient with sensor–cloud.
2. *Resource utilisation*: The cloud computing infrastructure provides resource sharing facility among the users. This information sharing mechanism can be implemented around the sensor network environments, and thus, same data can be accessible from geographically distributed regions at the same time.
3. *High utility cost*: Some analysis such as detecting faulty sensors, removing erroneous readings and fusing noisy measurements from several sensors is challenging and expensive. The sensor–cloud infrastructure is a cost-effective approach, where the existing cloud platform can be used.

A sensor–cloud architecture is conceptualised with a combination of sensor networks and cloud applications [5, 7]. Owing to the limitations in sensor network such as limited bandwidth, storage and memory, sensed data from the sensor nodes are processed through a cloud infrastructure in order to have reliable, cost-effective and real-time monitoring facilities. Cloud applications are able to collect and process huge data that are generated from sensor nodes through the gateways on the both sides – cloud gateways and sensor gateways. In such a publish/subscribe system, the sensor–cloud provides a platform for the execution of services that operate on sensed data and also satisfying and ensuring the derived trust and security requirements. A sensor–cloud enables the users to easily gather, access, process and search for a large number of data stored in the cloud infrastructure by using the computational and storage devices [8]. The dynamic behaviours of the sensor–cloud infrastructure facilitate automatic furnishing of its services as required by the users [9, 10]. In this paper, we design a framework for optimal gateway selection in an efficient, reliable and cost-effective manner in a sensor–cloud, that will help health monitoring applications.

The rest of the paper is organised as follows. In Section 2, we briefly present the literature review related to this work. Section 3 presents the system model for the sensor–cloud framework. We propose a solution for optimal gateway selection in Section 4. Some results are discussed in Section 5. Finally, we conclude the paper with a few research directions in Section 6.

2 Related work

Several works have been performed in the past years for health monitoring by using sensor–cloud infrastructure [9, 11–20]. In the existing health monitoring systems prior to sensor–cloud, patients' health data are collected by the workers (such as nurse, doctors) manually. This manual system leads to a cost-expensive, error-prone and increased latency, which are unsuitable for real-time monitoring.

To address these issues, Rolim *et al.* [9] discussed a solution for a cost-effective, real-time health monitoring system. In such a scenario, the health information is fed into a cloud environment, so that information is available from anywhere to the patients and also to the health-care centre. Thus, manual data collection issues have been addressed by the authors for real-time, always-on and automated health monitoring systems. However, in such a solution model, delay-optimised gateway selection mechanism is required for mapping sensed data with cloud applications.

In [11], a pub–sub-based model is proposed by the authors in order to integrate cloud applications with the sensor network. The authors represented publish and subscribe model as pub–sub model. According to the authors, users register their information on the *SaaS* application through the publish model. In the subscribe model, appropriate subscribers are found for each application by using an event matching algorithm. In such a model, published data are available through the existing web services, and are accessible only to the subscribed customers. They used a pub–sub broker to utilise the cloud infrastructure as *SaaS* to monitor, process and deliver the events to the users. However, this model is not autonomous. Periodic predictions are required to be calculated for the purpose of cost-optimisation.

In [14], the authors proposed a fast and flexible information dissemination system for automatic publish/subscribe mechanism. In such a system, a class-group index matching algorithm is proposed for minimising delay to receive subscribed content, and predictions are automatically calculated. However, performance degradation takes place when the number of customers is increased.

Aoki *et al.* [15] proposed a deep sensor-data aggregation mechanism for reducing congestion in network traffic for fast response real-time data collection. The authors focus on the issues related to cloud applications such as latency, limited bandwidth and long distance transmission for real-world implementations using wide area network (WAN) architecture. To address these issues, they divide the WAN architecture into two parts – 'front-side' and 'back-side'. The 'back-side' is the same as today's network architecture, and the 'front-side' is located between the WAN and the real-world part for handling real-world datastreams.

In [16], the authors propose a secure multicast sensor–cloud application based on the combination of group-key and time-key mechanisms instead of broadcast mechanism. The group-key is used for the group of users that satisfy the data arriving from the sensor–cloud. The time-key is used for optimisation of the key updating from joining or leaving of users. The authors show that by using multicast method, it is possible to reduce the computation and response time.

Mohapatra and Rekha [17] proposed a hybrid framework for monitoring patient health status by using a sensor–cloud. They demonstrated the benefits of using sensor–cloud architecture for patient health-status monitoring.

Biswas *et al.* [21–23] studied the sleep–wake cycle of the patients in a nursing home. The authors proposed a SAPM system for real-time health monitoring in a local nursing home. They also implemented a cost-effective circadian rhythm monitoring system and tested the mechanism on trial data collected from a nursing home. Biswas *et al.* [21] also claimed that their SAPM system works well for real-time health monitoring.

In [24], the authors proposed a cost-effective, versatile clinical tool for mobile voice monitoring to track the daily condition of the vocal system. They use a wearable sensor for data collection, and smart phone application platform for storing the real-time data. By using the smart phone application and the wearable sensors, the authors proposed that the treatment of hyper-functional vocal disorders can be effectively enhanced by providing real-time feedback to facilitate healthy vocal function.

It can be inferred from the existing works that most of the existing works addressed the issues related to the mapping sensor network with cloud applications for scalability and availability of the real-time monitoring system. However, still, an optimal gateway selection mechanism is needed to be addressed to establish an efficient, reliable and cost-effective monitoring system and to minimise the service delay in a dynamic sensor–cloud relationship.

3 System model

A sensor–cloud uses the publish/subscribe (pub/sub) model for receiving data from the sensor network, and for distributing that data to the interested cloud applications. All these intermediate operations are performed by the pub/sub broker that act as an intermediary entity between the sensor network and the cloud platform. We address the execution delay problem for optimal gateway selection in the sensor–cloud framework for the real-time monitoring systems.

In Fig. 1, a health monitoring system using sensor–cloud application is shown. In Fig. 1, G_1, G_2, G_3 and G_4 are the gateways between the sensor network and the cloud platform. P_1, P_2, P_3 and P_4 represents the patients in different areas. The sensed data from the sensor network are fed into the cloud environment with the help of the gateways. All the health-care centres, H_1, H_2, H_3 and H_4, can access the real-time health data cloud through the gateways, and according to that, take decisions.

Let N_G be the set of gateways and N_R be the set of requests from the users. Hence, our objective is to choose an optimal gateway $G_i \in N_\mathrm{G}$ for each $R_i \in N_\mathrm{R}$, where $i \in N$, (N is the set of natural numbers) such that all the requests, N_R, are mapped according to their priorities. After selecting a gateway, G_i, it is included into a candidate gateway set, C_i, for the request, R_i, and considered as the candidate gateway for optimal gateway selection strategy. There may be more than one candidate gateway for a request, R_i. In such a scenario, we also consider the relative bandwidth, that is, after selecting a gateway, G_i, for a request, R_i, the

doi: 10.1049/iet-wss.2013.0073

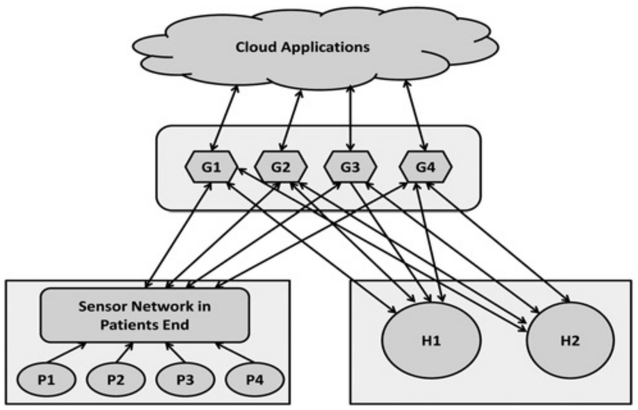


Fig. 1 *Schematic view of a health monitoring system using sensor–cloud applications*

bandwidth $B_{G(i)}$ of the gateway is reduced during the time interval for serving the request.

A gateway, $G_i \in N_G$, is selected for any request, R_i, as an optimal gateway. After serving the request, R_i, the gateway bandwidth, $B_{G(i)}$, is reduced by a constant value, say c_x. Hence, the new relative bandwidth, $B_{G(i)} - c_x$, is considered to serve another request, R_j. Our aim is to choose the optimal one from the set of candidate gateways, C_i, for transferring the user requested data to the sensor–cloud framework. For example, suppose we have two requests, R_1 and R_2, and two gateways, G_1 and G_2. Request R_1 can be served by either G_1 or G_2, and request R_2 can be served only by G_1. Now, let us consider that G_1 has higher bandwidth than G_2. In such a condition, R_1 is served first as it comes first and is served by the gateway, G_1, as it has higher bandwidth. After serving the first request, gateway G_1 does not have enough bandwidth to serve the request, R_2. Hence, in this circumstance, a proper gateway selection mechanism is needed to be addressed to serve all the requests optimally.

4 Solution approach

4.1 Game formulation

For transferring the user requested data to the sensor–cloud, we use a two-player zero-sum game [25–27] for optimal gateway selection for the reduction of delay. A game is said to be zero-sum, if for any outcome, the sum of the pay-offs to all players is zero, and can be expressed as

$$\sum_i U_i = 0, \quad i = 1, 2, \ldots, N \tag{1}$$

In such a zero-sum game model, the first player employs the strategy under the feedback information, whereas the second player utilises the strategy under the information structure of the first player. The problem is to serve the user requests by using the appropriate gateway in terms of bandwidth to reduce the delay for serving requests. In such a scenario, we consider the two-player game between the gateway and the user-request. In such a game model, the gateway acts as the leader and the user-request acts as the follower. Now, if the strategies of the two players are the same, the value of the pay-off functions for the players are identical. The first player, that is, the gateway chooses one strategy, and then the second player, that is, the user, chooses the strategies over the first player's strategy.

Definition 1: Let Z be a game defined by $(N, (A_i), (U_i))$, where N is the number of players, A_i is the set of pure strategies of player i and U_i is the corresponding pay-off for the player i. Then, Z is called a zero-sum game if [28]

$$\sum_{i=1}^{n} U_i(\boldsymbol{a}) = 0, \ \forall \boldsymbol{a} \in A \tag{2}$$

In our two-player zero-sum game model, $N = 2$. Hence, (2) may be written as

$$U_1(a_1, a_2) = -U_2(a_1, a_2), \ \forall a_1 \in A_1, a_2 \in A_2 \tag{3}$$

4.2 Nash equilibrium

In the proposed model, we show that there exists Nash equilibrium, and also it follows the properties below [28]:

1. All players exhibit the same value of pay-off at the Nash equilibrium points.
2. For a two-point Nash equilibrium, it is possible to replace the strategy of one player in the first point by the strategy of the same player at another point.

This can be formally written, as shown in Theorem 1.

Theorem 1: Let Z be the two-player zero-sum game defined by $((A_1, A_2), \pi)$. Let (τ_1, τ_2) and (σ_1, σ_2) be the two Nash equilibrium points for the game Z. Then,

1. $\pi(\tau_1 . \tau_2) = \pi(\tau_1, \sigma_2) = \pi(\sigma_1, \tau_2) = \pi(\sigma_1, \sigma_2)$
2. Both (σ_1, τ_2) and (τ_1, σ_2) are the Nash equilibria of Z.

Proof (Part I): (σ_1, σ_2) is a Nash equilibrium. Therefore, the first player, the gateway, plays for maximising the pay-off, and, thus, we have

$$\pi(\sigma_1, \sigma_2) \geq \pi(\tau_1, \sigma_2) \tag{4}$$

We have also another Nash equilibrium point as well, that is (τ_1, τ_2). For this Nash equilibrium point, the second player, user request, plays for minimising the pay-off, and, thus, we have

$$\pi(\tau_1, \sigma_2) \geq \pi(\tau_1, \tau_2) \tag{5}$$

After combining (4) and (5), we obtain

$$\pi(\sigma_1, \sigma_2) \geq \pi(\tau_1, \sigma_2) \geq \pi(\tau_1, \tau_2) \tag{6}$$

Similarly, we obtain the inequalities

$$\pi(\sigma_1, \sigma_2) \leq \pi(\tau_1, \sigma_2) \leq \pi(\tau_1, \tau_2) \tag{7}$$

Finally, from (6) and (7), we can prove that

$$\pi(\tau_1 . \tau_2) = \pi(\tau_1, \sigma_2) = \pi(\sigma_1, \tau_2) = \pi(\sigma_1, \sigma_2) \tag{8}$$

Proof (Part II): We observe that (σ_1, σ_2) is a Nash equilibrium point of the two-player game. From (8), we have

$$\pi(\alpha_1', \sigma_2) \le \pi(\sigma_1, \sigma_2) = \pi(\tau_1, \sigma_2), \quad \forall \alpha_1' \in A_1 \qquad (9)$$

and

$$\pi(\tau_1, \alpha_2') \ge \pi(\tau_1, \tau_2) = \pi(\tau_1, \sigma_2), \quad \forall \alpha_2' \in A_2 \qquad (10)$$

The combination of (9) and (10) implies that (τ_1, σ_2) is a Nash equilibrium as well.

In the same way, we can prove that (σ_1, τ_2) is also a Nash equilibrium.

In Table 1, we show the zero-sum game model, where the gateway and the request are the first and the second players, respectively. The gateway chooses the strategy based on its available bandwidth to maximise the pay-off to 1. With the chosen strategy of the gateway, the request follows it to obtain the service. Thus, the players, gateway and request, maximise their pay-off by choosing the (1, −1) strategy.

At the beginning of the game, we consider the pay-off for gateways and requests as zero. In our proposed approach, all the gateways try to maximise their pay-off from 0 to 1. At the same time, all the requests try to obtain service from the gateways as soon as possible in order to reduce the delay. Thus, we consider the pay-off for the serviced request as −1. After servicing all the requests, the sum of the pay-off of the requests and the gateways is zero, and thus, follows the zero-sum game approach, that is

$$\sum_{t=0}^{T} U_{\mathrm{G}} + \sum_{t=0}^{T} U_{\mathrm{R}} = 0 \qquad (11)$$

where U_{G} and U_{R} are the utilities of the gateway and the request, respectively.

1. *Utility of gateways*: We consider the gateway utility function as $U_{\mathrm{G}}(S, B_{G(i)}, \boldsymbol{M}, \alpha, \beta)$, where

Table 1 Zero-sum game model

	Gateway	
Request	(0, 0)	(1, −1)
	(−1, 1)	(0, 0)

(i) $B_{G(i)}$ is the gateway bandwidth value.
(ii) $\boldsymbol{M}$ is the $(N_{\mathrm{G}} \times N_{\mathrm{R}})$ matrix.
(iii) α and β are the decision variables of the two players.

Hence, for the first player (i.e. the gateway) the strategy is as follows

$$\begin{aligned}\max\left[U_{\mathrm{G}}(B_{G(i)(0)}, \boldsymbol{M}_0, \alpha, \beta)\right] &= \sum_{t=0}^{T-1} U_{G(t)}\left(B_{G(i)(t)}, \boldsymbol{M}_t, \alpha_t, \beta_t\right) \\ &\quad + U_{g(T)}\left(S_T, B_{G(i)(T)}, \boldsymbol{M}_T, \alpha_T, \beta_T\right)\end{aligned} \qquad (12)$$

1. *Utility of user-requests*: We consider the utility function of the second player (i.e. the user request) to be denoted as $U_{\mathrm{R}}(r_p, \boldsymbol{M}, \alpha, \beta)$, where

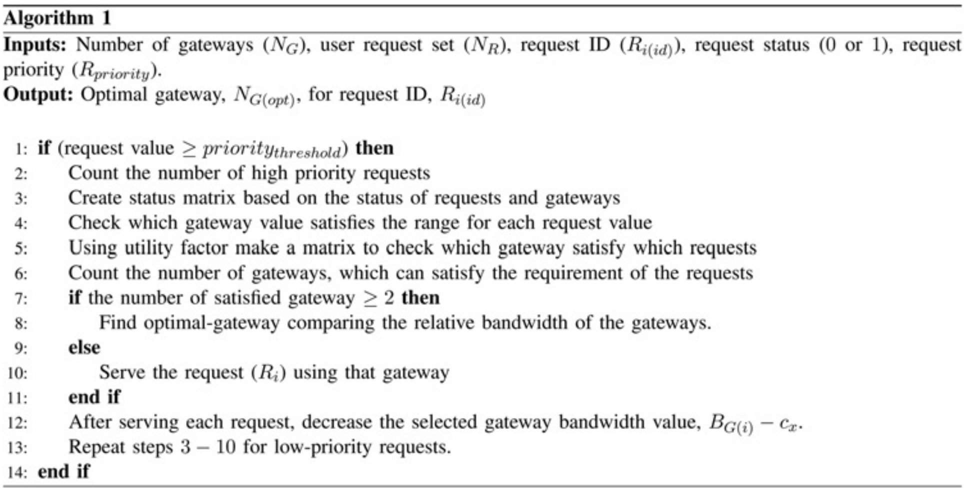
(i) r_p denotes the request priority.
(ii) $\boldsymbol{M}$ is the $N_G \times N_R$ matrix. In the matrix $\boldsymbol{M}$, the matrix value is 1, if the corresponding gateway and request status is unity.
(iii) α and β are the decision variables of the two players.

For the user request, the utility function is defined as follows

$$\begin{aligned}\max\left[U_{\mathrm{R}}\left(r_{p(0)}, \boldsymbol{M}_0, \alpha_0, \beta_0\right)\right] &= -\sum_{t=0}^{T-1} U_{R(t)}\left(r_{p(t)}, \boldsymbol{M}_t, \alpha_t, \beta_t\right) \\ &\quad + U_{M(T)}\left(r_{p(T)}, \boldsymbol{M}_T, \alpha_T, \beta_T\right)\end{aligned} \qquad (13)$$

4.3 Algorithm for optimal gateway selection

Our proposed algorithm (see Fig. 2) follows the 'first come first serve' approach with priority-based service. The algorithm finds out the high priority requests among the active ones, and serves those requests that come first. We consider that the proposed framework supports multiple gateways for each request. When the gateway receives user requested data to serve, the system first checks the gateway



```
Algorithm 1
Inputs: Number of gateways ($N_G$), user request set ($N_R$), request ID ($R_{i(id)}$), request status (0 or 1), request priority ($R_{priority}$).
Output: Optimal gateway, $N_{G(opt)}$, for request ID, $R_{i(id)}$

1: if (request value $\ge priority_{threshold}$) then
2:     Count the number of high priority requests
3:     Create status matrix based on the status of requests and gateways
4:     Check which gateway value satisfies the range for each request value
5:     Using utility factor make a matrix to check which gateway satisfy which requests
6:     Count the number of gateways, which can satisfy the requirement of the requests
7:     if the number of satisfied gateway $\ge 2$ then
8:         Find optimal-gateway comparing the relative bandwidth of the gateways.
9:     else
10:        Serve the request ($R_i$) using that gateway
11:    end if
12:    After serving each request, decrease the selected gateway bandwidth value, $B_{G(i)} - c_x$.
13:    Repeat steps $3 - 10$ for low-priority requests.
14: end if
```

Fig. 2 *Algorithm for optimal gateway selection in sensor–cloud* value to ascertain whether the gateway satisfies the request or not, instead of storing the data into cloud storage first. If the gateway does not satisfy the gateway value, then the data are stored in cloud storage to minimise the delay. Then, the optimal gateway is selected for serving the request.

5 Results and discussion

We used MATLAB to simulate the proposed solution. Table 2 shows the different parameters used for simulation. We evaluated the performance of the proposed solution in three ways: (i) utilisation of the gateway, (ii) delay for serving the requests and (iii) overall comparison of gateway utilisation and service delay.

We calculate the computational delay for each request, which are serviced by the gateways. We observe that the high priority requests are serviced first, according to the proposed framework. Among the first priority requests, they are serviced on a ‘first come first serve’ basis. We calculated the delay for each request ID, when ten gateways are available. Here, all the requests are serviced with optimal delay.

Table 2 Simulation parameters

Parameter	Value
maximum number of requests at a time	2000
maximum number of gateways	50
request priority	0 or 1
gateway bandwidth	≤40 unit
requested bandwidth	≤15 unit

In Fig. 3*a*, we show the number of requests served by the gateways, when 50 requests are served by 10 gateways. We observe that all the gateways are optimally utilised according to the number of requests. In Fig. 3*b*, we show the results of the service delay although 50 requests are served by 10 gateways. All the high priority requests are serviced first, and, thus, the delay is lower than the low priority requests.

In Fig. 4*a*, the utilisation of the gateways is shown for different gateway ID with the number of requests served, when we consider 100 requests and 10 gateways are available for serving the requests. We observe that all the gateways are optimally utilised to serve the requests, according to their available bandwidth. We also observed the changes in service delay although 10 gateways are available for 100 requests, as shown in Fig. 4*b*. We see that the delay increases compared with that in Fig. 3*b*.

Fig. 5*a* presents the utilisation of the gateways for 100 requests, when 25 gateways are considered to serve the requests. In this case as well, we observe that the gateways are utilised optimally with an increase in the number of gateways in comparison with Fig. 4*a*, according to the available bandwidth. In Fig. 5*a*, we see that since there are no remaining requests to be served, some of the gateways (such as 23, 24 and 25) are not used. In Fig. 5*b*, we show the waiting time for each request to be served.

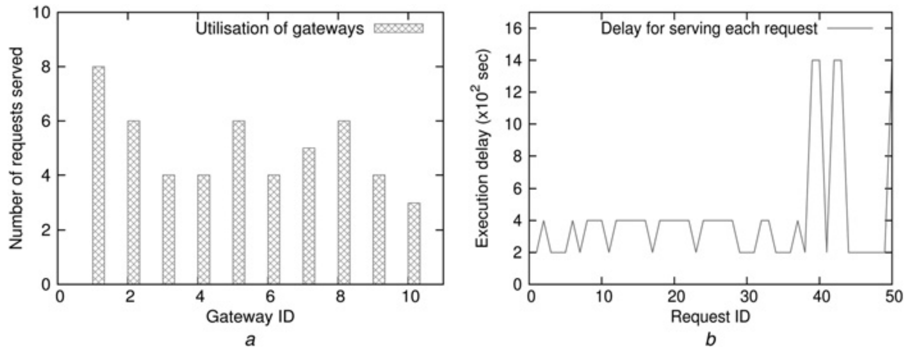


Fig. 3 *Fifty requests are served by 10 gateways*

a Utilisation of gateways
b Requests served with delay

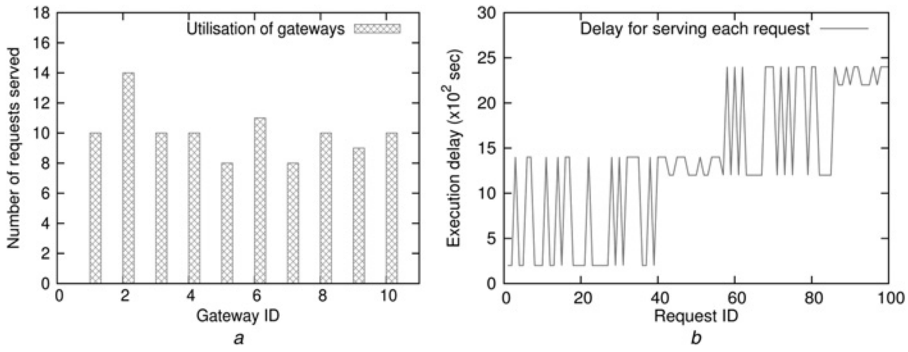


Fig. 4 *Hundred requests are served by 10 gateways*

a Utilisation of gateways
b Requests served with delay

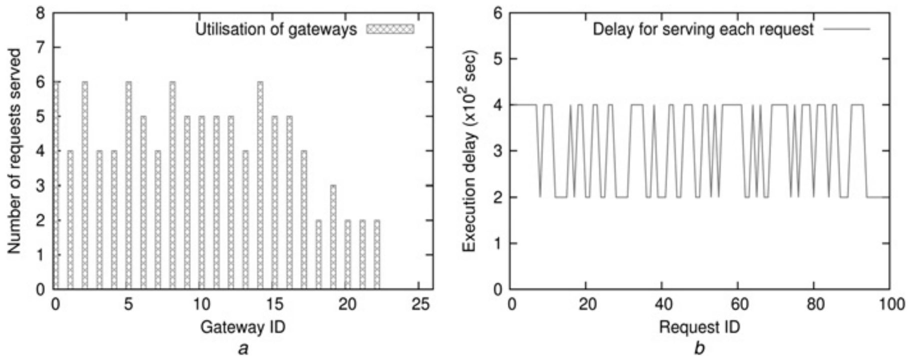


Fig. 5 *Hundred requests are served by 25 gateways*

a Utilisation of gateways
b Requests served with delay

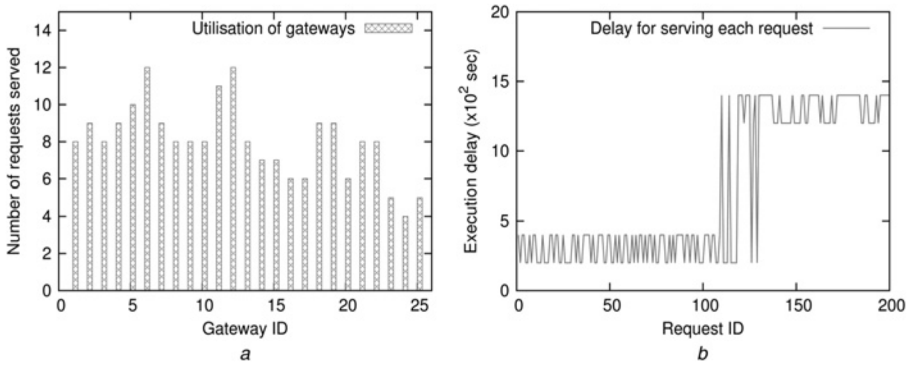


Fig. 6 *Two hundred requests are served by 25 gateways*

a Utilisation of gateways
b Requests served with delay

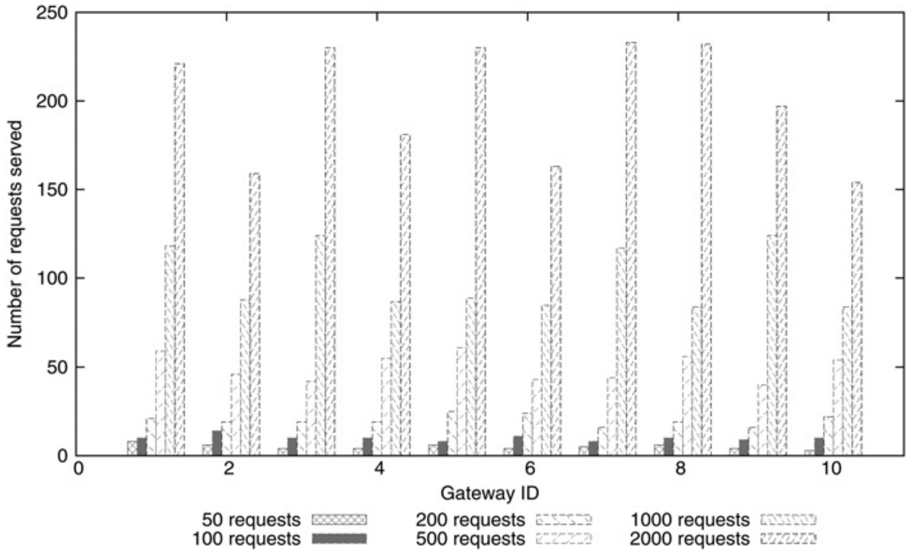


Fig. 7 *Comparison of gateway utilisation with the number of requests served*

Fig. 6*a* shows the utilisation of the gateways when 25 of these are considered for serving 200 requests. We observe that all the gateways are utilised for serving the requests unlike as seen in Fig. 5*a*. In Fig. 6*b*, with an increase in the number of requests and gateways, we calculated the delay for each request wait time to obtain service from the gateways. In Fig. 6*b*, we show the results, when 200 user requests are serviced by 25 gateways. We observed that with an increase in the number of gateways, delay is reduced compared with that in Fig. 4*b*, where 100 requests are serviced by 10 gateways.

doi: 10.1049/iet-wss.2013.0073

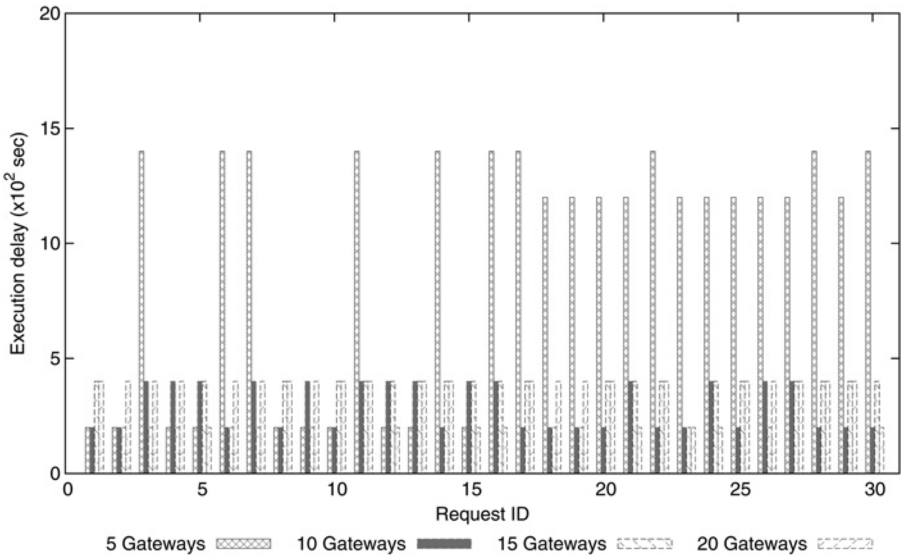


Fig. 8 *Comparison of delay with the number of gateways*

Overall comparison: Finally, we compared the number of requests served with fixed number of gateways in Fig. 7, and the delay for each request to obtain the service in Fig. 8.

We compared the number of requests served by each gateway in our proposed algorithm, although the total number of gateways is fixed (see Fig. 7). We observe that each individual gateway serves more number of requests with an increase in the total number of requests. Each gateway serves requests optimally to reduce the delay.

In Fig. 8, we show the results of the comparison when each request is serviced by different number of gateways having different waiting time. Here, we can also observe that with an increase in the gateway number, the waiting time is reduced optimally, as shown in Fig. 8.

6 Conclusion and future work

In this paper, we proposed a solution to the optimal gateway selection problem in the sensor–cloud. We modelled the problem as a delay optimisation problem in the sensor–cloud architecture. Based on the gateway selection problem, we have shown how the user requests can be mapped to the optimal gateway and serviced through the sensor–cloud environment. We observed that our proposed framework works well for delay optimisation. We considered request priorities for gateway selection, that is, all the high priority requests are serviced first. The future extension of this work includes how the requests can be serviced by the gateways more effectively in order to have a reliable, and efficient, real-time monitoring system.

7 References

1 Irwin, D., Shenoy, P., Cecchet, E., Zink, M.: ‘Resource management in data-intensive clouds: opportunities and challenges’. Proc. IEEE Workshop on Local and Metropolitan Area Networks (LANMAN), Long Branch, NJ, May 2010, pp. 1–6

2 Alamri, A., Ansari, W.S., Hassan, M.M., Hossain, M.S., Alelaiwi, A., Hossain, M.A.: ‘A survey on sensor-cloud: architecture, applications, and approaches’, *Int. J. Distrib. Sensor Netw.*, 2013, **2013**, pp. 1–18

3 Misra, S., Tiwari, V., Obaidat, M.S.: ‘LACAS: Learning automata-based congestion avoidance scheme for healthcare wireless sensor networks’, *IEEE J. Sel. Areas Commun.*, 2009, **27**, (4), pp. 466–479

4 Krishna, P.V., Misra, S., Joshi, D., Obaidat, M.S.: ‘Learning automata based sentiment analysis for recommeder system on cloud’. Proc. 2013 Int. Conf. Computer, Information and Telecommunication Systems (CITS), Piraeus-Athens, Greece, May 2013

5 Yuriyama, M., Kushida, T.: ‘Sensor-cloud infrastructure -physical sensor management with virtualized sensors on cloud computing’. Proc. IEEE Int. Conf. Network-Based Information Systems (NBiS), Takayama, September 2010, pp. 1–8

6 Varshney, U.: ‘Pervasive healthcare and wireless health monitoring’, *Mobile Netw. Appl.*, 2007, **12**, (2–3), pp. 113–127

7 Dash, S.K., Sahoo, J.P., Mohapatra, S., Pati, S.P.: ‘Sensor-cloud: assimilation of wireless sensor network and the cloud’, *Adv. Comput. Sci. Inf. Technol. Netw. Commun.*, 2012, **84**, pp. 455–464

8 Lan, K.T.: ‘Whats next? Sensor + cloud?’. Proc. ACM Int. Workshop on Data Management for Sensor Networks, 2010, pp. 978–971

9 Rolim, C.O., Koch, F.L., Westphall, C.B., Werner, J., Fracalossi, A., Salvador, G.S.: ‘A cloud computing solution for patients data collection in health care institutions’. Proc. IEEE Int. Conf. eHealth, Telemedicine, and Social Medicine, ETELEMED, 2010, pp. 95–99

10 Mondal, A., Misra, S.: ‘Dynamic coalition formation in a smart grid: a game theoretic approach’. Proc. Third IEEE Int. Workshop on Smart Communication Protocols and Algorithms (SCPA), Colocated with IEEE ICC 2013, Budapest, Hungary, June 2013, pp. 1–5

11 Hassan, M.M., Song, B., Huh, E.-N.: ‘A framework of sensor-cloud integration opportunities and challenges’. Proc. Int. Conf. Ubiquitous Information Management and Communication (ICUIMC), 2009, pp. 618–626

12 Yen, Y.-S., Chang, H.-C., Chang, R.-S.: ‘Routing with adaptive path and limited flooding for mobile ad hoc networks’. Proc. Int. Symp. Wireless Pervasive Computing, January 2006

13 Chen, W.-M., Li, C.-S., Chiang, F.-Y., Chao, H.-C.: ‘Jumping ant routing algorithm for sensor networks’, *Comput. Commun.*, 2007, **30**, (14–15), pp. 2892–2903

14 Hassan, M.M., Song, B., Huh, E.-N.: ‘A dynamic and fast event matching algorithm for a content-based publish/subscribe information dissemination system in sensor-Grid’, *J. Supercomput.*, 2010, **54**, (3), pp. 330–365

15 Aoki, H., Shikano, H., Okuno, M., *et al.*: ‘Cloud architecture for tight interaction with the real world and deep sensor-data aggregation mechanism’. Proc. IEEE Int. Conf. Software, Telecommunication and Computer Networks (SoftCOM), 2010, pp. 280–284

16 Nguyen, T.-D., Huh, E.-N.: ‘An efficient key management for secure multicast in sensor-cloud’. Proc. Int. Conf. Computers, Networks, Systems, and Industrial Engineering (ACIS/JNU), 2011, pp. 1–9

17 Mohapatra, S., Rekha, K.S.: ‘Sensor-cloud: a hybrid framework for remote patient monitoring’, *Int. J. Comput. Appl.*, 2012, **55**, (2), pp. 1–11

18 Lai, C.-F., Huang, Y.-M., Park, J.H., Chao, H.-C.: ‘Adaptive body posture analysis for elderly-falling detection with multisensors’, *IEEE Intell. Syst.*, 2010, **25**, (2), pp. 20–30

19 Lai, C.-F., Chang, S.-Y., Chao, H.-C., Huang, Y.-M.: ‘Detection of cognitive injured body region using multiple triaxial accelerometers for elderly falling’, *IEEE Sensors J.*, 2011, **11**, (3), pp. 763–770

20 Yen, Y.-S., Chao, H.-C., Chang, R.-S., Vasilakos, A.: 'Flooding-limited and multi-constrained QoS multicast routing based on the genetic algorithm for MANETs', *Int. J. Math. Comput. Model.*, 2011, **53**, (11–12), pp. 2238–2250

21 Biswas, J., Maniyeri, J., Gopalakrishnan, K., *et al.*: 'Processing of wearable sensor data on the cloud – a step towards scaling of continuous monitoring of health and well-being'. Proc. IEEE Annual Int. Conf. Engineering in Medicine and Biology Society (EMBC), Buenos Aires, August–September 2010, pp. 3860–3863

22 Biswas, J., Maniyeri, J., Louis, S., Yap, P.: 'Fast matching of sensor data with manual observations'. Proc. IEEE Annual Int. Conf. Engineering in Medicine and Biology Society, (EMBC), Minneapolis, MN, September 2009, pp. 1675–1678

23 Biswas, J., Jayachandran, M., Shue, L., Gopalakrishnan, K., Yap, P.: 'Design and trial deployment of a practical sleep activity pattern monitoring system'. Proc. Int. Conf. Smart Homes and Health Telematics, (ICOST), Tours, France, June–July 2009

24 Mehta, D., Zaartu, M., Feng, S., Cheyne, H., Hillman, R.: 'Mobile voice health monitoring using a wearable accelerometer sensor and a smartphone platform', *IEEE Trans. Biomed. Eng.*, 2012, **59**, (11), pp. 3090–3096

25 Zhang, X.-Y.: 'A note on dynamic zero-sum games'. Proc. Chinese Conf. Control and Decision (CCDC), 2009, pp. 4758–4760

26 Fang, Z., Liu, Y., Liu, S., Chen, H.: 'Functional model for zero-sum game and its matrix solution concerning uncertain information'. Proc. IEEE Int. Conf. Systems, Man, and Cybernetics (SMC), Seoul, October 2012, pp. 17–23

27 Akers, S.B.: 'Threshold logic and two-person, zero-sum games'. Proc. Second Annual Symp. Switching Circuit Theory and Logical Design (SWCT), Detroit, MI, USA, October 1961, pp. 27–33

28 Owen, G.: 'Game theory' (Academic Press, New York, 1982)

doi: 10.1049/iet-wss.2013.0073