Regular paper | Journal of the KIECS. pp. 679-686, vol. 15, no. 4, Aug. 31. 2020, t. 102, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2020.15.4.679

# SDN 환경에서의 데이터 생성 형태와 서버 응답시간을 고려한 효율적인 부하분산 기법

김종건* · 권태욱**

# Efficient Load Balancing Technique Considering Data Generation Form and Server Response Time in SDN

Jong-Geon Kim* · Tae-Wook Kwon**

요 약

전 세계 데이터 총량이 2025년이면 175 ZB로 늘어날 것으로 전망되면서[1] 네트워크 영역에서의 데이터 처리 능력은 더욱 중요해지고 있다. 특히 데이터센터는 데이터 사용량이 늘어남에 따라 고집적화되고 있어서 유입되는 데이터로부터 유발되는 부하는 비용 절감과 효율적인 운영 등을 위해 적절히 분산되어야 한다. 기존 네트워크 체계의 한계점을 극복하기 등장한 SDN 기술은 네트워크 장비에서 H/W와 S/W를 분리하여 Legacy 체계의 경직성을 해소하는데, S/W 기반의 유연성을 이용해서 데이터센터에서의 부하분산에 효과적으로 적용할 수 있다. 본 논문에서는 SDN 기술을 활용하여 사용자로부터 받은 데이터를 유형별로 분류하고, 분류된 데이터를 데이터센터내 서버의 응답시간 순서대로 전송·처리함으로써 효율을 높일 수 있는 방법을 제안한다.

ABSTRACT

With global data totals expected to grow to 175 ZB by 2025, data processing capabilities in the network area are becoming more important. In particular, data centers are becoming more stubborn as data usage increases, and the load generated by incoming data should be appropriately distributed to reduce costs and efficiently operate. The SDN technology, which emerged to overcome the limitations of the existing network system, removes rigidity of the Legacy system by separating H/W and S/W from the network equipment, and can be effectively applied to load balancing in the data center using S/W-based flexibility. In this paper, we propose ways to increase efficiency by classifying data received from users by type by utilizing SDN technology, and transmitting and processing classified data in order of response speed of servers in the data center.

키워드

SDN(: Software Defined Networking), Load Balancing, Data Center, Server Response Time
소프트웨어 정의 네트워크, 부하 분산, 데이터 센터, 서버 응답 속도

* 국방대학교 컴퓨터공학과(201926001@kndu.ac.kr)
** 교신저자 : 국방대학교 컴퓨터공학과
· 접 수 일 : 2020. 06. 15
· 수정완료일 : 2020. 07. 15
· 게재확정일 : 2020. 08. 15
· Received : Jun. 15, 2020, Revised : Jul. 15, 2020, Accepted : Aug. 15, 2020
· Corresponding Author : Tae-wook Kwon
Dept. of Computer Engineering, Korea National Defense University,
Email : kwontw9042@mnd.go.kr

## Ⅰ. 서 론

글로벌 시장 조사기관 IDC는 전 세계 데이터 총량이 2018년 33 ZB(: zettabytes)에서 2025년 175 ZB로 늘어날 것으로 전망했다[1]. 이처럼 5G와 같은 광대역 통신망의 확산과 사물인터넷의 등장으로 확산되는 스마트 디바이스의 일상화[2][3] 등의 이유로 데이터 사용량은 더욱 늘어날 것으로 전망됨에 따라 네트워크 영역은 더욱 높은 효율의 데이터 처리와 부하분산 능력이 필요하게 되었다. 그러나 특정 벤더에 의해 H/W와 S/W가 결합되어 제공되는 기존의 네트워크 장비로 인해 유발되는 경직성은 증대되는 데이터 처리에 제한점을 발생시켰다. 네트워크 관리자는 특정 벤더 중심이 아니라, 사용자 중심적이고, 더욱 유연하고, 효율적인 네트워크 기능을 요구하게 되었으며, 이런 변화요구를 수용하고, Legacy 체계의 한계점을 극복하기 위해 SDN(: Software Defined Network)이 등장하게 되었다.

SDN은 H/W와 S/W 기능이 분리되어 여러 네트워크 장비에 대해 동시에 일관되고 유연성 있는 정책을 구현할 수 있게 함으로써, 4차 산업혁명의 데이터 사용 요구를 충족시키는 데이터 처리와 부하분산에 적합하여 많은 연구가 이루어지고 있다.

데이터센터는 4차 산업혁명 시대의 관련 산업 활성화를 위한 데이터의 처리, 유통 및 저장을 수행하는 핵심 인프라로써 중요성이 증대되고 있다. 구글, 아마존 등은 전 세계에 데이터센터를 활발하게 구축하고 있으며, 국내에서는 상업용으로 2019년까지 158개가 지어졌으며, 2025년까지 32개가 신규로 구축될 예정으로 연평균 15.9% 성장할 것으로 전망된다[4]. 데이터센터의 중요성이 부각되면서 효율적인 데이터 처리와 부하분산은 전력사용량 감소, 비용 절감, 안정적인 운영 등의 측면에서 매우 중요한 부분으로써 주목받고 있다. 근래의 데이터센터는 대규모의 서버 클러스터를 가상화하여 운영하기 때문에 가상화에 바탕을 둔 SDN을 활용한 부하분산이 데이터센터 전체 효율의 높이는 데 큰 역할을 할 것으로 기대된다.

본 논문에서는 데이터센터로 일반적인 데이터가 유입되었을 때 서버 클러스터에 속한 각 서버의 응답속도가 빠른 순서대로 데이터를 전송하고, 동일 유형의 데이터가 폭증하여 유입되는 상황이 발생하면, 해당 데이터를 분류한 뒤 서버 클러스터 중 가장 응답시간이 빠른 서버로 전송하는 기법을 제안한다. 이 기법은 기존, 데이터의 생성 형태만 고려한 부하분산 기법[5]에서 데이터를 수신하는 서버의 응답속도가 고려되었기 때문에 더 높은 효율을 기대할 수 있다.

## Ⅱ. 관련연구

### 2.1 SDN(: Software Defined Networking)

SDN은 네트워크 장비에서 H/W와 S/W 기능을 분리하는 것이 핵심이다. H/W 영역은 Data Plane으로써 데이터를 전송하는 역할을 한다. S/W 영역은 Control Plane과 Application Plane으로 나뉘고, Control Plane은 네트워크에 대한 정책이 적용되는 영역으로 볼 수 있으며, Application Plane은 정책 적용을 위한 사용자 지원 영역이다. Control Plane은 SDN Controller로 구현되며, 논리적으로 중앙 집중화하여 프로그래밍 가능하도록 고안되었고, 이를 통해 네트워크의 제어나 흐름, 혼잡 제어 등에 유연성을 제공한다. 그림 1은 래거시 체계에 SDN을 적용했을 경우를 비교하여 설명한 것이다.

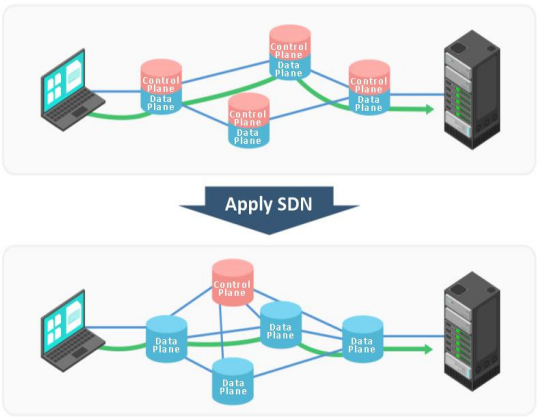


그림 1. 기존 네트워크와 SDN 비교[6]
Fig. 1 The comparison between legacy network and SDN

기존 체계에서 각각의 네트워크 장비별로 이루어지던 Control Plane의 역할이 SDN에서는 하나의 SDN Controller가 수행함으로써 전체적인 트래픽을 제어할

수 있게 되어 네트워크 영역에서의 유연성을 발휘할 수 있다. 이러한 S/W 기반의 SDN 특징을 트래픽 부하분산에 적용하는 많은 연구가 진행되고 있다.

D. Kim은 기계 학습 기법을 사용하여 Flow를 분류한 후, 전송 지연을 측정하여 효율적인 전송경로를 선택함으로써 부하를 분산하는 기법을 제안하였다[7]. J. Son은 서비스별로 우선순위를 정하고 서비스에 따라 패킷을 전송하여 네트워크 자원의 효율적인 사용과 패킷 전송 시 생기는 지연을 최소화하는 방안을 제안하였다[8]. K. Soleimanzadeh는 스위치 포트와 서버 응답시간을 기반으로 트래픽에 대한 정보를 정기적으로 수집하는 알고리즘을 제안하였다[9]. 위의 많은 연구에도 불구하고, 데이터센터로 데이터가 유입되는 측면과, 유입된 데이터를 서버 클러스터로 처리하는 측면, 두가지를 모두 고려하지 못한 한계점이 있다.

### 2.2 데이터센터와 서버 클러스터

데이터센터는 조직의 방대한 정보저장을 위한 서버, 네트워크 회선 등을 제공하여 데이터를 안정적으로 통합·관리하는 인프라 시설이다. 가상화 기술 발전과 클라우드 등의 서비스 제공과 함께 빠르게 발달중이다. 데이터센터는 웹을 기반으로 하는 다양한 서비스 제공을 위해 부하분산 목적의 서버 클러스터를 구축하고 있으며, 데이터 사용량이 늘어날수록 서버는 고집적화되고 있다. 고용량의 서버가 집적되어 있는 데이터센터에서 부하분산은 매우 중요한 부분이며, SDN을 기반으로 S/W를 이용해서 서버 클러스터를 제어하고 활용하는 것은 데이터센터 효율을 높이는데 필수적이다.

### 2.3 데이터 생성 형태를 고려한 부하분산

J. Yoon은 Client가 특정 이슈에 영향을 받아 편중된 데이터를 요구할 때, Middle Box를 활용하여 데이터의 유형을 분류하고, 그중 폭증하는 데이터를 우선처리함으로써 부하를 분산하는 개념을 제시하였고[5] 그림 2와 같이 도식화 할 수 있다.

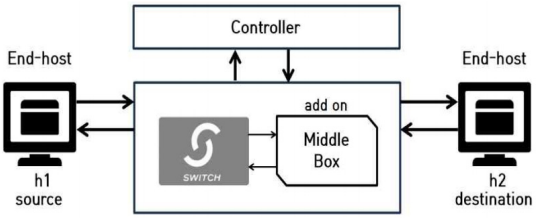


그림 2. [5]'s System design
Fig. 2 [5]의 시스템 설계

Middle Box를 통한 폭증 데이터 분류 세부 정책을 SDN Controller를 통해서 사용자가 제어함으로써 SDN 환경에서 그 효과가 극대화되도록 하였다. 이 기법은 외부로부터 여러 사용자의 요청을 처리하는 데이터센터에 적용하면 부하를 분산시키는 데 효과적이다.

### 2.4 서버 응답시간 기반의 부하분산

H. Zhong은 Control Plane과 Data Plane이 분리되어 있는 SDN 특성을 활용하여 Control Plane으로 서버의 응답시간을 측정하고 측정된 응답시간을 사용자 요청 처리 과정에서 서버를 선택하는데 반영함으로써 부하를 분산하였다[10]. 이 방법은 SDN의 장점을 활용하여 부하를 효과적으로 분산시켰으나, 데이터센터로 들어오는 사용자 요청의 특성을 고려하지 않고 서버의 상태만 고려한 한계점이 있다.

## III. 제안사항

### 3.1 제안하는 기법의 동작방법

제안하는 기법의 동작은 ① Middle Box에서 패킷을 모니터링해서 분류하는 동작과 ② Controller에서 각 서버의 응답시간을 측정하여 각 사용자 요청을 처리할 서버를 선정하고 ③ 스위치에서 선정된 서버에 따라서 패킷을 포워딩하는 동작, 세 가지로 나뉜다. 전제적인 시스템 설계는 그림 3과 같으며, 그림 4와 같은 알고리즘으로 작동한다.

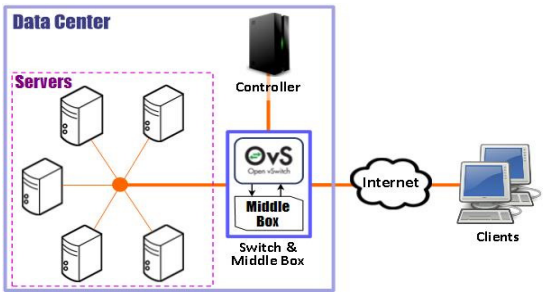


그림 3. System design
Fig. 3 시스템 설계

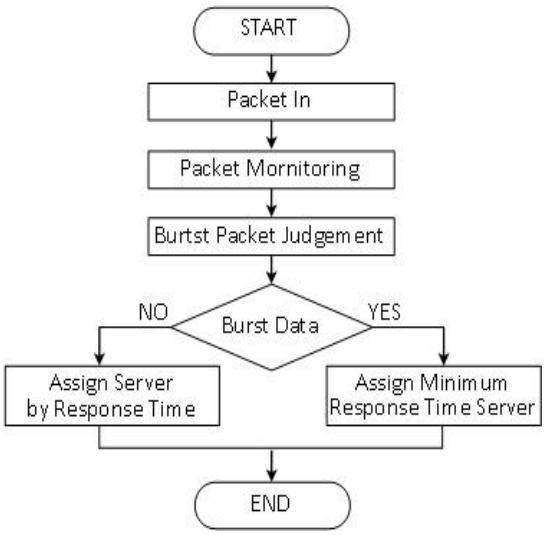


그림 4. 제안하는 기법의 알고리즘 순서도
Fig. 4 Algorithm flowchart of the proposed technique

### 3.1.1 Middle Box의 동작

일반적인 데이터는 도착 순서대로 Middle Box의 일반큐로 저장되다가, 사용자의 정책-Controller에 입력된-에 해당하는 특정 형태의 데이터가 폭증하는 상황이 발생하면, 폭증하는 데이터는 우선처리큐로 저장된다. 이때 우선처리큐에 저장된 데이터에 우선순위를 부여하여 서버로 먼저 전송되고, 이런 절차는 특정 데이터가 폭증하는 상황이 종료될 것으로 예상되는 시점까지 지속된다.

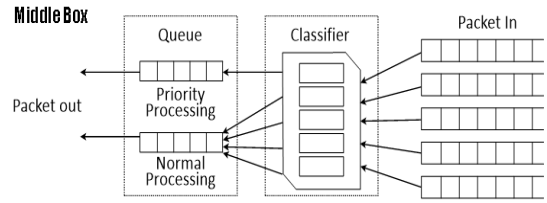


그림 5. 미들 박스의 동작
Fig. 5 Middle box processing

### 3.1.2 Controller의 동작

시스템이 시작되면 Controller는 각 서버로 응답시간 확인을 위한 Packet-out 메시지를 전송하고 전송시간 $T_{send}$를 기록한다. 스위치를 거쳐 Packet-out 메시지를 받은 서버는 해당 메시지를 폐기하고, 응답 메시지인 Packet-In 메시지를 Controller로 전송한다. 응답 메시지를 받은 Controller는 수신시간 $T_{arrival}$을 기록하고 $T_{response} = T_{arrival} - T_{send}$를 산출한다. 이 과정은 모든 서버를 대상으로 진행되며, 일정 주기를 반복한다. 산출된 서버별 $T_{response}$는 올림차순으로 정렬 후 그 결과를 Switch로 전송한다.

### 3.1.3 Switch의 동작

Switch는 Middle Box로부터 전달받은 분류된 데이터를 서버로 포워딩한다. 이때, ① 일반큐에 저장된 일반 데이터는 오름차순으로 정렬된 $T_{response}$-Controller로부터 전달받은-순서에 따라서 서버를 선정한 뒤 데이터를 포워딩하고, ② 데이터 폭증 상황이 발생하면 우선처리큐에 저장된 폭증 데이터는 $T_{response}$값이 가장 작은 서버에 할당하여 포워딩하고, 이 과정과 동시에 일반큐의 데이터는 우선처리큐를 담당하는 서버를 제외하고 ① 과정을 진행한다.

## Ⅳ. 실험 결과

### 4.1 시스템 구현

폭증하는 데이터를 Client에서 서버로 전달할 때 부하분산 기법의 효과를 확인하기 위해 패킷 생성기 역할을 하는 Client와 데이터를 분류하는 Middle Box, 서버 클러스터 내의 서버 응답속도를 측정하고 그 결괏값을 Switch로 전달하는 Controller, Client가 보낸

데이터를 수신하고, 응답을 재전송하는 Server를 각각 파이썬 스크립트로 구현하였으며, 미니넷 환경에서 구동한다.

### 4.2 실험 환경

제안하는 부하분산 기법을 가상 네트워크 환경에서 구현하여 효과를 평가하였다. 실험 환경은 CPU Intel (R) Core(TM) i7-3770 @ 3.40GHz, OS Windows 10, RAM 16GB, Virtual Box, Ubuntu 14.04.4 가상머신 환경에서 미니넷 2.2.2 에뮬레이터와 floodlight 컨트롤러를 사용하였다.

### 4.3 실험 방법

실험은 Fig. 3의 시스템 설계에 따른다. Client는 패킷을 생성하여 가상의 데이터센터로 전송하고, 전송된 데이터가 Middle Box에서 분류된 후 Controller에서 산출된 서버 우선순위에 따라서 서버로 전달되며, 전달된 패킷이 다시 Client까지 돌아오는 Latency를 측정한다.

제안한 기법의 효과를 확인하기 위해서 데이터 생성 형태를 고려하여 부하를 분산한 [5]의 기법을 대조군으로 설정하여 실험을 진행한다. 먼저 [5]의 기법으로 분류된 패킷을 서버로 포워딩하는데, 이때 패킷을 수신해서 처리할 서버를 Round Robin 방식으로 선정한 뒤 포워딩하여 Latency를 측정하고, 제안한 기법을 적용한 결과와 비교한다.

서버 클러스터 내의 각 서버의 성능치를 달리하여 실험을 진행한다. 데이터센터로 들어오는 Client 요청은 서버에 할당되어 처리되기 때문에 실시간으로 모든 서버의 부하는 달라지기 때문이다. 총 5대의 서버 중 첫 번째 서버의 성능치를 기준으로 두 번째 서버부터 5%씩 증가하면서 임의의 부하를 부여함으로써 서버에 각기 다른 부하가 걸리는 상황을 구현한다.

패킷은 특정 상황에서 동일 유형의 데이터가 폭증하는 상황을 가정한 군집형태의 데이터를 전송한다. 데이터는 총 5가지 유형의 패킷으로 이루어져 있으며, 패킷 중 특정 유형의 패킷이 폭증하도록 무작위로 생성되어 전송된다.

Client로부터 생성되어 전송되는 패킷의 수는 각 실험별로 100,000개, 500,000개, 1,000,000개로 설정한다. 이것은 서버에서 처리하는 데이터의 양을 달리하여 Latency를 측정하고 비교 분석함으로써 제안 기법의 효과를 확인하기 위함이다.

### 4.4 전송된 데이터 지연시간 측정결과

Fig. 6은 첫 번째 실험에 대한 결과로써, 100,000개의 패킷을 생성하여 전송한 결과이다. [5]의 기법으로 측정결과, Latency가 최저 0.6900초, 최대 2.0500초였으며, 제안한 기법으로 측정결과, Latency의 최저는 0.6200초, 최대는 1.5939초로 측정되었다. 평균 Latency는 [5]의 기법은 1.0901초, 제안한 기법은 1.1499초가 측정되었다. 100,000개 패킷에 대해 제안한 기법은 [5]의 기법과 유의미한 차이를 확인할 수 없었다.

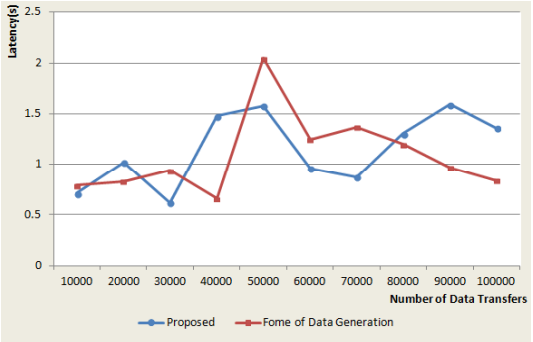


그림 6. 패킷 100,000개 전송시 지연시간
Fig. 6 Latency to send 100,000 packets

Fig. 7은 두 번째 실험에 대한 결과이며, 500,000개의 패킷을 전송한 결과이다. 각 기법의 최소 및 최대 Latency를 보면, [5] 기법은 3.3540 / 4.4850초, 제안한 기법은 2.6700 / 4.4040초로 측정되었으며, Latency의 평균은 [5] 기법 4.0034초, 제안 기법 3.1501초로써 27.1% Latency 감소효과가 있었다.

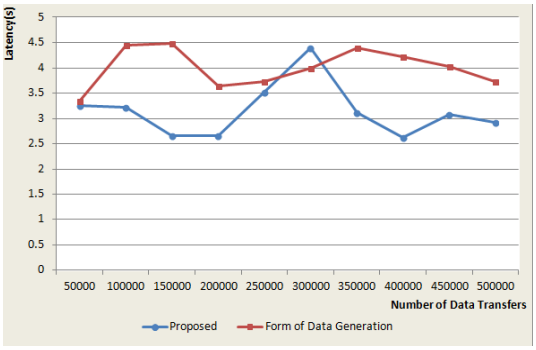


그림 7. 패킷 500,000개 전송시 지연시간
Fig. 7 Latency to send 500,000 packets

Fig. 8은 마지막 실험에 대한 결과로써, 1,000,000개의 패킷을 전송한 결과이다. Latency의 [5]기법의 최소 및 최댓값은 5.4201 / 6.6100초, 제안기법은 4.0890 / 5.5540초로 측정되었다. 평균값은 [5]기법 6.0071초, 제안 기법 4.4894초로 33.8%의 Latency 감소효과가 확인되었다.

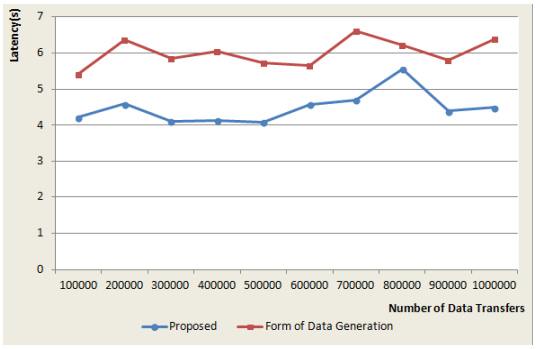


그림 8. 패킷 1,000,000개 전송시 지연시간
Fig. 8 Latency to send 1,000,000 packets

패킷의 수가 증가할수록 지연시간 감소효과가 증가하는 것은 제안한 기법을 적용했을 경우, 단위 패킷에서는 매우 작은 효과만을 보이기 때문인데, 이것은 서버의 성능치를 각 서버별로 5%의 차이만 주었기 때문이다.

전체 실험 결과를 정리하면 Table 1과 같다.

표 1. 지연시간 측정결과
Table 1. Latency measurement results

| Load Balancing Method | avg. Latency of data(s) | | |
|---|---|---|---|
| | 100,000 Packets | 500,000 Packets | 1,000,000 Packets |
| Proposed | 1.14990299 | 3.15010600 | 4.48940697 |
| Form of Data Transfers | 1.09010301 | 4.00340803 | 6.00710902 |
| Latency reduction | 0.05979998 | -0.85330203 | -1.51770205 |
| | -5.2% | 27.1% | 33.8% |

## V. 결론 및 향후 연구

본 논문에서는 SDN 환경에서, 데이터센터로 유입되는 데이터를 생성 형태에 따라 분류하고, 응답속도가 빠른 서버가 분류된 데이터를 처리하도록 함으로써 부하를 분산하는 방안을 제안하였다. 실험 결과와 같이 특정 유형의 데이터가 폭증하는 상황에서 서버 클러스터에 속해있는 서버들의 응답속도를 고려하여 Client의 요청을 처리했을 경우, [5]에서 제안한 방법보다 부하 분산에 효과가 있음을 확인할 수 있었다.

제안한 기법은 데이터센터로 유입되는 데이터를 분류하는 것과 유입된 데이터를 서버로 할당하는 것, 두 가지 측면을 모두 고려했기 실험에서 나타난 바와 같이 방대한 양의 데이터를 처리하는 데이터센터에서 효율을 높이는데 기여할 것으로 기대된다. 특히 본 논문의 내용은 SDN 환경에서 구현되어 네트워크 관리자가 특정 상황에서 선택할 수 있는 정책 중 하나의 옵션이 될 수 있다. 따라서 네트워크의 트래픽 관리에 매우 유연하게 대처할 수 있다.

향후 연구로, 현재 SDN이 적용되어 있는 군 데이터센터를 에뮬레이터 OPNET을 이용해 구현하고, 본 논문에서 제안한 방법을 적용시 효과를 확인할 예정이다. 특히 데이터센터 운영비용의 대부분을 차지하는 전력사용량을 산출하여 효과를 확인하고자 한다.

감사의 글

※ 위 논문은 "2020년 봄철학술대회 우수논문"입니다.

## References

[1] M. Kim, "AI, the database of the big data era: graph data", https://www.samsungsds.com/global/ko/support/insights/1208783_2284.html (accessed June 29, 2020.)

[2] K. Park, "IoT Based Office Environment Improvement Plan," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 15, no. 1, Feb. 2020, pp. 61-70.

[3] S. Lee, "Design and Application of LoRa-based Network Protocol in IoT Networks," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 14, no. 6, Dec. 2019, pp. 1089-1096.

[4] S. Sin, "Data Center Market and Major Enterprise Trends", *GBSA*, no. 2020-03, Mar. 2020

[5] J. Yoon and T. Kwon, "An Efficient Load Balancing Technique Considering Forms of Data Generation in SDNs," *J. Korea Multimedia Society*, vol. 23, no. 2, Feb. 2020, pp. 247-254.

[6] S. Sin, "SDN Changes the Network World", https://www.sktinsight.com/98995 (accessed June 29, 2020.)

[7] D. Kim, C. Hong, "Transmission Delay Estimation-based Forwarding Strategy for Load Distribution in Software-Defined Network," *J. of the Korean Institute of Information Scientists and Engineers*, vol. 23, no. 5, May. 2017, pp. 310-315.

[8] J. Son, C. Hong, "SDN-Based Packet-Forwarding and Delay Minimization Algorithm for Efficient Utilization of Network Resources and Delay Minimization," *J. of the Korean Institute of Information Scientists and Engineers,* vol. 21, no. 11, Nov. 2015, pp. 727-732.

[9] K. Soleimanzadeh, M. Ahmadi, M. Nassiri, "SD WLB: An SDN aided mechanism for web load balancing based on server statistics," *J. Electronics and Telecommunications Research Institute*, vol. 41, no. 2, Dec. 2018.

[10] H. Zhong, Y. Fang, J. Cui, "LBBSRT: An efficient SDN load balancing scheme based on server response time," *J. Future Generation Computer Systems*, vol. 80, Mar. 2018, pp. 409-416.

## 저자 소개

**김종건(Jong-Geon Kim)**

2009년 해군사관학교 전산학과 졸업(공학사)
2019년 ~ 현재 국방대학교 대학원 컴퓨터공학과 석사과정
※ 관심분야 : 네트워크, SDN

**권태욱(Tae-Wook Kwon)**

1986년 육군사관학교 컴퓨터공학과 졸업(공학사)
1995 미국 해군대학원 컴퓨터공학(공학석사)
2001년 연세대학교 컴퓨터공학과(공학박사)
2007년 ~ 현재 국방대학교 컴퓨터공학과 교수
※ 관심분야 : 네트워크, Sensor Networking, CCN, SDN, NFV