Cooperative and priority based on dynamic resource adaptation method in wireless network

Kazuaki Togawa* and Koji Hashimoto

Graduate School of Software and Information Science,
Iwate Prefectural University,
152-52, Sugo, Takizawa, Iwate,
020-0693, Japan
Email: g231o022@s.iwate-pu.ac.jp
Email: hashi@iwate-pu.ac.jp
*Corresponding author

Abstract: In recent years, network traffic has been increasing and when large events or natural disasters occur more network resources are requested at end points of network. Also, with the spread of smart devices can communicate with high-speed such as LTE, anyone are becoming to be able to communicate with high-speed. In order to efficiently handle traffic that locally and temporarily increases, it is effective to utilise smart devices owned by users. However, because there is a limit to the amount of the traffic that a smart device can handle, it is necessary to cooperate smart devices, nevertheless a system that cooperates smart devices and aggregates network resources has not been established. In this paper, we proposed a dynamic resource adaptation method that aggregates the network resources of smart devices and increases the available bandwidth. In evaluation experiments, a relationship between the amount of smart devices and network throughput was evaluated.

Keywords: priority control; resource adaptation method; software defined networking; SDN; wireless network.

Reference to this paper should be made as follows: Togawa, K. and Hashimoto, K. (2018) 'Cooperative and priority based on dynamic resource adaptation method in wireless network', *Int. J. Space-Based and Situated Computing*, Vol. 8, No. 1, pp.40–49.

Biographical notes: Kazuaki Togawa is a Master's student in the Graduate School of Software and Information Science at the Iwate Prefectural University of Japan. He received his Bachelor's degree from the Faculty of Software and Information Science at the Iwate Prefectural University in 2016. His main interests include software defined networking, network management and mobile networks. He is a student member of the Information Processing Society of Japan (IPSJ).

Koji Hashimoto is a Professor in the Graduate School of Software and Information Science at the Iwate Prefectural University of Japan. He received his PhD from the Tohoku University of Japan in 2001. His research interests include wide range of computer networks, especially audio video communication, live streaming protocol and communication middleware. He has been developing distributed audio-video streaming system for multi-point video communications with dynamic configuration functions of on demand remote controllable audio-video mixer. He is a member of the Information Processing Society of Japan (IPSJ) and the IEEE.

1 Introduction

In recent years, we have become able to obtain information through the Internet regardless of time and place since smart devices such as smart phones and tablets become widespread. Today we use the various internet services such as social network services, cloud applications, media streaming services, etc. Additionally, with the advent of IoT, there has been a gradual growth of devices connected to the Internet. As a result, the amount of network traffic is rapidly increasing. Cisco surveyed that global mobile data traffic reached 7.2 exabytes per month at the end of 2016 and has grown 18 fold over the past five years (Cisco Visual Networking Index, 2017).

Furthermore, the amount of network traffic locally and temporarily increases when large events are held or large natural disasters occur than normal times. In large events, broadcasting the state of events through media streaming services has becoming common. In the future it is expected that media streaming services using high resolution media such as 4K resolution, as a result, not only the whole network but also local and temporary traffic will increase more and more. Moreover, in a large natural disaster, NTT Docomo is mobile network operator in Japan surveyed that the packet traffic doubled in the Great East Japan Earthquake in 2011 compared with normal time (NTT Docomo, 2011). In Addition, it is reported that the mobile data traffic around the Japan Self-Defence Forces bases

increased than normal times in the Kumamoto earthquakes in 2016 (KDDI, 2016).

As described above, packets are delayed or lost when there is traffic request exceeding the network resources since almost all Internet services do not guarantee bandwidth due to the best effort services. In a large natural disaster, if important packets are delayed or lost, there is a possibility of life-threatening.

To this day, many researches and developments have been made to handle with an increase in network traffic. However, many methods have focused on an increase of network traffic in the whole network, and have not focused on a local and temporary increase of network traffic. For this reason, it is currently mobile network operators dispatch mobile base station vehicles to handle the local and temporary increase of the network traffic. However, since it is not always possible to dispatch the mobile base station vehicles to handle the increasing network traffic at the time of large events or natural disasters, it is necessary that a system to cooperate the available equipment and solve the above mentioned problems is established.

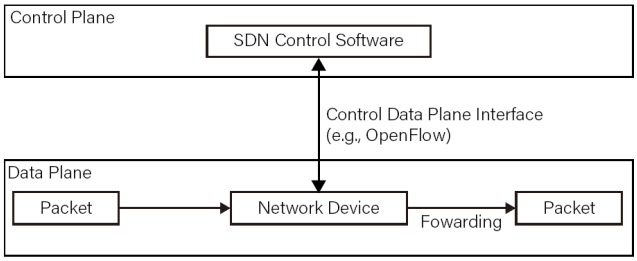
On the other hand, smart devices supporting LTE and LTE-Advanced which specifications are standardised by the Third Generation Partnership Project (3GPP, 2018) are spread, environments anyone can communicate with high-speed have been gearing up. Moreover, in 5th generation wireless systems that are next generation communication standard, it is expected to further to improve the communication speed because the 10 Gbps transmission experiment in the outdoor mobile environment has been successful.

For these reasons, it is effective to utilise smart devices which can communicate with high-speed for the local and temporary increases of the network traffic. Furthermore, many smart devices have tethering functions, and this functions make it possible to use the network resources of smart devices with other devices, however, there is a limit to the amount of traffic that can be handled by a smart device. For example, when broadcast of events at multiple points, it is assumed that the broadcasting at the same time with high quality is difficult. If it becomes possible to aggregate network resources of smart devices as necessary, broadcasts at multiple points will be possible with high quality.

Furthermore, handling packets according to the priority of information is necessary in addition to aggregating network resources of smart devices. If it only increases network resources, it is possible that bandwidth will be used for low priority information. By processing according to the priority, the increased available bandwidth is more effectively used.

Moreover, it is preferable that management of smart devices that provide network resources can be centrally managed by a single equipment. For example, when there are many necessary network resources, the number of smart devices providing network resources also increases. If smart devices manage the network configuration by themselves, it is necessary to change the network configuration on each device each time a smart device is added. It is possible to easily change the network configuration by centrally managing smart devices with a single equipment.

Figure 1 SDN overview



Additionally, in recent years, a new network technology called software defined networking (SDN) has been studied. Figure 1 shows the architecture of SDN. SDN's features is separating the control plane and the data plane. The control plane decides how to handle the packets. The data plane forwards the packets according to decisions that the control plane makes. For that reason, the network system using the SDN can forward the packets scalable and dynamically. And, it can change the network system compositions, too. In addition, it is possible to centralised control by a single equipment. Altogether, the SDN can configure a centralised programmable network which controls the entire network dynamically by a software. Therefore, SDN seems to be able to dynamically aggregate network resources and the problems can be solved.

In this paper, we propose a dynamic resource adaptation method that aggregates network resources and increases the available bandwidth using SDN. We construct a prototype system using the OpenFlow which is standardised by the open networking foundation to realise SDN (Open Networking Foundation, 2018).

The rest of this paper is organised as follows. Section 2 introduces related works. Section 3 explains our proposed system overview, system configuration and architecture. Section 4 describes the details of our proposed method. Section 5 explains a prototype system. Section 6 describes the experimental evaluation results of the proposed method. Section 7 presents the summary of this paper and future work.

2 Related work

There are several studies in the literature focus on routing and resource management.

Sato et al. (2013) propose a network system called never die network (NDN). NDN makes it possible to communicate in any situation by dynamically switching multiple networks connecting to the internet. However, NDN cannot aggregate network resources, although it has multiple Internet access networks such a satellite communication, 3G and FTTH. Even if there are multiple Internet access networks, which is not broken down or does

not occur network congestions, it is difficult to aggregate the network resources.

Huang et al. (2015) propose a network system which improves the data transfer performance of the GridFTP by allocating parallel TCP streams to different paths. However, in this system, paths which transfer the data are determined in the paths existing before the transfer. It is difficult to dynamically add network resources and change the number of paths. Celenlioglu and Mantar (2015) propose a scalable routing and resource management model for SDN-based intra-domain networks. However, the system considers only pre-established paths.

There are studies which avoid network congestions and failures by switching different access networks, improve the data transfer performance by aggregating multiple paths. However, currently, there are no methods to dynamically adapt network resources and to increase the available bandwidth. In order to increase the available bandwidth, it is necessary a new network system that aggregates network resources of smart devices.

3 System overview

Based on related works, we describe the system overview. In this paper, we propose the dynamic resource adaptation method that increases the available bandwidth. Generally, it is not easy to increase the available bandwidth. Therefore, we aim to increase the available bandwidth by realising the system that aggregates network resources of smart devices. Moreover, we realise a mechanism to determine and forward the priority of packets. In this system, IP address and port number are used to determine priority and register in advance. Our method determines the priority stream based on registered rules, and forward packets on the priority basis. Additionally, we realise a dynamically network resources adaptation to detect that a smart device is connected to a switch. In addition, in order to handle sudden requirements of network resource, we realise a system that the construction is easy by using general purpose computers. It seems that our proposed system can handle sudden requirements by aggregating network resources of smart devices when large natural disasters occur or large events are held. In summary, we described our proposed system overview. We realise the network system that dynamically detects and aggregates network resources of smart devices, handling the packets according to them of priority and increases the available bandwidth by using general purpose equipment.

3.1 System configuration

Figure 2 shows the system configuration. In this figure, the broken line shows the control message flow, and the solid line shows the data flow, in addition the thickness of the lines show the data amount. The proposed system constructs of three components: network controller, network switch, and cooperative device. We explain each component as follows.

1 Network controller

Network controller always connects to network switch and cooperative device through the internet, and manages all network switches and cooperative devices. For example, it determines IP routing rules and selects cooperative device for packet forwarding. Moreover, determine priority of streams by registered information.

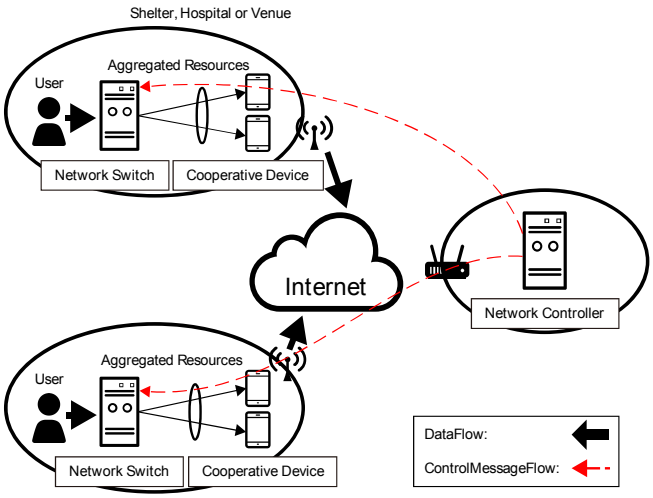
2 Network switch

Network switch is installed in places where a lot of network resources are requested like in hospitals, shelters, event venues, etc. Moreover, it operates as the Internet gateway and forwards packets by rules which are determined by network controller.

3 Cooperative device

Cooperative device is smart device that provides owned network resources to network switch. It connects with network switch, and operates as a connection node. Cooperative device receives packets sent by users through network switch and transfers the forwarded packets.

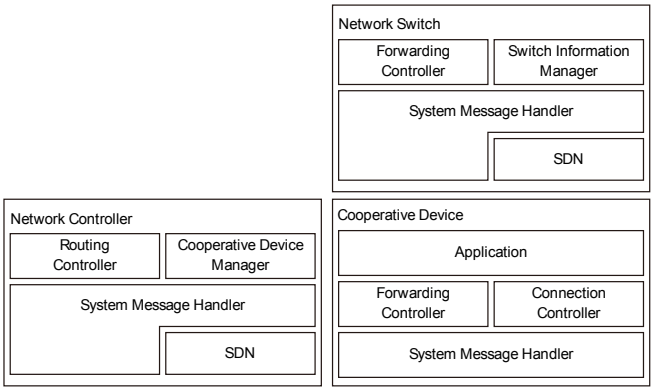
Figure 2 System configurations (see online version for colours)



3.2 System architecture

Figure 3 shows system architecture of proposed system. First, we explain the architecture of network controller. Network controller consists of cooperative device manager, routing controller, system message handler and SDN. Cooperative device manager manages cooperative devices connected to network switch. Routing controller controls routing that preferentially transfers packets with higher priority and cooperates with cooperative device manager to determine Cooperative device to which packets are transferred. System message handler controls system messages for controlling network switches and cooperative devices. SDN handles the OpenFlow protocol with system message handler.

Figure 3 System architecture



Next, we explain the architecture of network switch. Network switch consists of switch information manager, forwarding controller, system message handler and SDN. Switch information manager manages information of cooperative devices connected to network switch. The managed information is sent to cooperative device manager of network controller. Forwarding controller forwards packets according to rules determined by routing controller of network controller.

Finally, we explain the architecture of cooperative device. Cooperative device consists of application, forwarding controller, connection controller, and system message handler. Application manages all modules of cooperative device. Forwarding controller of cooperative device transfers packets forwarded from network switch. Connection controller cooperates with switch information manager of network switch, and communicates connections of cooperative device with network switch.

4 Cooperative device selection algorithm

Figure 4 shows the functional flow of the proposed system. The system performs the following steps:

1 Cooperative device is connected to network switch.

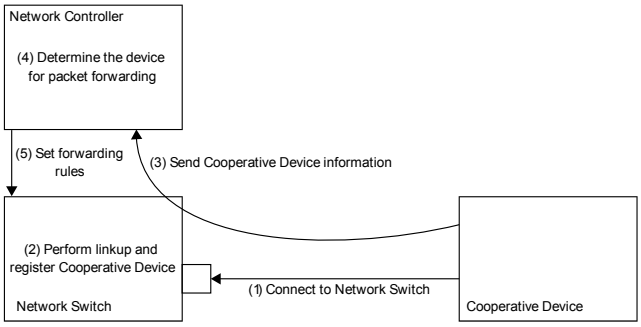
2 Network switch registers cooperative device as a connection node and performs linkup. Thus, the connection between network switch and cooperative device is established.

3 Cooperative device sends self-information to network controller through network switch. This information is a MAC address on the side of network switch and the established port number of network switch.

4 Network controller determines cooperative device as a connection node to transfer packets. This determination is executed when a new flow to be described later is detected.

5 Network controller notifies network switch of the forwarding rule determined at step 4. Network switch transfers packets according to the forwarding rules notified from network controller.

Figure 4 Functional flow of the proposed system



Next, we describe the way how the proposed system selects a device for packet forwarding. As shown in Figure 2, the proposed system aggregates network resources of cooperative devices, and distributes TCP or UDP streams from users to each cooperative device through network switch. Therefore, the streams considered by the cooperative device selection algorithm are transferred to the WAN side through network switch.

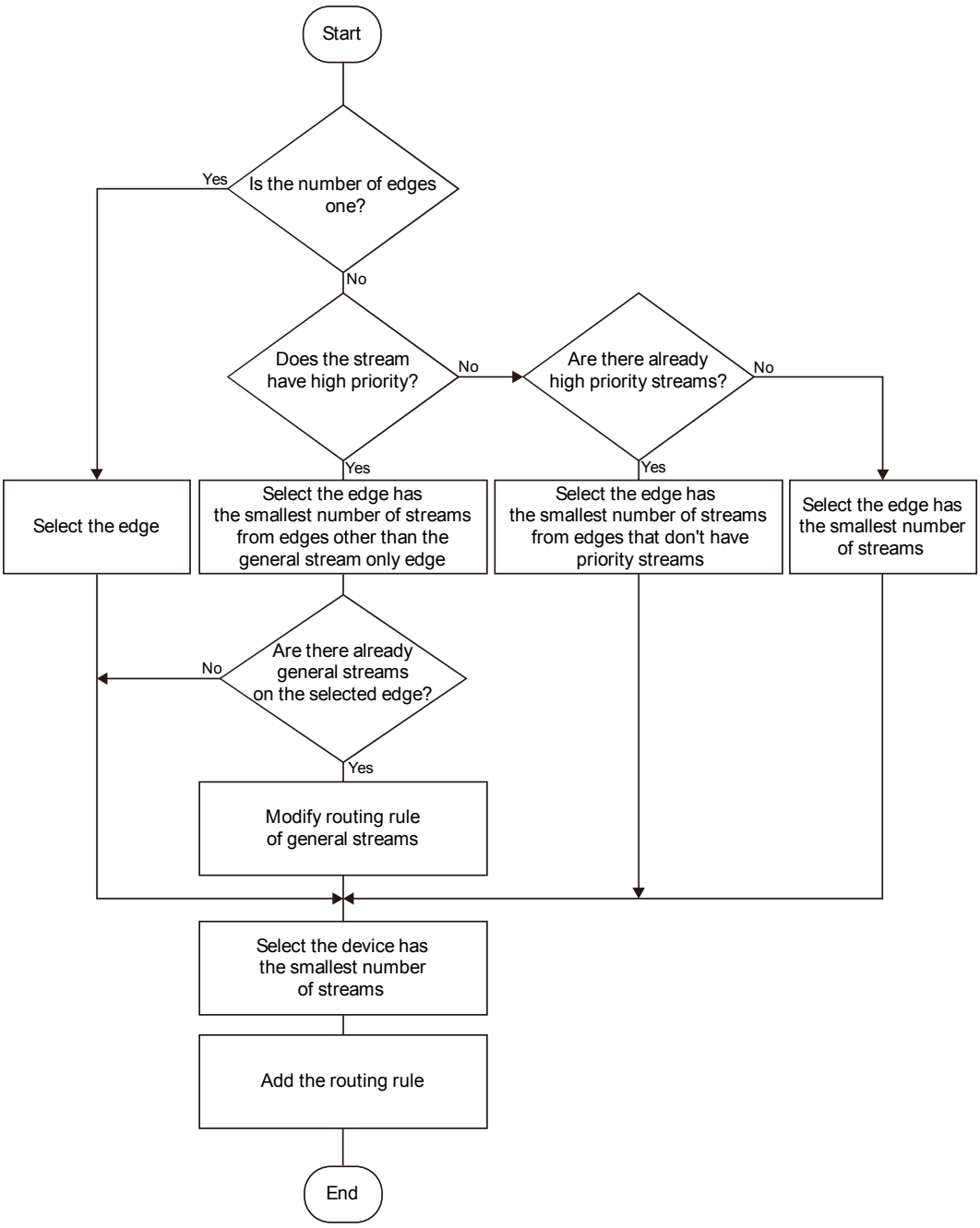
On the other hand, there are a plurality of users who communicate through network switch in parallel at the same time. In order to distribute the streams to each cooperative device, the proposed system distinguishes them using the source MAC address and the source port number. A pair of this source MAC address and the source port number is called flow by the proposed system and the streams are managed. When a new flow is detected, the cooperative device selection algorithm is performed to determine cooperative device used for forwarding traffic.

Figure 5 describes the cooperative device selection algorithm. On the assumption that, in the cooperative device selection algorithm, there are multiple edges and cooperative devices. Furthermore, cooperative devices are connected to edges are connected to the WAN. Moreover, as an example, the edge in Figure 5 shows a base station in the cellular network or an AP in the prototype system to be mentioned below. With this algorithm streams are distributed to cooperative devices and the edge to which cooperative device is connected.

First, check the number of configured edges. At this time, when there is only one edge, it is selected the edge. Then, it checks whether the stream requested for connection is high priority. The priority of streams is determined by a pair of the destination IP address and the destination port number.

If the stream is high priority, it selects an edge has the smallest number of streams from edges other than a general stream only edge. It is because that, high priority streams are sent by edges with fewer the number of streams. Also, the general stream only edge exists because when multiple high priority streams exist, general streams are aggregated in one edge. In addition, if general streams are there already on the selected edge, the rules are modified so as to send through the edge not having high priority streams. By this way, general streams that had already sent to the selected edge are aggregated to the general stream only edge.

Figure 5 Cooperative device selection algorithm



If the stream requested for connection is a general stream, it is checked whether high priority streams have already been sent. If high priority streams have already been sent, it selects an edge has the smallest number of streams from edges that do not have high priority streams. As a result, the bandwidth for the priority stream is ensured, and the general streams are sent out. Moreover, as mentioned above, there is the general stream only edge, so there are edges without the high priority streams that can be selected.

On the other hand if high priority streams do not have already been sent, it selects an edge has the smallest number of streams. In this way, the edge for sending streams is selected in four ways.

When an edge is selected, it selects cooperative device in the devices connected the selected edge. Finally, it adds a rule to send through the selected edge and the cooperative device in the algorithm.

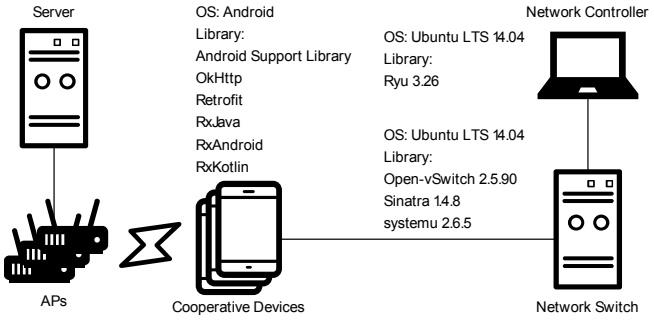
5 Prototype system

To verify the effectiveness of the proposed system, we developed a prototype system by using general purpose equipment. Figure 6 shows prototype system overview. In this section, we describe the prototype system. Network controller and network switch consist of general purpose

equipment and Android devices are used as cooperative devices. We developed the prototype system based on the Ryu (2018) for a SDN framework and the Open vSwitch (OVS) (2018) for a software switch. As it can be noticed for Figure 5, the network switch and the cooperative devices are controlled by a single equipment.

Moreover, we developed a RESTful application for sending cooperative device information to network controller through network switch by using several libraries: OkHttp (2018), Retrofit (2018), ReactiveX (2018) and Sinatra (2018). The Systemu (2018) is used to perform linkup and register cooperative device as a connection node.

Figure 6 Prototype system overview



We established the links between cooperative device and network controller using the build-in USB tethering function of Android. There are three kinds of tethering function of Android: the WiFi tethering, the Bluetooth tethering and the USB tethering. Android cannot turn on the WiFi connection when the WiFi tethering function is on. Moreover, a battery has a short battery life. The Bluetooth tethering has a long battery life than the WiFi tethering, however, low throughput. The USB tethering can be charged even if it is ON, and high throughput. The USB tethering seems to be the best in the three types of tethering functions.

Table 1 Hardware specification

Network controller	CPU: Intel Core i7-X940
	RAM: 8GB
	OS: Ubuntu 14.04 LTS
Network switch	CPU: Intel Core i7-6700
	RAM: 16GB
	OS: Ubuntu 14.04 LTS
Cooperative device	Nexus5X
	CPU: Qualcomm Snapdragon 808
	RAM: 2GB
	OS: Android 7.1.1
Access point	Buffalo WHR-1166DHP3

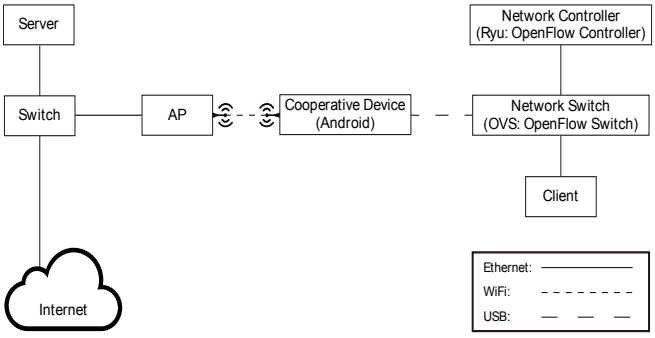
Table 1 shows hardware specification of prototype system. In addition, all NICs of general purpose equipment were built with 1000 Base-T, and the connection between access points and smart devices were built with the IEEE802.11ac. The performance evaluation experiment using this prototype system will be described in Table 1.

6 Performance evaluation

6.1 Experiment to distinguish priority

We constructed a testbed network by using general purpose equipment, and evaluated our proposed system. In the experiments, we installed the iPerf (2018) on the client and the server to measure the throughput between the client and the server.

Figure 7 Overview of testbed network configuration for experiment to distinguish priority



First, we performed experiment of distinguish priority. Figure 7 shows the network configuration for the performance evaluation experiment to distinguish priority. In this experiment, the stream A assumed to be a general stream and the stream B assumed to be a priority stream are sent and throughput was measured. We compared the cases where the priorities are distinguished and not. Moreover, QoS function to distinguish packets priority by Open vSwitch is used in the experiment.

Table 2 Parameter of experiment to distinguish priority

Parameter	*Value*
Bitrate of stream A	1 Mbps
Bitrate of stream B	600 kbps
Bandwidth limit	1 Mbps
Guaranteed bandwidth	500 kbps
Protocol	UDP
Number of trails	3
Measurement time	180 s

The experiment scenario is as follows and Table 2 shows detailed parameters of the experiment. This experiment is for comparing the throughput in case of sending the priority traffic in the situation where the general stream runs out of bandwidth.

1 the stream A is sent

2 15 seconds after the stream A is sent, the stream B is sent.

Figure 8 and 9 show the effective throughput summarised which are calculated from the average transferred bytes per second. Moreover, these graphs also show the average used bandwidth and the standard deviation of each stream after the stream B is sent.

Figure 8 Effective throughput without priority control (see online version for colours)

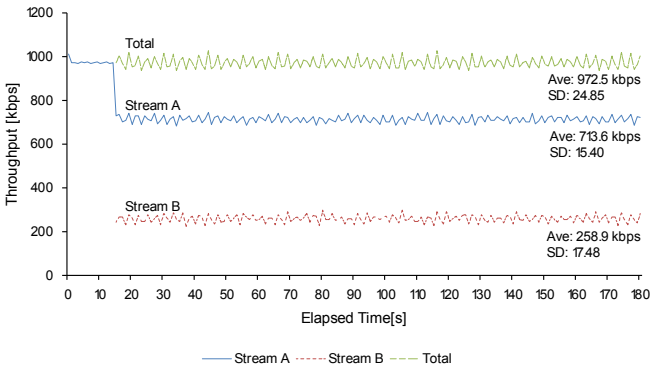


Figure 9 Effective throughput with priority control (see online version for colours)

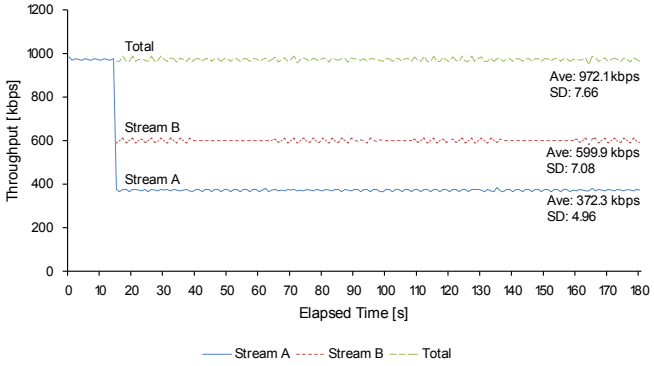
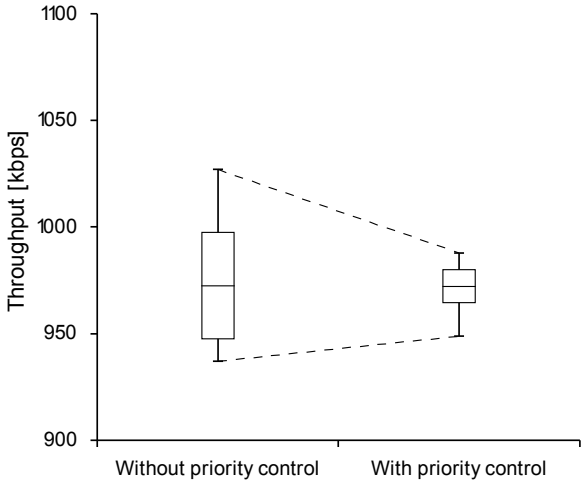


Figure 8 shows that the throughput of the stream A was higher than the throughput of the stream B after sending the stream B that assumed be a high priority stream since the priorities were not distinguished. On the other hand, Figure 9 shows that the throughput of the stream B was higher than the throughput of the stream A. This shows that the proposal system determines the priority of the streams by a pair of the destination IP address and the destination port number and can perform transfer controlling. Moreover, the standard deviations of each stream is lower than experiment without distinguishing priority of streams. In particular, although the throughput of the stream A was decreased, a scatter of throughput was also reduced.

Figure 10 shows the average, standard deviation, maximum value and minimum value transition of total throughput by experiment to distinguish priority. We can see that the standard deviation of total throughput is low when the priority is distinguished.

From the results, the transfer performance of high priority stream was improved by distinguishing the priorities and performing transfer controlling. Also, although the transfer performance of the general stream decreased, the standard deviation decreased. In a condition which the network traffic is locally and temporarily increased, the transfer controlling according to the priority seems to be effective.

Figure 10 Average, standard deviation, max and min value transition of total throughput by experiment to distinguish priority



6.2 *Experiment to increase the number of cooperative devices*

Figure 11 shows the network configuration for experiment to increase the number of cooperative devices. In this experiment, to verify a relationship between the number of cooperative devices and the throughput, we examined a measurement test of the throughput when cooperative device is increased. In the experiment, we sent out three streams of A, B and C. In addition, we increased the number of cooperative devices from one to three. Also, we increased the number of access points. The experiment scenario is as follows and Table 3 shows detailed parameters of the experiment.

1 the stream A is sent

2 15 seconds after the stream A is sent, the stream B is sent

3 30 seconds after the stream A is sent, the stream C is sent.

Table 3 Parameter of experiment to increase the number of cooperative devices

Parameter	*Value*
Number of devices	1~3
Number of streams	3
Bitrate	100 Mbps
Protocol	UDP
Number of trials	3
Measurement time	180 s

Figure 11 Overview of the testbed network configuration for experiment to increase the number of cooperative devices

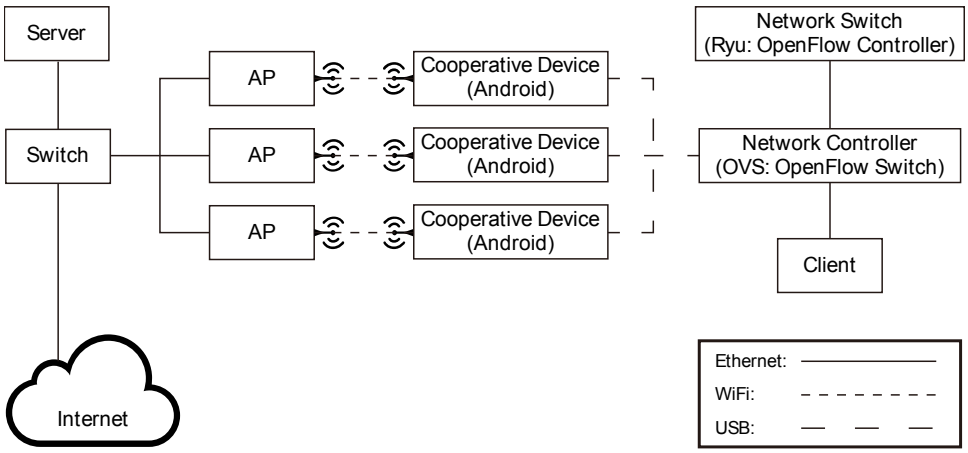


Figure 12, 13 and 14 show the used bandwidth summarised as stacked graphs which are calculated from the average transferred bytes per second. Moreover, these graphs also show the average used bandwidth and the standard deviation of each stream after the stream C is sent.

Figure 12 Effective throughput of each stream with one cooperative device (see online version for colours)

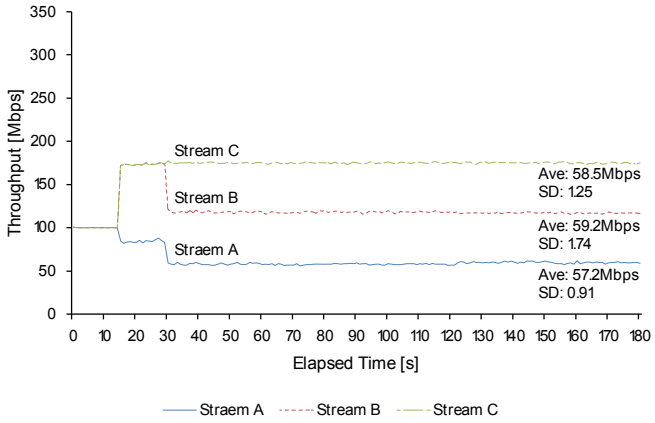


Figure 13 Effective throughput of each stream with two cooperative device (see online version for colours)

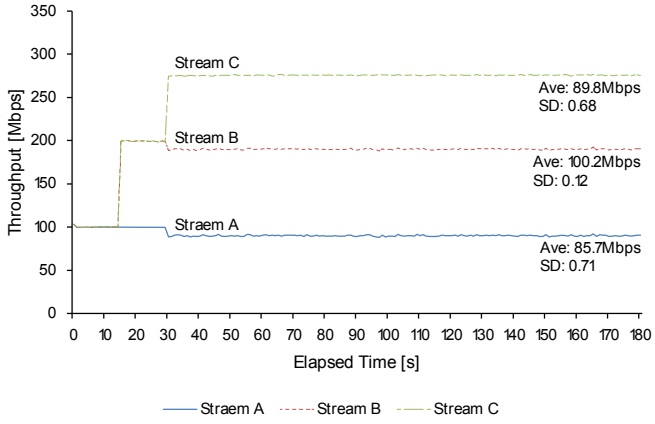


Figure 12 shows that the proposed system cannot handle network traffic since network resources that cannot handle by one cooperative device are requested. Figure 13 shows that the proposed system handles traffic well better than in the case of one cooperative device since the number of devices increased to two. However, cooperative device that is sending two streams cannot handle all traffic. Figure 14 shows that the proposed system can handle all traffic since the number of devices increased to three. In particular, in Figure 14, we can see that the performance of each stream was very stable.

Figure 14 Effective throughput of each stream with three cooperative device (see online version for colours)

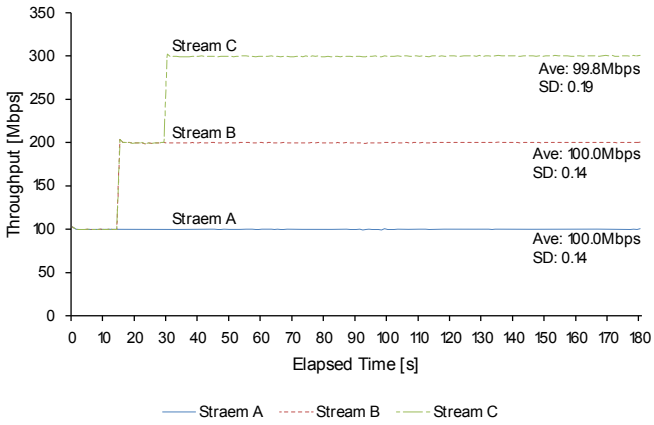


Table 4 Results of the experiment to increase the number of cooperative devices

Number of devices	*Effective throughput*	*Standard deviation*
1	174.9 Mbps	0.75
2	275.7 Mbps	0.44
3	299.9 Mbps	0.35

Table 4 shows the experimental results which are calculated from the average transferred bytes after the stream C is sent. The results show that our proposed system successfully aggregates network resources of cooperative devices, and improved the data transfer performance. In the case of using two cooperative devices, the proposed system increases the used bandwidth by about 58% since the case of one cooperative device. In addition, in the case of using three cooperative devices, the proposed system increases the used

bandwidth by about 71% since the case of one cooperative device. Moreover, in the case of using three cooperative devices, the used bandwidth is about 300 Mbps, which is all of the traffic sent out. From this results, our proposed system aggregates the network resource of several cooperative devices, it is possible to handle much traffic that cannot be handled by one cooperative device. Furthermore, the proposed system lowered the standard deviation. As a result, the proposed system improves the data transfer performance as well as it is effective in real time communication by increasing the available bandwidth.

Figure 15 Average total throughput by experiment to increase the number of cooperative devices

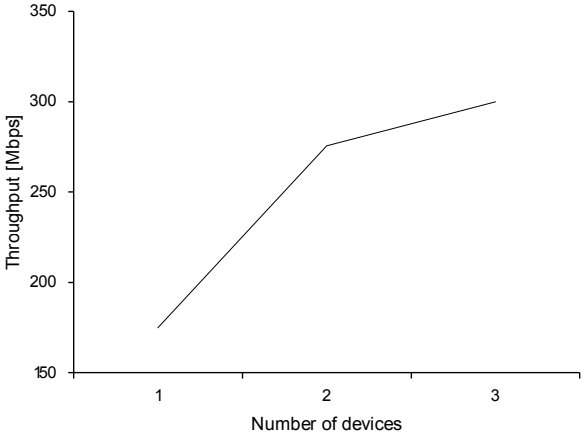
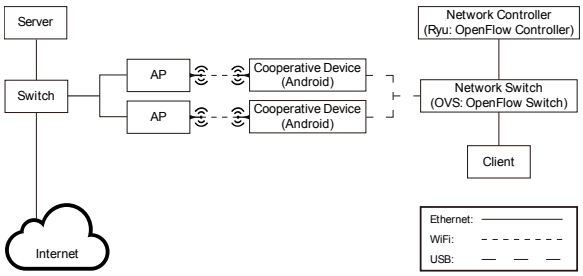


Figure 15 shows the average total throughput for each cooperative device. As mentioned above, as the number of cooperative devices increased, the average of the throughput increased. From the results, by increasing the number of cooperative devices, stable communication with higher throughput can be provided.

6.3 *Experiment for disconnection of cooperative devices*

Furthermore, we performed the experiment for disconnection of cooperative devices. Figure 16 shows the network configuration. The purpose of the experiment is to confirm that the proposed system can handle disconnection of the device and can perform scalable network resource management. The experiment scenario is as follows and Table 5 shows detailed parameters of the experiment.

Figure 16 Overview of the experimental network configuration for experiment of disconnection of cooperative devices



1 two cooperative devices are connected to the network switch

2 two streams are sent via each cooperative device

3 30 seconds after the streams are sent, one of cooperative devices is disconnected.

Table 5 Experimental parameters for disconnection of cooperative devices

Parameter	*Value*
Number of streams	2
Bitrate	10 Mbps
Protocol	UDP
Number of trails	3
Measurement time	60 s

Figure 17 Effective throughput when cooperative device is disconnected (see online version for colours)

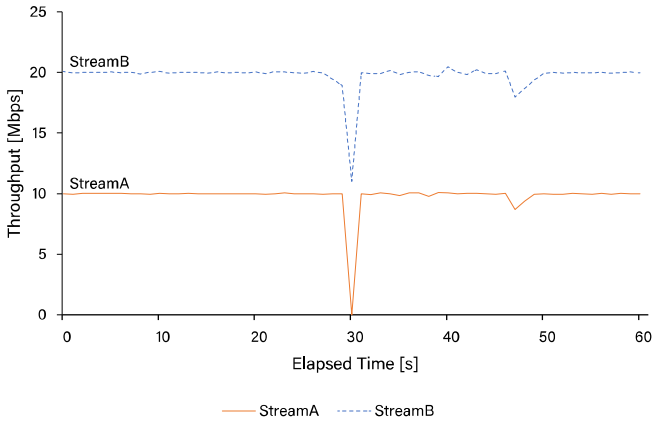


Figure 17 shows the used bandwidth summarised as a stacked graph which is calculated from the average transferred bytes per second. When the one of cooperative devices is disconnected, the throughput temporarily decreased. However, since the throughput recovers quickly, we observed the proposed system can dynamically handle disconnection of cooperative devices and can forward packets. Since the prototype system manages cooperative devices every second, it is assumed that packet forwarding could be recovered immediately.

7 Conclusions

In this paper, we proposed the dynamic resource adaptation method that increases the available bandwidth and performs transfer controlling according to the priority. The proposed method makes it possible to dynamically aggregate network resources of smart devices called cooperative device and handle traffic that the locally and temporarily increases. In addition, the proposed system determined the priority by a pair of the destination IP address and the destination port number, and performed the transfer controlling. This seems to be able to handle the network traffic that locally and

temporarily increases. Moreover, we developed a prototype system for evaluation experiment using the general purpose equipment. In the developed prototype system, the network switch and the cooperative devices are controlled by a single equipment. This seems to be able to easily change the network configuration even if the number of cooperative devices increased in order to increase the network resources. In the evaluation experiments using the prototype system, our proposed system improved the data transfer performance of the high priority stream by performing the transfer controlling according to the priority. Although the data transfer performance of the general stream became small, the standard deviation was low. Furthermore, our proposed system increases the available bandwidth as the number of cooperative devices increase. This makes it possible to handle much traffic that cannot be handled by one cooperative device by aggregating the network resources of several cooperative devices. Moreover, our proposed system is effective in real time communication such as streaming services since the proposed system makes the standard deviation was low. Therefore we consider the proposed system is effective for live streaming services such as YouTube Live and Twitch, and VoIP services such as Skype and Google Hangouts.

However, in the evaluation experiments, the number of cooperative devices is only three, which is very small scale experiments. Although we showed that the available bandwidth can be increased in the evaluation experiments by increasing the number of cooperative devices to three, it is necessary to evaluate the traffic volume that can be handled in the case of more increase. We assume from Figure 15 that the available bandwidth increase even if the number of cooperative devices is set to 4 or more, however, there is a limit because the available bandwidth does not increase linearly. Moreover, there is a limit to the number of APs that can deploy without overlapping channels even though prototype system is configured using 5GHz WiFi. In the future, we will perform evaluation experiments on a large scale which increased the number of cooperative devices. We will also perform experiments in a more dynamic environment, considering disconnection of cooperative devices. Furthermore, in this evaluation experiments, we did not perform an aggregation of network resources of cooperative devices and transfer controlling according to the priority at the same time, but only an individual evaluation experiment was performed. In the future, we will evaluate the proposed algorithm that simultaneously with aggregating network resources of cooperative devices and transfer controlling according to the priority. Additionally, we are considering adaptation to cellular networks. It seems to be necessary in real environments is assumed adaptation to cellular networks. In that case, it is assumed that the amount of network resources of each network edge is different. The current algorithm is based on the number of streams, however network edge should be selected with a weight on network resources. To that end, we plan to improve the algorithm to be more effective aggregating network resources in cellular networks.

References

3GPP (2018) [online] http://www.3gpp.org/ (accessed 12 January 2018).

Celenlioglu, M.R. and Mantar, H.A. (2015) 'An SDN based intra-domain routing and resource management model, in *2015 IEEE International Conference on Cloud Engineering*, pp.347–352.

Cisco Visual Networking Index (2017) *Global Mobile Data Traffic Forecast Update*, 2016–2021 White Paper [online] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobilewhite-paper-c11-520862.html (accessed 12 January 2018).

Huang, C., Nakasan, C. and Ichikawa, K. (2015) 'A multipath controller for accelerating GridFTP transfer over SDN', *IEEE 11th International Conference on e-Science*, pp.439–447.

iPerf (2018) *The TCP, UDP and SCTP network bandwidth measurement tool* [online] https://iperf.fr/ (accessed 12 January 2018).

KDDI (2016) [online] http://www.kiai.gr.jp/jigyou/h28/PDF/1220p7.pdf (accessed 12 January 2018).

NTT Docomo (2011) [online] http://www.soumu.go.jp/main_content/000117676.pdf (accessed 12 January 2018).

OkHttp (2018) [online] http://square.github.io/okhttp/ (accessed 12 January 2018).

Open Networking Foundation (2018) Open Networking Foundation is an operator led consortium leveraging SDN, NFV and cloud technologies to transform operator networks and business models [online] https://www.opennetworking.org/ (accessed 12 January 2018).

Open vSwitch (2018) [online] http://openvswitch.org/ (accessed 12 January 2018).

ReactiveX (2018) [online] http://reactivex.io/ (accessed 12 January 2018).

Retrofit (2018) http://square.github.io/retrofit/ (accessed 12 January 2018).

Ryu (2018) *SDN Framework* [online] https://osrg.github.io/ryu/ (accessed 12 January 2018).

Sato, G., Hashimoto, K., Uchida, N. and Shibata, Y. (2013) 'Network link selection method for disaster oriented mobile network based on OpenFlow framework', *Proc. of Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, *2013 Seventh International Conference*, pp.326–330.

Sinatra (2018) [online] http://www.sinatrarb.com/ (accessed 12 January 2018).

Systemu (2018) *RubyGems.org your Community Gem Host* [online] https://rubygems.org/gems/systemu/ (accessed 12 January 2018).