# Demo: Multiple Network Function Execution in ICN-based Crowdsensing Service

Yoji Yamamoto
Osaka University
y-yamamoto@ist.osaka-u.ac.jp

Yuki Koizumi
Osaka University
ykoizumi@ist.osaka-u.ac.jp

Toru Hasegawa
Osaka University
t-hasegawa@ist.osaka-u.ac.jp

Giulio Rossi
University of Rome "Tor Vergata"
giulio.rossi@uniroma2.it

Andrea Detti
University of Rome "Tor Vergata"
andrea.detti@uniroma2.it

Onur Ascigil
University College London
o.ascigil@ucl.ac.uk

Ioannis Psaras
University College London
i.psaras@ucl.ac.uk

## ABSTRACT

The paper describes the prototype of object search service wherein multiple IoT devices execute the network function of identifying the object in the specified area.

## CCS CONCEPTS

• **Networks** → **Network design principles**; **Naming and addressing**.

## KEYWORDS

Crowdsensing, Edge computing, ICN

**ACM Reference Format:**
Yoji Yamamoto, Yuki Koizumi, Toru Hasegawa, Giulio Rossi, Andrea Detti, Onur Ascigil, and Ioannis Psaras. 2019. Demo: Multiple Network Function Execution in ICN-based Crowdsensing Service. In *6th ACM Conference on Information-Centric Networking (ICN '19), September 24–26, 2019, Macao, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3357150.3357407

## 1 INTRODUCTION

Location-based *crowdsensing service* wherein IoT devices upload data obtained at their locations has been becoming popular; however, computation of uploaded data in a cloud incurs large processing delay and data transfer. While edge computing based on network functions [2] is promising, it has still an issue about where they are executed in the case that computation is performed for data pieces obtained by multiple IoT devices. To address this issue, the paper implements the object search service wherein a network function is executed by multiple edge routers and IoT devices.


*ICN '19, September 24–26, 2019, Macao, China*

ACM ISBN 978-1-4503-6970-1/19/09.
https://doi.org/10.1145/3357150.3357407

## 2 CROWDSENSING SERVICE

### 2.1 Design Rationale

The paper discusses the service wherein the same network function is executed for large location data pieces in a specified area. An example service is *object search* wherein a target object like a lost dog or a lost child is detected from images obtained by multiple cameras in the street. The network function is designed so that it is executed at *computing nodes* which have location data pieces, such as routers which cache such location data pieces and IoT devices themselves. To achieve this goal, the network function has a list of names of location data pieces as its arguments. We call the names *location names* and location data pieces are obtained by IoT devices. A computing node locally executes the network function for location data pieces in its cache, and otherwise it invokes the network function to downstream computing nodes including IoT devices. The way of execution avoids retrieval of large location data and enables it for multiple computing nodes to execute the network function in parallel.

### 2.2 Service Implementation

(1) Naming Scheme [1]: A name of an Interest packet is a routing name followed by tagged names like `/edge0/#f:detect #a:[/3/0/0, 3/0/1,/3/0/2] #a:[/dog]`. Tag `#f:` specifies a name of a network function and tag `#a:` specifies a list of arguments. Network function `detect` is calculated for all the combinations of the lists' names like the three pairs (`/3/0/0` and `/dog`), (`/3/0/1` and `/dog`) and (`/3/0/2` and `/dog`). A computing node retrieves data pieces specified by such names according to ICN. Here, names of location data pieces called *location names* are represented according to Z-order [3]. The name specifies a location and a size of a square and this square can be divided to four small squares by adding one component like `/3/0/1`.
(2) Location Update: IoT devices and edge routers work as both workers of crowdsensing and computing. They periodically upload their locations and roles such as the *sensor* and the *computing node* to the central server.
(3) Protocol Sequence: An example protocol sequence is illustrated in Fig. 1. First, the requestor downloads the current map of workers' locations and then chooses the name of the computing node and

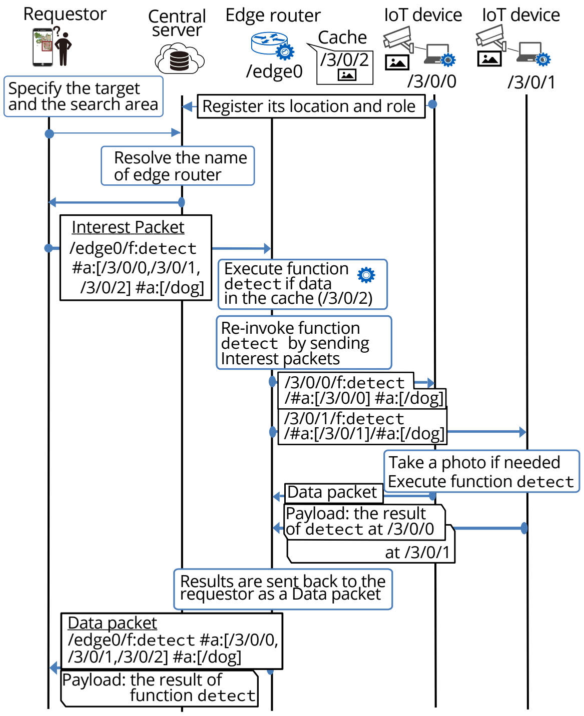


**Figure 1: Protocol Sequence**

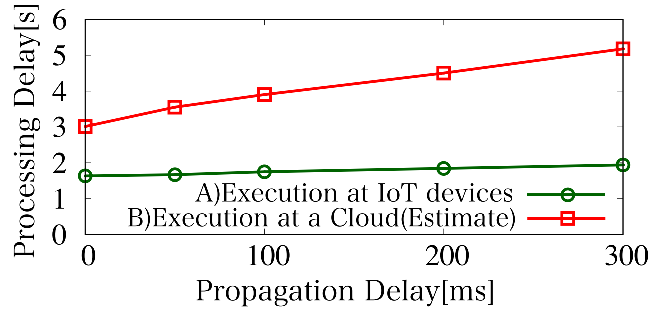


**Figure 2: Processing delay and propagation delay**

the location names of interest. In the figure, computing node name /edge0 and location names /3/0/0, /3/0/1 and /3/0/2 are chosen. Then the requestor sends the Interest packet which has them and target object's name /dog as the arguments.

Second, when computing node /edge0 receives the Interest packet, it checks whether each location data piece specified by each location name in the list is in the cache or not. If the location data piece is in the cache, the requestor executes function detect for the location data piece and argument /dog. Otherwise, the location data piece need be retrieved from the corresponding IoT device. Rather than retrieving it, the requestor recursively invokes the network function at the IoT device. In this case, the routing name is set at the location name of the IoT device like /3/0/1. Finally, all the results of function execution are replied as the list to the requestor. The requestor knows the location at which the target object is identified.

(4) Implementation: Edge routers, IoT devices and servers are implemented on PC-based platforms and Named Data Networking Forwarding Daemon (NFD) is used to forward packets. The machine-learning based object detection module in OpenCV is used for the object detection.

## 3 EVALUATION

(1) Experiments: This section compares A) the execution at IoT devices and B) that at a cloud in terms of processing delay. The experiment conditions are as follows: five PCs of a requestor, an edge router and three IoT devices are connected via a LAN switch. NFD and the implemented software are installed at all the PCs. PCs have an Intel 1.90 GHz CPU device and the bandwidth of each link is 1 G Byte/s. Images obtained by cameras are 4K images and each image is about 5Mbytes. Propagation delay is inserted at Network Interface Cards (NICs) and it is changed between 0 ms and 300 ms. The time of identifying an object is about 1.05 seconds, which is experimentally measured at the above PC.

We measure processing delay in the two cases A) and B). In the case of A), function detect is executed at the IoT devices, whereas in the case of B), it is executed at the requestor. The processing delay is a sum of computation time at PCs, propagation delay and data transfer time. Please note that case B) assumes execution in a cloud and that the processing delay includes data transfer time of three 4K images.

(2) Results: Figure 2 shows the averages of measured processing delay for 10 trials. The propagation delay is changed from 0 ms to 300 ms. Obviously, A) the network function execution at IoT devices achieves shorter processing delay than B) that at a cloud. The observations are summarized below: First, the processing delay of A) is about 1.63 seconds when the propagation delay is 0 seconds. It is a little longer than the time of function detect, i.e., 1.05 seconds. This is because three IoT devices execute the function in parallel. Second, avoiding large image data transfer contributes to reducing processing delay of A) . In the case of A), the only small size results are replied to the requestor.

## 4 CONCLUSION

The paper describes the porotype implementation of a framework of executing a network function for location data pieces obtained by multiple IoT devices.

## ACKNOWLEDGMENTS

The work for this paper was performed in the context of the Horizon2020/NICT EU-JAPAN ICN2020 project Grant Agreement No. 723014 and Contract No. 184.

## REFERENCES


[1] M Ascigil, Sergi Reñé, George Xylomenos, Ioannis Psaras, and George Pavlou. 2017. A keyword-based ICN-IoT platform. In *Proceedings of ACM ICN*. 22–28.
[2] Michał Król and Ioannis Psaras. 2017. NFaaS: named function as a service. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*. ACM, 134–144.
[3] Yoshiki Kurihara, Yuki Koizumi, and Toru Hasegawa. 2019. Computing Node Selection for Location-based Service in NDN Networks. In *Proceedings of IEEE LANMAN 2019*. IEEE.