# 네트워크 부하에 따른 부분 오프로딩 효과 분석

백재석[1] · 남광우[2] · 장민석[2] · 이연식[2*]

[1](주)해라이트 · [2]군산대학교

# Analysis of partial offloading effects according to network load

Jae-Seok Baik[1] · Kwang-Woo Nam[2] · Min-Seok Jang[2] · Yon-Sik Lee[2*]

[1]Halight, Co., [2]Kunsan National University

E-mail : gwings@naver.com / {kwnam,msjang,yslee}@kunsan.ac.kr

## 요 약

본 논문은 FEC 환경에서 응용 서비스의 처리 지연시간 최소화를 위하여 선행연구 제안한 부분 오프로딩 시스템의 네트워크 부하에 따른 오프로딩의 효과를 분석한다. 모바일 장치와 FEC 서버 간의 2계층 협력 컴퓨팅 시스템으로 구성된 제안 시스템을 로컬 전용 및 에지 서버 전용 처리와 비교한다. 제안 시스템은 다중 분기구조의 재구성 선형화를 통한 부분 오프로딩 알고리즘[1]과 두 계층 간의 최적 협업 알고리즘[2]을 포함한다. 실험은 다중 분기구조의 DAG 토폴로지를 갖는 논리적 CNN 모델을 대상으로 계층 스케줄링을 적용하여 수행하였으며, 실험 결과 제안 시스템은 로컬이나 에지 전용 실행과 비교하여 항상 효율적인 작업 처리 전략 및 처리 지연시간을 제공함을 입증하였다.

## ABSTRACT

This paper proposes a partial offloading system for minimizing application service processing latency in an FEC (Fog/Edge Computing) environment, and it analyzes the offloading effect of the proposed system against local-only and edge-server-only processing based on network load. A partial offloading algorithm based on reconstruction linearization of multi-branch structures is included in the proposed system, as is an optimal collaboration algorithm between mobile devices and edge servers [1,2]. The experiment was conducted by applying layer scheduling to a logical CNN model with a DAG topology. When compared to local or edge-only executions, experimental results show that the proposed system always provides efficient task processing strategies and processing latency.

키워드

Fog/Edge computing, Partial offloading, DAG topology, Network load, Latency time

## Ⅰ. 서 론

본 논문은 부분 오프로딩을 위한 다중 분기구조의 분할 알고리즘과 모바일 장치와 에지 서버의 협업 알고리즘을 제안한 선행연구를 기반으로 네트워크의 부하에 따른 오프로딩의 효과를 분석하고자 한다. 분석을 위한 실험은 DAG 토폴로지를 갖는 논리적 CNN 모델을 대상으로 수행하고, 오프로딩 적용 시스템과 로컬(모바일 장치) 및 에지 전용 실행과 비교 분석한다.

* corresponding author

## Ⅱ. 부분 오프로딩 시스템 및 알고리즘

### 2.1 제안된 부분 오프로딩 시스템

로컬에서 요구한 응용 서비스에 대한 연산의 총 처리시간을 최소화하기 위한 부분 오프로딩 시스템은 로컬과 에지 서버 간의 2계층 협력 컴퓨팅 시스템으로 구성된다. 실행단계에서 로컬은 정책모듈을 통하여 자신의 채널 정보, 특성 및 요구 사항을 수집/저장하고, 파티션 생성 및 오프로딩을 수행하여 에지 서버를 통하여 작업을 완료한다. 에지 서버는 수신된 코드/데이터를 이용하여 로컬에 대한 나머지 작업을 처리하고 결과를 전송한다.

제안 시스템은 응용 서비스의 m개의 구현 모듈을 각각 파티션 대상으로 간주하여 DAG = {DAG1, D

AG2, · · ·, DAGm}으로 정의하고, 오프로딩 정책모듈에 의해 DAGi를 각각 로컬과 에지 서버에서 처리되는 DAGli와 DAGei로 구분하며, 임의의 모듈에 대한 '로컬 처리시간이 에지 서버 처리시간과 전송시간의 합보다 크면' 해당 모듈을 에지 서버로 오프로딩하여 처리한다. 응용 서비스 실행모듈의 실행흐름을 나타내는 DAG는 부분 오프로딩 결정을 위한 프로파일로써, 로컬에 저장되며 실행단계에서 DAG의 처리 방법을 알 수 있다. DAG의 각 정점(모듈)은 입력 모형(이전 모듈의 출력 모형)에 대한 연산을 수행하고, 다음 모듈로 해당 정점의 출력 모형을 전송한다.

2.2 부분 오프로딩 알고리즘

로컬과 에지 서버 간의 협업에 의한 작업의 총 처리 지연시간(T)은 다음 식과 같다.

$$T = T_l + T_e + T_{tt} = \sum_{i=1}^{m} (t_i^l + t_i^e) + \sum_{i=1}^{n} tt_i$$

$T_l$은 모든 DAGli 처리시간의 합이며, $T_e$는 모든 DAGei 처리시간의 합이고, $T_{tt}$는 오프로딩으로 발생하는 로컬과 에지 서버 간 출력 전송시간의 합이다.

로컬에서 수행되는 응용 서비스의 실행파일 토폴로지는 단순하지 않다[3]. 따라서 제안 시스템은 부분 오프로딩을 위하여, DAG 토폴로지의 각 정점의 다중 분기구조를 분석하여 협업 실행경로 생성을 위한 DAG으로 재구성하고[1], 재구성된 DAG으로부터 최단경로 탐색 알고리즘[2]을 이용하여 처리 지연시간 최소화를 유도한다.

제안된 부분 오프로딩 알고리즘은 단일 기지국 내에서만 최적의 오프로딩 방법을 제공하므로, 장치의 이동성으로 핸드오버가 발생하면 결정경로는 가장 빠른 경로라 할 수 없다[4]. 따라서 기지국 핸드오버 발생 시, 매 정점(모듈)의 처리시간이 기지국 사용자의 거주 시간보다 작은 범위에서 실행 위치가 결정되어야 한다[5,6]. 이는 정점별 최단경로 DAG를 도출함으로써 만족시킬 수 있다.

## III. 실험 및 평가

3.1 실험 환경

CNN은 여러 다중계층을 스케줄링하여, 한 컴퓨팅 서버에서 작업 일부를 처리하고 다른 서버에서 나머지 작업 실행이 가능하므로 부분 오프로딩 실험 모델로 사용이 적합하다[7]. 본 논문에서의 실험은 1) 9개의 인셉션 모듈 포함한 22층의 GoogLeNet의 DAG 토폴로지(GLND)의 논리적 모델과 2) shortcut을 제외하고 순차구조로 표현되는 50층의 Plain ResNet의 DAG 토폴로지(PRND)의 논리적 모델을 대상으로 수행한다. 로컬에 배치된 CNN은 먼저 여러 계층을 사용하여 작업을 처리하고 CNN의 숨겨진 계층의 출력인 작업의 중간 데이터를 에지 서버로 전송한다. 에지 서버에 배치된 CNN은 작업을 수신하면 해당 계층을 예약하고 로컬에 대한 나머지 작업을 처리한다.

로컬(모바일 장치)은 Intel core i5-6200U(2.3GHz)와 메모리 8GB인 노트북이며, 에지 서버는 CPU(3.80GHz 8core), GPU(Geforce GTX 1600) 및 32GB 메모리를 가진다. Access Point는 IEEE 802.11n draft 2.0 PoE AP(2.4/5GHz)를 사용한다. 실험은 단일 기지국 내에서 CSMA-CA 방식으로 통신하는 WiFi(100Mbps) 네트워크 조건에서 수행한다. 전송 편차가 적은 SINR 범위(-53 ~ -63dBm)에서 실험하고 signal은 차폐 및 조절기능을 이용하며, 전송시간은 Ixia wave test 장비로 측정한다.

에지 서버에 DAG 실행코드를 미리 저장한 상태에서, 네트워크 부하를 적용하지 않고 두 가지 실험 대상 코드를 각각 로컬과 에지 전용으로 실행한 결과, 에지 서버 대비 로컬의 실행시간이 GLND는 6.5배, PRND는 4.7배 소요되었다. 이와 같은 2계층 협력 컴퓨팅 시스템 환경에서 네트워크의 부하를 3단계로 구분하여 오프로딩의 효율성을 검증한다.

3.2 처리 지연시간 분석 및 평가

표 1과 표 2는 네트워크 부하가 각각 다른 환경에서, 두 가지의 실험 대상 DAG 토폴로지인 GLND와 PRND 각각에 대한 로컬 전용, 에지 서버 전용 및 제안 알고리즘을 적용한 총 처리 지연시간을 평가한 결과이다.

[표 1] Latency speed-up comparison for GLND topology

| N/W overload | low | medium | high |
|---|---|---|---|
| Local only | 1.00 | 1.00 | 1.00 |
| Edge only | 5.17 | 4.26 | 1.18 |
| PPOS | 5.83 | 5.10 | 1.67 |

[표 2] Latency speed-up comparison for PRND topology

| N/W overload | low | medium | high |
|---|---|---|---|
| Local only | 1.00 | 1.00 | 1.00 |
| Edge only | 4.45 | 3.82 | 1.45 |
| PPOS | 4.14 | 3.91 | 2.65 |

5G와 같은 경부하 네트워크 조건에서 에지 전용 실행 결과는 오프로딩 시간이 거의 소요되지 않으므로, 두가지 대상 모두 에지 서버에서 직접 실행과 비슷한 성능을 나타내었다. 제안 알고리즘 적용 시 데이터 및 처리결과 전송시간이 로컬에서의 처리시간보다 짧아서 에지 서버 전용 실행과 거의 비슷한 성능을 보였으나, PRND는 입출력에 대한 전송으로 인하여 성능이 약간 떨어짐을 보였다. 경부하 네트워크에서의 전체 처리시간은 로컬과 에지 서버의 컴퓨팅 성능에 따라 달라지므로, 에지 서버의 부하에

따라 모듈들의 처리 위치 조정이 필요하다.

Wi-Fi 네트워크의 부하를 상대적 중간 정도인 환경에서 측정한 결과, PPOS는 로컬 전용 대비 GLND는 5.1배, PRND는 3.91배 속도 향상 성능을 보였다. 이는 작업 부하가 적어지면 제안 알고리즘에 의한 부분 오프로딩의 효율성이 유의미하게 증가함을 나타낸다. 또한 제안 알고리즘은 DAG 토폴로지 및 체인 토폴로지에 대한 효율적 파티션 방법을 제공하며, 로컬 및 에지 전용 실행 대비 항상 성능이 우수함을 알 수 있다. 다만 모바일 장치의 배터리 성능이 제한적이라면 에지 서버에 더 많은 실행모듈을 오프로딩하는 방법이 더욱 효율적일 수 있다.

무선 네트워크 전송에서 네트워크의 작업 부하가 심할 경우, 제안 알고리즘은 로컬 전용 실행 대비 GLND는 1.67배, PRND는 2.65배 속도 향상 성능을 보였다. 이는 병목 현상 발생으로 입력이나 출력의 업로드 시간이 많이 소요되므로, 로컬에서 더 많은 처리를 하도록 함에 기인한다. 과부하 환경에서 순차 구조로 이루어진 PRND의 성능이 분기구조로 이루어진 GLND 대비 급격하게 나빠지지 않음은 실행 코드의 특성에 따른 오프로딩 대상 결정 및 그에 따른 전송시간의 차이에서 발생함을 알 수 있다. 물론 실행모듈들의 입력 또는 출력 데이터 크기가 작으면 전송시간을 줄일 수 있으며, 파티션된 실행모듈을 에지 서버로 더 많이 오프로딩할 수 있다. 만일 입력 데이터의 크기가 크고 네트워크의 작업 부하가 많을 때는 오프로딩의 효율성 확보가 어려움을 알 수 있다.

## IV. 결론

본 논문은 FEC 환경에서 응용 서비스 실행을 위한 처리 지연시간 최소화를 위한 기반연구로써, 서비스 구현 모듈에 대한 계산 오프로딩의 효용성을 분석하였다. 선행연구를 통하여 제안된 부분 오프로딩 알고리즘[1]과 로컬과 에지 서버의 최적 협업 알고리즘[2]을 포함하는 부분 오프로딩 시스템은 DAG 토폴로지와 체인 토폴로지 모두에 적합함을 알 수 있다. 또한 실험을 통하여 제안 시스템인 부분 오프로딩 기반의 모바일 장치와 FEC 서버 간의 2계층 협력 컴퓨팅 시스템은 네트워크의 부하에 따라 부분 오프로딩 대상의 처리 위치를 효과적으로 결정함으로써, 로컬 및 에지 전용 실행 대비 항상 유의미한 처리 속도 향상 성능과 효율적인 작업 처리 전략을 제공함을 알 수 있다. 따라서 부분 오프로딩 시스템은 모바일 장치에서의 응용 서비스 최적화를 위한 모델의 경량화 및 엣지 리소스 워크로드의 효율적 분배 관련 연구에 적용 가능하며, 로컬과 에지 서버 간의 효율적 데이터 전송을 위한 이동에이전트 적용 연구의 기반이 된다.

## Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1 04776 8) and a grant (21RITD-C161698-01) from Regional Innovation Technology Development Program funded by Ministry of Land, Infrastructure and Transport of Korean government.

## References

[1] J. Baek, M. Jang, Y. Lee, “Branch structure partitioning of DAG for partial offloading,” *in Proc. of KSCI 2022*, 30(2), pp. 621-624, 2022

[2] J. Baek, K. Nam, M. Jang, Y. Lee, “Extraction of collaborative execution path between local and edge server in an FEC environment,” *in Proc. of KSCI 2022*, 30(2), pp. 625-628, 2022

[3] X. Tian, J. Zhu, T. Xu and Y. Li, “Mobility-Included DNN Partition Offloading from Mobile Devices to Edge Clouds,” *Sensors 2021*, vol. 21(01), 229, 2021

[4] Y. Lee, K. Nam, M. Jang, “Extracting optimal moving patterns of edge devices for efficient resource placement in an FEC environment,” *Journal of the Korea Institute of Information and Communication Engineering*, 26(1), pp. 162-169, 2022

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, “A survey on mobile edge computing: The communication perspective,” *IEEE Communication Survey Tutorial*, vol. 19, no. 4, pp. 2322-2358, 2017

[6] Z. Ning, P. Dong, X. Kong, and F. Xia, “A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things,” *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4804-4814, 2019

[7] S. Yu, X. Wang, and R. Langar, “Computation offloading for mobile edge computing: A deep learning approach,” in Proceedings of IEEE 28th Annual International Symposium on Indoor Mobile Radio Communication, pp. 1-6, 2017