IProIoT: an In-network Processing Framework for IoT using Information Centric Networking

Qian Wang, Brian Lee, Niall Murray, Yuansong Qiao
Software Research Institute
Athlone Institute of Technology
Athlone, Co. Westmeath, Ireland
{qwang|ysqiao|nmurray}@research.ait.ie, blee@ait.ie

***Abstract* — The Internet of Things (IoT) network supports various network applications through billions of heterogeneous connected devices. Efficient processing of enormous amounts of IoT data collected and exchanged by these devices is a key problem. Another issue is addressing information-centric IoT application data aggregation. To achieve these goals, this paper utilizes the novel Information Centric Networking (ICN) technology to define a framework for IoT in-network processing (IProIoT). The philosophy of ICN offers a perfect match with the feature of IoT applications. It regards data as the first-class entity so that all operations on data is based on its unique name rather than the address of data storage. By leveraging a flexible naming scheme, our framework implements a set of computational components to execute and optimize IoT tasks. Thus, data is processed by appropriate IoT nodes on the path to consumers. Preliminary testing has shown the feasibility of our design in reducing network traffic and round-trip delay.**

Keywords—Internet of Things (IoT); Information Centric Networking (ICN); In-network processing

I. INTRODUCTION

The number of devices linked to the Internet of Things (IoT) will reach 50 billion by the year of 2020 [1]. These devices capture and transfer massive quantities of raw data for various IoT applications. This data then needs to be processed into meaningful information according to different requirements. However, many IoT devices are resource-limited and are unable to execute complex processing locally. Processing on cloud servers [8] is a popular solution for big data in terms of high speed and computing power. Cloud servers require all sensed data to be deliverd to the cloud for processing, with the result then returned to the actuators within IoT network. Consequently, this approach scarifices interaction between IoT devices. It also contributes to potential network congestion. To remedy this situation, T-Res [2] utilizes Constrained Application Protocol (CoAP) to execute monitoring and actuatoring tasks within the sensor network through direct communication between nodes. It sheds light on enabling more processing work within the IoT network.

Considering the amount of data that is and will be produced by IoT devices, a trade-off approach should be explored between IoT in-network and out-of-network (e.g. cloud) data processing. For example, complex costly processing could be performed outside the IoT network (saving resources within the IoT network), whilst less complex processing (e.g. data filtering at the source nodes and data aggregation on the fly) could be performed within the IoT network (with the benefit of reduced network traffic and processing delay). In order to fulfil in-network processing, the challenge is to evaluate potential devices capabilities and coordinate multiple sub-tasks due to the heterogeneity and constraint-resource of IoT devices.

The above mentioned IoT approaches (out/in network processing) are built upon today's IP network which has inherent limitations to support IoT applications. On one hand, IoT applications largely tend to be information-oriented. Because consumers desire more to use meaningful services instead of building connections with multiple devices to get raw data. On the other hand, it's also common for IoT applications to request data to be processed at intermediate nodes (e.g. IoT gateways). IP does not support these requirements well as it is host-centric. Thus, researchers are moving towards Information Centric Networking (ICN) [3] as the architecture to improve performance in many IoT domains, such as: V2X network [6] and smart home [14].

ICN focuses on information rather than location. This feature fits well with the requirements of IoT applications. It also provides in-network content caching along the routing path, which helps to reduce network delays. Potentially, IoT can take advantage of this feature of ICN to enable and improve in-network data transmission. However, as ICN is originally designed to improve large scale content distribution (one-to-many) such as YouTube service, it considers less for many-to-one communication paradigm. For example, multiple sensors collect and return data for one sink node in IoT scenarios [7]. Another missing design of ICN is to process information within network, which is the main demand for IoT. The challenge is to reshape ICN as the proper underlying backbone for IoT network to deliver and process data.

This paper is motivated to define a generic processing framework (IProIoT) for IoT by solving the above mentioned challenges. Our work is built on Named Data Networking (NDN) [4] which is one of the active ICN platform and mainly focuses on how to deploy and optimize data processing within IoT network. The main contributions of this paper are:

- Design a naming scheme to cache data and processing logic within IoT network in order to save traffic load and support in-network computation tasks.
- Build a framework consisting of capable IoT devices to cooperate together for all computational services.
- Define a working scheme to automatically schedule task execution according to specific requests and capability of devices.
- Set up a real-world test bed to verify our design.

The rest of the paper is organized as following: Section II discusses the related work of IoT and ICN. The framework and functional design are described in Section III. Section IV shows experimental evaluation and result analysis. Section V concludes the whole paper.

II. RELATED WORK

The proliferation of smart devices in communicating and actuating accelerates the growth of IoT network. However, these devices in many cases, are limited in terms of resources and as such, have difficulty in storing large dataset and handling complicated tasks. To remedy this situation, researchers have moved data out of the IoT network to Cloud computing environments for flexible configuration and easy upgrade of processing logics [8][12]. Without any doubt, the Cloud is attractive due to its powerful capability and sufficient resource. Cloud providers, eg: Microsoft [10] and Amazon [11], offer common platforms with high performance in terms of data storage, management and processing. As shown in Fig. 1, various IoT applications are able to obtain customized service thanks to the Cloud, which manipulates underlying sensing platform. As we discussed before, this approach multiples network traffic as IoT data will be sent in and out of network for different purpose. Current IoT data processing relies much on Cloud so that less efforts are made to explore the potential capacity within IoT network.

Massive amounts of IoT devices nearly consist of everything around us. For example: temperature sensors in smart homes, vehicles in smart transportation and mobile phones in urban sensing. Current Internet works well when the connection between two static machines are stable. However, the overhead costs associated with mobility scenarios are very demanding for IoT devices with limited resources. ICN defines a new communication paradigm centred on data itself, which holds great support for IoT ecosystems. For example: ICN for automotive network [15] improves data delivery in poor-quality link; data retrieval is allowed from multiple-source nodes by sending one request [7]. Most existing ICN for IoT solutions cope with data caching and transmission. This paper aims to extend ICN to enable process information, with special focus on IoT scenarios.

Named Function Network (NFN) [5] proposes to name functions as a special type of data using Lambda expression. However, NFN doesn't mention how the locality-of-execution is discovered and decided by the network, which is one of the contributions of this paper. The authors' initial work can be found in [13] [17].

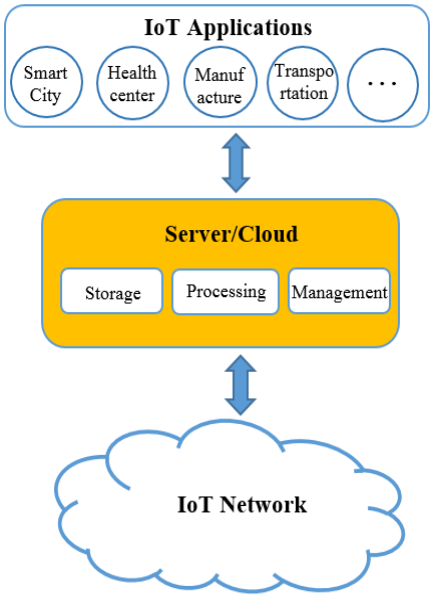


Fig. 1. Cloud Computing for IoT

III. FRAMEWORK AND FUNCTIONAL DESIGN

To exploit the IoT network inherent processing ability but also to support the information centric nature of IoT applications, the authors propose IProIoT: an in-network processing IoT framework based on ICN. IProIoT provides a generic computation model for different types of data processing. It also provides self-management protocols to optimize network performance.

A. Architecture Overview

The technical innovation of this paper is to propose IProIoT as a data process framework spanning over the underlying infrastructure (including IoT devices and edge/centralized could). Without any doubt, it requires coordination between clouds and IoT devices to complete tasks, such as: task divide, execution progress and so on. Our research focus on enabling computation on IoT devices as the first step.

Considering information-centric nature of IoT applications, IProIoT builds upon Named Data Networking (NDN). Our framework takes full use of NDN original functionality to forward, retrieve and cache data by exchanging two name-based packets. Requests are described as Interest and replies are called Data.

The heterogeneous nature of IoT network devices means the performance capability of devices will vary significantly. As a result, it is necessary to assign tasks according to the device's capabilities. A node that is able to process data is defined as one of the Computation ExecutOrs (CEO) of IoT in our design. Because it is not practical for users that are outside of the IoT network to have details of the ability of numerous devices. A Computation Manager (CM) within IoT is required to bridge users' requests and appropriate node(s). Fig.2 shows the responsibilities of both CEO and CM for IProIoT.

- Computation Manager

The CM has the whole picture of IoT network. Each functional component is essential to assign tasks and optimize execution, which could be controlled centrally and locally by CM, or distributed within the network. The details are described as below.

Computation Resource Database: CM discovers and updates all computing related resources within the IoT network. Then it stores their names in different repositories. Function Repository (FR) saves all data processing logics and Executor Repository (ER) has a list of available CEOs. To discover executors, requests and replies are sent between CM and CEOs.

Task Deployment: this procedure is triggered when a user sends a computational request to the IoT network, which involves four steps before real data processing begins. Firstly, task resolution is essential for IProIoT not only to judge whether this request can be processed but also to select appropriate CEO(s). Secondly, possible execution plans are made after checking available computation resources in the database. Thirdly, optimization for execution is performed based on Network Topology. This step is optional depending on current network resources, available execution plans and specific requests and so on. Last but not least, Interest reconstruction combines the name of selected CEO with the computational tasks for dissemination, which ensures the task can arrive at the CEO(s) by name-based routing.

- Computation Executor

The CEO requires data and functions in order to carry out processing. There are two options here: the CM is responsible for Interest resolution and then assigns the data, functions and tasks together to selected CEO; or alternatively, the CEO performs Interest Resolution in order to get all required data and functions from received task. The latter is the one implemented in current design. It is worth to mention that some IoT devices are grouped as Data Source (DS), which means they only provide pure data without any assigned computational tasks. Thus, the CEO sends sub-interests to request data from the DS(s) as well as functions from the FR. Processing starts when all content are ready for CEO(s).

B. *Functional Interest*

Name based forwarding and caching are natively supported by NDN. IProIoT extends this functionality to support in-network computation. This means functions are also named and stored within network as a special type of data. The result is that users are able to express which computing services are required and on what data, without worrying about how and where data is processed.

The functional Interest of IProIoT consists of function names and data names. Each name starts with a slash symbol and is constructed hierarchically. Short dash is used to separate two names and simplifies Interest resolution. For instance, a user wants to know the average temperature of room 1 with two readings collected by sensor1 and sensor2. The corresponding Interest name in IProIoT is:

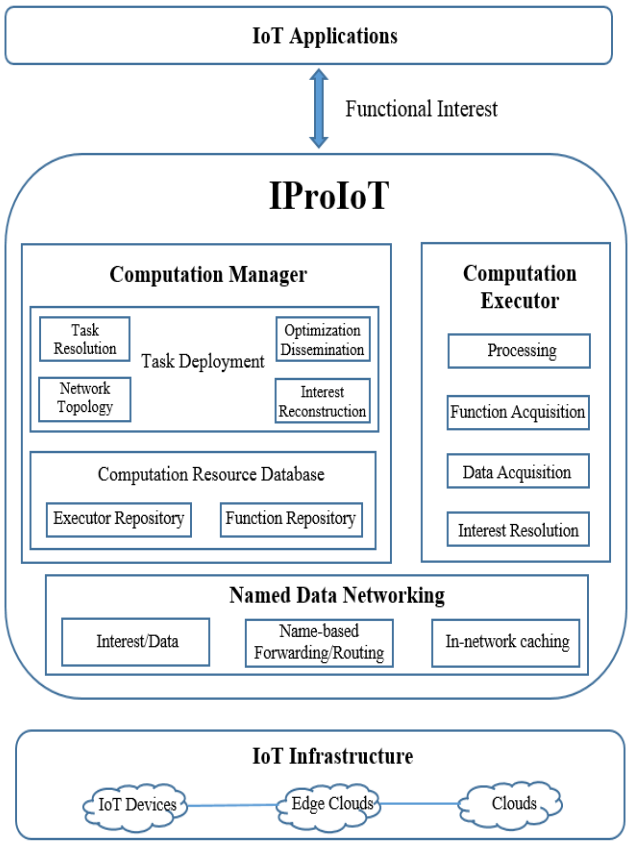


Fig. 2. IProIoT Architecture

/proiot/compute-/mean-/room1/sensor1-/room1/sensor2 (a)

It includes three types of information: (i) /proiot/compute, which requests IProIoT to perform computation; (ii) /mean, the required function name to execute and (iii) /room1/sensor1 and /room1/sensor2, the required data for the computational task. The naming scheme is flexible for combining multiple names for complicated requests. The authors acknowledge that using short dash cannot express very complicated computational tasks, i.e. it is unable to distinguish function with data. We will address this as a part of future work.

C. *Illustration of IProIoT workflow*

For clarity, we use (a) as an example of user's Interest and the CEO(s) will demand necessary content according to specific task. Furthermore, the FR is regarded as an independent role within the IoT network and the DSs provide required data. Fig. 3 shows the whole procedure and the following steps explain in detail:

Step 1: a user sends functional Interest (a) to the IoT network.

Step 2: The CM of IProIoT receives and starts to process (a). It retrieves the real computing task from the user's initial Interest described as (b).

/mean-/room1/sensor1-/room1/sensor2 (b)

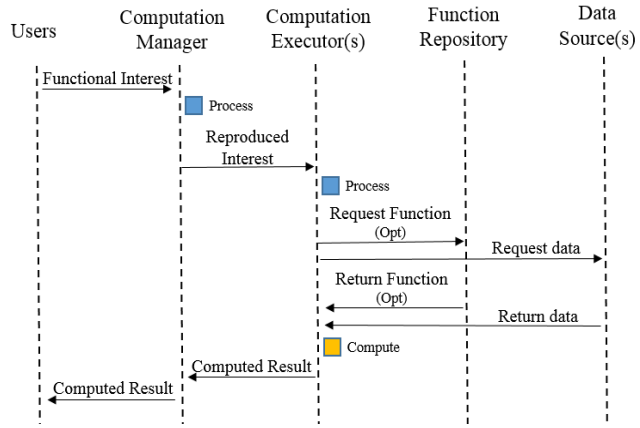


Fig. 3. IProIoT workflow

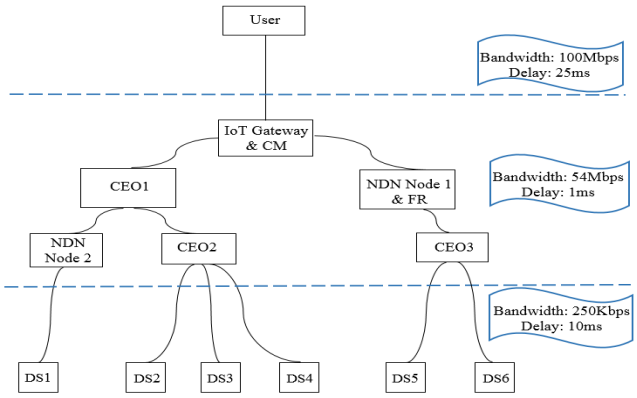


Fig. 4. Testbed Topology

It then checks the availability of nodes within IoT network from the computation resource database. The selected node is the optimized CEO. The next phase is to deploy the computation task. The CM adds the name of the selected CEO, such as: /ceo/node* at the beginning of (b). The reorganised Interest, e.g. as per (c), is sent out by CM.

/ceo/node*-/mean-/room1/sensor1-/room1/sensor2 (c)

Step 3: the matched CEO receives (c) and resolves it to get one function name and two data names: /mean, /room1/sensor1 and /room1/sensor2. It will send out these names as sub-Interests. It is worth to mention that function request is optional because the function could be available from local cache if previous task processing has already used it. Similarly, the CEO will not request data if it's permanent. Otherwise, the CEO needs to request current sensing data for each received task.

Step 4: The FR is responsible for all functions. It will receive the function Interest and return the executable code with name "/mean". Similarly, the matched DSs return the desired data to CEO.

Step 5: The CEO performs the computation when both function and data are obtained. The processed result is then returned to the CM.

Step 6: Finally, the CM returns the processed result to user.

IV. TEST AND EVALUATION

A. Test Design

In order to verify the feasibility of IProIoT, a testbed was set up over Doopnet [16] which enables to use Mininet to manipulate link connections between Docker containers. Each node in our tests is a virtual host simulated by a Docker container running on Linux OS. All nodes are equipped with NDN software and functionality.

The network parameters and topology employed are presented in Fig. 4. We configured three types of data transfer speed (bandwidth + delay): 100 Mbits per second + 25 milliseconds between user network and IoT network, 250kbits per second + 10 milliseconds between sensors and other IoT nodes, and 54 Mbits per second + 1 millisecond within IoT core network. There are six nodes acting as DS's (s1~s6), three nodes simulating CEOs (CEO1~CEO3), and one CM. We assign Function Repository to NDN Node 1 for function storage. The lines between nodes denote they are neighbours in the NDN routing table. All nodes are capable of NDN original functionalities.

We defined two test scenarios: one computes partial IoT data and the other processes data from all data sources. In addition, each scenario has five use cases running on two different network conditions: with and without bandwidth competition. The background traffic is generated by iPerf tool.

 # Test Scenario 1

The first scenario aims to compare the performance of our design with out-of-network processing for IoT data. The following shows 5 use cases and their specific names for test:

(1) /proiot/compute-/max--all

(2) /proiot/compute-/sum--all

(3) /proiot/compute-/min--all

(4) /proiot/compute-/count--all

(5) /proiot/compute-/mean--all

To simulate the IoT out-of-network processing, the user node in Fig. 4 is designated as the executor for every computing task. It has functions stored locally. For example, the above mentioned use case (1) is the Interest received and requested currently. First of all, the user node is responsible for resolving the entire Interest so that getting both function and data names: /max, /s1, /s2, /s3, /s4, /s5 and /s6. The function name (/max) will be looked up at the local FR of the user node. If the function name is not found, the processing will stop immediately because requested functions are not available. Afterwards, the user node will send six sub-Interests to request data with corresponding names. All IoT nodes will forward every Interest until matched Data is found at DSs. The user node waits for all data to be returned and then starts computing.

For in-network processing, two sets of tests are executed to evaluate the performance of IProIoT and the effects of CEO

deployment on the performance. In the first set of tests, the computation task is assigned to a specific CEO (CEO3) to process.

In the second set of tests, all the CEOs are involved in the computation. As the task involves all sensing data, the CM makes the optimized execution plan for distributed processing. In details, CEO3 is in charge of s5 and s6 (to get sub-result-i), CEO2 deals with s2, s3 and s4 (to get sub-result-ii) and CEO1 deals with s1 and sub-result ii (to get sub-result-iii). The IoT Gateway aggregates sub-result-i and sub-result-iii as the final result and then return to the user node. It is worth to mention that current optimization of IProIoT is manually implemented, which will be an important part of our future work.

 # Test Scenario 2

The second scenario is designed to illustrate show how the location of CEO effects on the IProIoT performance when partial sensing data are requested. The use cases are shown as (6) – (10). In the tests, s2, s3 and s4 are selected as the data source for the computation. CEO2 and CEO3 are selected as the computation node in two separate sets of tests. CEO2 is the most appropriate computation node because it is not only the nearest computation-capable node within the IoT network but also on the routing path to user node. In contrast, CEO3 is a multiple-hop neighbour from all required DSs.

(6) /proiot/compute-/max-/s2-/s3-/s4

(7) /proiot/compute-/sum-/s2-/s3-/s4

(8) /proiot/compute-/min-/s2-/s3-/s4

(9) /proiot/compute-/count-/s2-/s3-/s4

(10) /proiot/compute-/mean-/s2-/s3-/s4

B. Test Result and Evaluation

In this section, we present the results of our test bed simulations. We focus on evaluating network traffic as well as processing time for all test cases.

 # Network Traffic

In terms of processing all sensing data, the multiple CEOs execution approach of IProIoT produces around 4485 bytes (Fig. 5 solid filled columns in red) traffic in all use cases. This number dramatically increases to 8300 bytes (green columns in Fig. 5) when the same tasks are executed outside of IoT network. The reason is IProIoT combines data and function

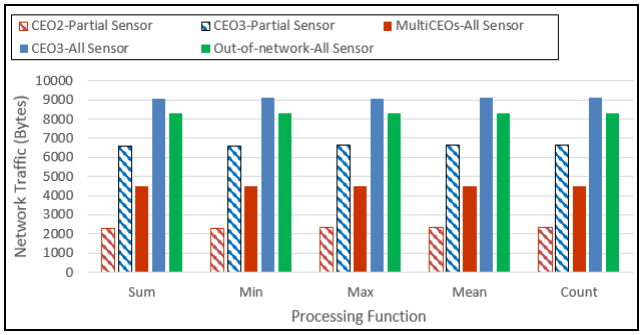


Fig. 5. Network Traffic for Both Scenarios

within one Interest packet and returns one Data packet for each task. In the case of out-of-network processing, multiple Interests will be sent in IoT network and corresponding Data packets will be delivered out of the network. This results in duplicated overhead from each packet.

However, in-network processing performs worse than out-of-network approach if the computation resources is not assigned properly (results for CEO3-All Sensor tests). As the blue solid columns in Fig. 5 show that IProIoT transmits more data within IoT because the selected CEO is far away from most DSs. To this end, the authors argue that with computational resource optimization, in-network processing could significantly lower IoT network traffic than out-of-network approach.

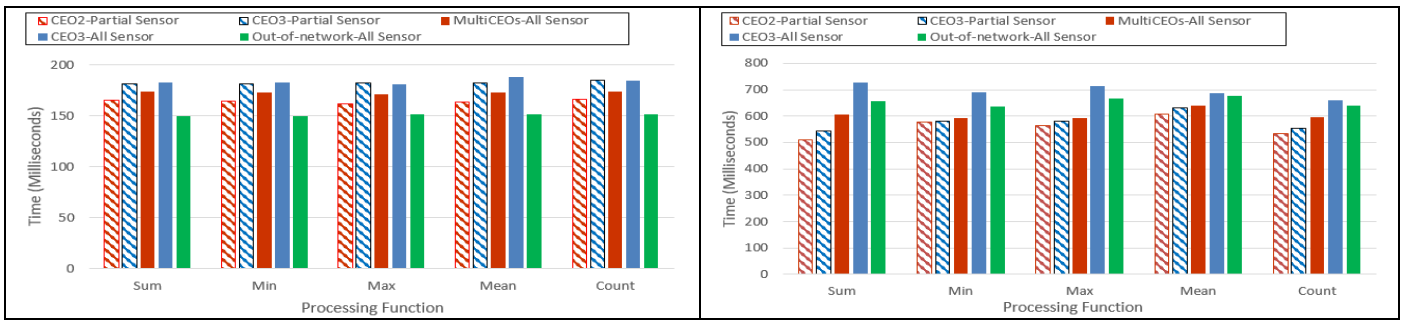
Speaking of process partial IoT sensing data, columns in Fig. 5 with red pattern filled are the network traffic cost by CEO2, which is around 2300 bytes. However, more than 2.8 times of the traffic is delivered if the same tasks are assigned to CEO3 (blue pattern fill columns). As CEO2 is closer to DSs compared with CEO3, data for computation traverses less devices so that much network traffic and resource will be saved, especially when the network size increases. In a word, our design is capable of assigning the task to the most appropriate nodes for better performance.

 # Processing Time

Fig. 6 (a) and (b) are the processing time for both scenarios running with and without background traffic.

- Without Background Traffic

For test scenario 1, out-of-network processing costs 150



(a) Without Background Traffic

(b) With Background Traffic

Fig. 6. Processing Time for Both Scenarios

milliseconds, which is 20 milliseconds less than IProIoT with optimization (multiple CEOs involved), see Fig. 6 (a) red and green solid colour columns respectively. In the case of out-of-network approach, there is almost no delay on resolving requests as the user is responsible for the process. The whole processing delay roughly equals to the time on data transmission. However, for our optimized execution plan, both CM and CEOs join in task resolution and processing, which costs a few more milliseconds delay. The performance is worse when only one computing node (CEO3) fulfils the whole task. More time is spent on delivering data on longer routes. In this way, it may infer that transferring data out of IoT is a good choice when the network condition is ideal enough in real world.

For test scenario 2, CEO2 spends less time than CEO3. It is mainly caused by the number of links between the CEO and the DSs. Fig. 6 (a) shows, when the network link load is empty, the delay is increased from 160 milliseconds by CEO2 to 180 milliseconds by CEO3. Because data needs to transfer more hops to reach CEO3.

- With Background Traffic

For test scenarios 1, the test results are drawn as solid colour columns in Fig. 6 (b). In this situation, the out-of-network approach spends more time (30-70 milliseconds) than multiple CEOs collaboration of IProIoT. Moreover, the delay will be increased if the packet is lost in poor network conditions because the user needs to resend the same request until the processed data is obtained. In-network processing without optimization (only CEO3 involved) still has longest delay compared with other two approaches due to waiting on more data to be delivered.

For test scenario 2, CEO2 keeps performing better than CEO3. Furthermore, the difference of processing time between CEO2 and CEO3 is bigger (about 15-30 milliseconds) when the network link is shared with other traffic, shown as Fig. 6 (b) pattern filled columns.

V. CONCLUSION

This paper presents an in-network processing and information-centric framework (IProIoT) for IoT networks. As the current Internet is host-centric, which poorly supports the on-path data process requirement of IoT applications, we employ ICN as the underlying network support. There is no doubt that complicated IoT applications benefit from running application logic in the Cloud servers, however, in-network computation is complementary for handling simple tasks with better performance. Thus, the proposed design enables appropriate IoT devices to undertake in-networking data process work. Moreover, computation management scheme optimizes network resource allocation to offer best execution plan. A set of use cases for two scenarios have proven the improved performance our design in significantly saving network traffic and process delay.

As future work, our plan is to make the programmable framework more flexible and powerful, such as: process complex data tasks and achieve autonomous network optimization. A testbed with larger scale will be set up on physical devices to study improved performance in real IoT scenarios.

ACKNOWLEDGMENT

This publication has emanated from research supported by research grants from Institutes of Technology Ireland (IOTI) under Postgraduate Scholarship Initiative 2014, Athlone Institute of Technology under President's Seed Fund 2016, and Science Foundation Ireland (SFI) under Grant Number 13/SIRG/2178.

REFERENCES

[1] Cisco, Available: "Cisco visual networking index: Global mobile data traffic forecast update, 2014–2019," Accessed: Dec-2015.

[2] D. Alessandrelli, M. Petracca, and P. Pagano, "T-res: Enabling reconfigurable in-network processing in IoT-based WSNs," *2013 IEEE* International Conference on Distributed Computing in Sensor Systems (DCOSS), Cambridge, MA, pp. 337–344, 2013.

[3] Y. Zhang, et al, "Requirements and challenges for IoT over ICN", IRTF ICN Research Group, Nov. 2015.

[4] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, kc claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named Data Networking," SIGCOMM Comput Commun Rev, vol. 44, no. 3, pp. 66–73, 2014.

[5] M. Sifalakis, B. Kohler, C. Christopher, and C. Tschudin, "An information centric network for computing the distribution of computations," in Proceedings of the 1st international conference on Information-centric networking, pp. 137–146, 2014.

[6] W. Drira and F. Filali, "NDN-Q: An NDN Query Mechanism for Efficient V2X Data Collection," 2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking Workshops (SECON Workshops), pp.13-18.

[7] M. Amadeo, C. Campolo and A. Molinaro, "Multi-source data retrieval in IoT via named data networking," ICN'14 Proceedings of the 1st International Conference on Information-centric Networking, ACM New York, USA, pp. 67-76, 2014.

[8] M. Kovatsch, S. Mayer, and B. Ostermaier, "Moving application logic from the firmware to the cloud: Towards the thin server architecture for the internet of things," 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), Palermo, Italian, pp. 751–756, 2012.

[9] G. Xylomenos, C. N. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, G. C. Polyzos, and others, "A survey of information-centric networking research," Commun. Surv. Tutor. *IEEE*, vol. 16, no. 2, pp. 1024–1049, 2014.

[10] Microsoft Azure IoT Suite, http://www.microsoft.com/en-ie/server-cloud/internet-of-things/, Accessed on 24th June 2016.

[11] Amazon AWS IoT Platform, https://aws.amazon.com/iot/, Accessed on 24th June 2016.

[12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Future Generation Computer Systerm., vol. 29, no. 7, pp. 1645–1660, September 2013.

[13] Y. Ye, B. Lee, , N. Murray and Y. Qiao, "PIoT: Programmable IoT using information centric networking," IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, 2016.

[14] M. Amadel, C. Campolo, A. Iera and A. Molinaro, "Information Centric Networking in IoT scenarios: the Case of a Smart Home," IEEE ICC 2015 SAC – Internet of Things, pp. 648–653, 2015.

[15] M. Amadel, C. Campolo and A. Molinaro, "Information Centric Networking for Connected Vehicles: A Survey and Future Perspectives," IEEE Communication Magazine, pp. 98–104, 2016.

[16] Y. Qiao, X. Wang, G. Fang and B. Lee, "Doopnet: An emulator for network performance analysis of Hadoop clusters using Docker and Mininet", IEEE Symposium on Computers and Communication (ISCC), Italy, 2016.

[17] Q. Wang, B. Lee, , N. Murray and Y. Qiao, "CS-Man: Computation Service Management for IoT In-Network Processing," the 27th Irish Signals and Systems Conference (ISSC), Derry, Northern Ireland, 2016.