2014 IEEE International Conference on Internet of Things (iThings 2014), Green Computing and Communications (GreenCom 2014), and Cyber-Physical-Social Computing (CPSCom 2014)

An Efficient Data Storage Method of NoSQL Database for HEM Mobile Applications in IoT

Wei-Chen Chen, Chao-Lin Wu
Intel-NTU Connected Context Computing Center
National Taiwan University
Taipei, Taiwan, R.O.C.
zarcen@acm.org, clwu@ieee.org

Ya-Hung Chen, Li-Chen Fu
Dept. Computer Science & Information Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
nizi1127@gmail.com, lichen@ntu.edu.tw

***Abstract*—In this work, an efficient data storage method of NoSQL database is proposed for mobile devices and Internet applications to monitor appliance status in a home environment in real-time, and a home energy management (HEM) mobile App for monitoring this information is also developed. An HBase cluster is constructed as a testbed for evaluation. The evaluation is conducted based on the developed APP accessing data stored by the proposed method, and it shows promising efficiency. This App together with the proposed method can help users better aware of the details of their home energy usage, and can even help other remote software agents/applications to deliver further useful applications, e.g. large scale monitoring on demand response.**

Keywords—smart phones; IoT applications; NoSQL database; HBase;

I. INTRODUCTION

As Internet of Things (IoT) grows and smart phones get more and more popular, there are increasing remote applications deployed on mobile devices or built as web services on Internet. One of the most popular applications is to interact with appliances in a home environment. And in order to support this kind of applications, there are several existing solutions as implementation of [1] and [2]. Provided that all the appliances are equipped with the capability of network communication like Ethernet, a network among appliances in a home environment could be constructed and provide direct access for smart mobile devices or applications to control or monitor the status of appliances. This sort of appliances is called "Smart Appliance" such as Smart TV or Smart Laundry Machine. In the latest decade, several international companies have begun to devote their effort to develop Smart Appliances. With the help of the industrial support mentioned above, Smart Appliances could automatically communicate with other objects on the Internet without additional gateway, and thus join and become a part of IoT. Nevertheless, currently it is still not feasible to assume that all the home appliances have the sufficient capability of networking and computing as Smart Appliances. Therefore, in order to support those conventional appliances with constrained capability, a home gateway is a common method to construct the network communication for them to join IoT and interact with other objects, e.g. smart mobile devices or web services.

With the network communication support for appliances, the most common functionality of a smart application for a home environment is to gather data about the home environment (e.g. appliance status mentioned above) and to provide appropriate services based on the gathered data. Furthermore, some applications even require these data to be recorded in a long term way thus to do further analysis. For instance, the development of home energy management (HEM) also begins to step toward smart mobile devices [3], [4]. Thus, no matter how to interact with the appliances in a home environment, it is important to not only construct the network connection but also to provide an efficient method to store and access data generated by the home appliances. In this paper, we propose an efficient method to store the information in NoSQL database which greatly suits for accommodating huge amount of log data that have no complicated relation and structure. NoSQL database is generally applied in many cloud platforms or services since its characteristics fit well in distributed systems and web environment and it has great scalability [5].

In Brewer's theorem [6], a distributed database impossibly reaches the following three conditions in the same time: consistency, availability, partition tolerance. In Burtica's work [7], they analyze different features on several NoSQL database. Cai et al. [8] further evaluate the performance of NoSQL database through several testing scenarios. It is difficult to point out which NoSQL database is the best one due to their various features. Among these NoSQL database, MongoDB is quite popular because it has abundant support from many web frameworks and its Json-style semantics is easy to use and understand. Except for MongoDB, HBase and Cassandra are two most popular NoSQL database. HBase has a better performance on consistency and partitions tolerance (CP); Cassandra chooses to focus on availability and partitions tolerance (AP) otherwise. Because IoT applications in a home environment usually generate numerous data frequently, the consistency becomes vital that whenever accessing the database it should always return the latest written data. As a result, we choose HBase to realize our design due to its principal of pursuing excellent consistency.

978-1-4799-5967-9/14 $31.00 © 2014 IEEE
DOI 10.1109/iThings.2014.62

With the support of data access mentioned above, we have also implemented an application on smart phones to verify the accessibility and functionality of the proposed method. The smart phone application is designed for three purposes: 1) remote control of the appliances in a home environment, 2) monitoring of the accumulated electricity bill, and 3) prediction of monthly electricity charge. These three purposes need to efficiently access the recorded data in a home environment thus to work appropriately.

II. DATABASE DATA

In Yamamoto's work [9], a cloud database, named Scallop4SC, which was implemented as a prototype on 12 Linux servers combines MySQL (SQL) and HBase (NoSQL) together to deal with different type of data in smart city's application. The configuration data that depict the hierarchy relation are stored in MySQL such that it can support various queries efficiently. In our application, however, we focus on simply home-scale application. Learning from the design of Scallop4SC, we adopted HBase to store the log data in a home environment. However, in order to achieve better suitability to collaborate with applications on smart mobile devices, we have proposed a further customization by using a hash function to reverse the timestamp of each inserted data record thus to ensure a better performance when querying the newest data in HBase. The proposed method is detailed as follows.

Using distributed database systems like HBase sometimes could cause inefficient queries. The reason is that HBase stores data in a way that those data with sequential rowkey value are concentrated together. This characteristic makes the data records mostly related to timestamp gather in a single server easily, resulting in a limited query performance because the stored data are not evenly distributed. For more evenly distributing these data records, it is reasonable to hash the rowkey and use a two-layer table structure. With the hashed rowkey, the records of log data could be prevented from concentrating in a single server. In addition to rowkey hashing, Yamamoto's method also uses a permutation of four search premixes to further duplicate the hash value pointer 24 times thus to enhance the query latency in various demands. In our application, we only need three search premixes including timestamp, houseID, and applianceID. These three premixes are permuted to generate six replicas of the hash value for a rowkey. But to make sure the latest data record, which will own a largest timestamp value accordingly, appears first when scanning the table, the final three premixes we adopted to permute are [<*LongMax*-timestamp>, houseID, applianceID], and the hash function we adopted is MD5 function. *LongMax* stands for the maximum value of long data type in HBase. The data type of timestamp is long; thus, <*LongMax-timestamp*> would also be a unique long value in HBase. Since the latest data will have a smallest value of <*LongMax-timestamp*>, it could be obtained first whenever the scan operation is performed on HBase. It is highly frequent to invoke the query of the latest data particularly for the status monitoring function of smart phone Apps. Therefore, it is beneficial to reverse the stored order of timestamp for each inserted data record thus to achieve efficient data query. TABLE I. and TABLE II. show parts of the final contents in our HBase database.

TABLE I. CONTENT OF TABLE 'HOUSEENERGYINDEX'

Row	Column+Cell
<LongMax-timestamp>.houseID.applianceID	hlkey
9223370638792244807.ac_new_bedroom.bl313	6714c3c1b6b85748 27c92611ff79b2d4
9223370638792244807.ac_new_openspace.bl313	9a2e46c2beeee0c7 1e074ef215e61c73
9223370638792244807.bl313.ac_new_bedroom	6714c3c1b6b85748 27c92611ff79b2d4
9223370638792244807.bl313.ac_new_openspace	9a2e46c2beeee0c7 1e074ef215e61c73
9223370638792249807.ac_new_bedroom.bl313	989c99d58fa6b302 bbdb1a1be48d47f7
9223370638792249807.ac_new_openspace.bl313	305b8114158b753a 27bf0e1e8efb5846
9223370638792249807.bl313.ac_new_bedroom	989c99d58fa6b302 bbdb1a1be48d47f7
...	...

TABLE II. CONTENT OF TABLE 'HOUSEENERGYLOG'

Row	Column+Cell	
MD5 hash value	acc_energy	power
0004702f55f4ae6e0811e91457794273	4.38	33.44
0004759b0dea63fb07a1470e51de2f15	0.02	0.0
00047bed4bd2a4ba0c29274f69133844	0.39	33.0
00047e1ed48d3ff66e9e3c1da9a2aa60	1.04	29.7
...	...	...

III. SMART PHONE APP

The implemented smart phone App has the following three features: Reminder, Cost Viewer, and Cost Analyzer.

A. Reminder

It visualizes energy usage in a home environment and provides inhabitants the capability to control appliances remotely. The snapshot of its view is shown in Fig 1. Wherever the user is, s/he can always check the Reminder view to see which appliance is not turned off. Like Fig 1(a), it indicates that there are three appliances not turned off where yellow blocks stand for on-status of appliances and white ones for off-status. After a confirmation, the user could click on the screen to turn off appliances like Fig 1(b) where the Television has been turned off. The Reminder page shows the status of each appliance in the order of how much money would be charged because we figured out this kind of visualization is quite helpful for users concerning their electricity bills after testing on several users. Literally, the function aims to remind users which appliances they might forget to turn off. Thus, we use a brighter color to emphasize the appliances that are turned on so that users could notice them quickly.

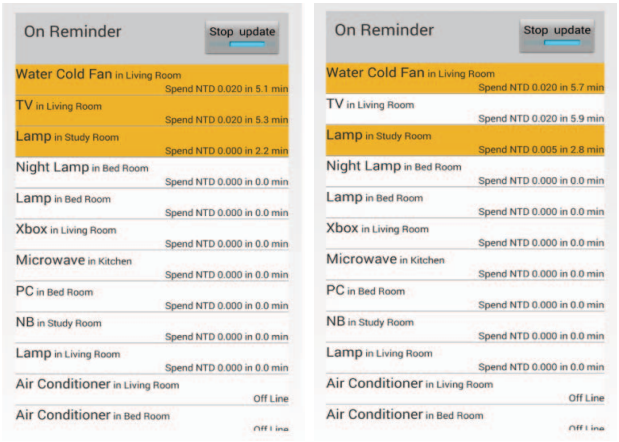
B. Cost Viewer

It illustrates a pie chart to show the energy cost of each appliance in a selected time period in the past. In Fig 2(a),

it gives an example that shows the energy cost in the past one month. To simplify the selection, it provides basically three options: 1) the past one month, 2) the past one week, and 3) the past three days. Whenever users invoke this view to query a specific period of energy usage information, the mobile client connects to the backend database to obtain the necessary data for visualization.

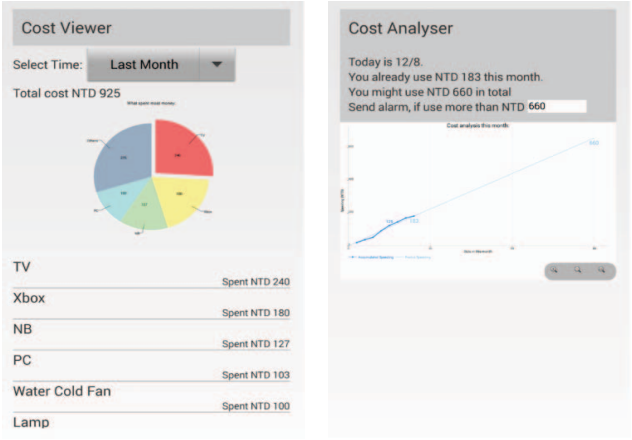
C. Cost Analyzer

It aims to make prediction of how much money the users should pay for the electricity bills in the end of month. It plots a curve diagram to demonstrate the inclination of energy consumption in this house and gives a rough prediction based on the current consumption pattern. The user could set an alarm so that the App could notify him/her whenever the potential electricity bills would exceed the original goal. In the example shown in Fig 2(b), it predicts that the bill will charge NTD 660 in the end of December. The prediction varies as day passed since the pattern of energy usage could vary as well. In fact, its main goal is not to generate an accurate prediction but to influence the users' habits through the visualized view in pursuit of making user self-disciplined.



(a) (b)

Fig 1. The view of Reminder



(a) (b)

Fig 2. The view of electricity visulization

IV. EXPERIMENTAL EVALUATION

We use 6 computers to build an HBase cluster as our database testbed. Its configuration is shown in TABLE III. The conducted experiment is described as follows: We create a number of client threads each of which executes the query of returning the latest information of a certain appliance to simulate that there are simultaneously a corresponding number of web services or our designed mobile Apps viewing the Reminder page. The simulation program firstly creates the specific number of threads and holds them until all threads are born, and then makes them query the database at the same time thereafter. Assumed that the time consumed on the context switch in the OS can be ignored, the above experimental setup can evaluate the response time of the database testbed under our proposed data storage method when being queried by numerous mobile clients almost simultaneously.

Each query constructs a connection to zookeepers and HBase regionservers remotely. We have three zookeepers to coordinate the load balancing as shown in TABLE III. Still, there must be a variance of network connection time once the visiting of database lasts long enough. To alleviate the influence of the variance of network latency, we measured the network latency time for each query at each round and discard the results from those queries which possess far higher latency than others as shown in Fig 4, thus to ensure the experimental results are reasonable. In our experiment, we only retain the results from those clients whose network latency are less than 1.3 seconds as illustrated in Fig 4.

There are two parameters in our experiment. One is the number of mobile clients to be created; the other is the number of repetition times. The number of repetition times means the number of rounds the simulation should be executed. A round refers that all generated clients have finished their queries and reported their response time. The number of repetition times is set to 20 in our experiments, and its corresponding experimental results for different number of clients are shown in Fig 3. From the results, we could see that nearly 95% of queries could finish in 0.5 seconds. Only when the number of clients exceeds 2000, about 5% of clients in each round will spend a relatively long time for a query.

TABLE III. CONFIGURATION OF THE DATABASE TESTBED

Node Name	Node Type	CPU	Mem.
robotlab-sh4	Zookeeper + HRegionServer	Intel Core i7 2.60GHz	16G
robotlab-sh5	Zookeeper + HRegionServer	Intel Core i7 2.60GHz	16G
robotlab-sh6	Zookeeper + HRegionServer	Intel Core i7 2.60GHz	16G
robotlab-sh7	HRegionServer	Intel Core i7 2.60GHz	16G
robotlab-sh8	HRegionServer	Intel Core i7 2.60GHz	16G
robotlab-sh9	HRegionServer	Intel Core i7 2.60GHz	16G

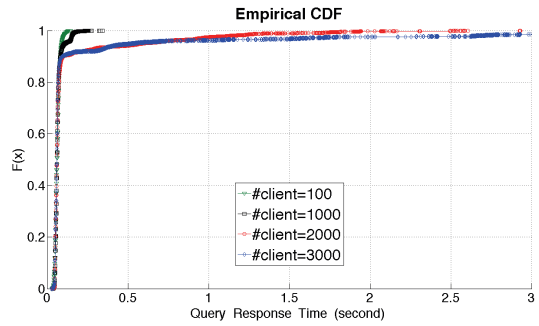


Fig 3. Empirical CDF of Query Response Time

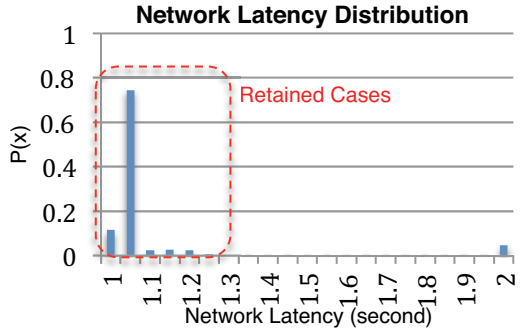


Fig 4. The distribution of the network latancy during our experiments and rounds with high network latency are discarded

V. CONCLUSION

This paper proposes an efficient data storage method for NoSQL database with a well-designed rowkey system which can support numerous HEM queries from IoT at the same time. We also implemented a smart phone App that exploits the edge of the proposed method, and with the help from the designed database system, it could serve user's needs agilely. To evaluate the proposed method, a testbed built by HBase database is implemented to accommodate the needs for HEM and an experiment is conducted to obtain the required time for a query from an IoT client, either a smart mobile device or a web application. From the evaluated result, over 95% of clients could obtain their information in 0.5 seconds while other hundreds or thousands of applications running at the same time. It shows the proposed method for storing household data in HBase can nicely serve the needs of at least hundreds of mobile applications simultaneously.

REFERENCES

[1] J. Nichols and B. A. Myers, "Controlling Home and Office Appliances with Smart Phones," *Pervasive Computing, IEEE*, vol. 5, no. 3, pp. 60–67, Jul. 2006.

[2] G. M. S. M. Rana, A. A. M. Khan, M. N. Hoque, and A. F. Mitul, "Design and implementation of a GSM based remote home security and appliance control system," presented at the Advances in Electrical Engineering (ICAEE), 2013 International Conference on, 2013, pp. 291–295.

[3] J. LaMarche, K. Cheney, S. Christian, and K. Roth, "Home energy management products & trends," *Fraunhofer Center for Sustainable Energy Systems. Cambridge, Massachusetts*, 2011.

[4] S. Aman, Y. Simmhan, and V. K. Prasanna, "Energy management systems: state of the art and emerging trends," *Communications Magazine, IEEE*, vol. 51, no. 1, pp. 114–119, 2013.

[5] J. Pokorny, "NoSQL databases: a step to database scalability in web environment," presented at the iiWAS '11: Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, New York, New York, USA, 2011, p. 278.

[6] E. Brewer, "CAP twelve years later: How the' rules' have changed," *Computer*, vol. 45, no. 2, pp. 23–29, 2012.

[7] R. Burtica, E. M. Mocanu, M. I. Andreica, and N. Tapus, "Practical application and evaluation of no-SQL databases in Cloud Computing," presented at the Systems Conference (SysCon), 2012 IEEE International, 2012, pp. 1–6.

[8] L. Cai, S. Huang, L. Chen, and Y. Zheng, "Performance analysis and testing of HBase based on its architecture," presented at the Computer and Information Science (ICIS), 2013 IEEE/ACIS 12th International Conference on, 2013, pp. 353–358.

[9] S. Yamamoto, S. Matsumoto, and M. Nakamura, "Using cloud technologies for large-scale house data in smart city," presented at the Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, 2012, pp. 141–148.