Journal of Korea Multimedia Society Vol. 24, No. 1, January 2021(pp. 51-57)
https://doi.org/10.9717/kmms.2020.24.1.051

포그 컴퓨팅 환경에서 기회적 포그 컴퓨팅 노드들을 고려한 서비스 요구사항 기반 테스크 분배 방법

경 연 웅†

Task Distribution Scheme based on Service Requirements Considering Opportunistic Fog Computing Nodes in Fog Computing Environments

Yeunwoong Kyung†

ABSTRACT

In this paper, we propose a task distribution scheme in fog computing environment considering opportunistic fog computing nodes. As latency is one of the important performance metric for IoT(Internet of Things) applications, there have been lots of researches on the fog computing system. However, since the load can be concentrated to the specific fog computing nodes due to the spatial and temporal IoT characteristics, the load distribution should be considered to prevent the performance degradation. Therefore, this paper proposes a task distribution scheme which considers the static as well as opportunistic fog computing nodes according to their mobility feature. Especially, based on the task requirements, the proposed scheme supports the delay sensitive task processing at the static fog node and delay in-sensitive tasks by means of the opportunistic fog nodes for the task distribution. Based on the performance evaluation, the proposed scheme shows low service response time compared to the conventional schemes.

Key words: Fog Computing, Opportunistic Fog, Service Requirements

1. 서 론

현재 스마트폰, Wearable 장비 등 수많은 스마트 기기들이 다양한 물리적인 상황들을 탐지하고 확인하기 위하여 센서들을 탑재하여 출시되고 있다. 이에 따라 기기 간 연결을 통해 다양한 정보들이 수집되고 공유되는 개념인 IoT(Internet of Things)가 현실화 되었고, 스마트 팩토리, 스마트 환경 등 다양한 IoT 어플리케이션들이 등장하였다[1-3]. 이러한 다양한 어플리케이션들은 각 특성에 맞는 다양한 요구사항들을 갖고 있기 때문에 요구사항을 만족시킬 수 있는 처리가 필요하다. 예를 들어 AR(Augmented Reality) navigation 등과 같이 서비스 지연에 민감한 어플리케이션들은 그에 맞게 짧은 지연시간을 제공하도록 엄격하게 요청을 처리해야 하며 컨텐츠 푸시 서비스 등과 같이 지연에 둔감한 특성이 있는 어플리케이션들은 상대적으로 엄격하게 자원을 활용할 필요는 없다[4,5]. 이러한 IoT 어플리케이션들의 요청들은 유연한 컴퓨팅 자원 활용 및 대량 데이터 처리 등을 위하여 주로 클라우드 기반으로 처리될 수 있다. 하

※ Corresponding Author : Yeunwoong Kyung, Address: (18101) Hanshindae-Gil 137, Osan, Gyeonggi, Korea, TEL : +82-10-5454-7149, FAX : +82-10-5454-7149, E-mail : ywkyung@hs.ac.kr
Receipt date : Jan. 8, 2021, Revision date : Jan. 13, 2021
Approval date : Jan. 15, 2021
† School of Computer Engineering, College of IT, Hanshin University
※ This work was supported by Hanshin University Research Grant

지만 이러한 클라우드 기반 처리는 코어 네트워크의 부하 증가 및 처리 지연 등의 제한점이 존재하기 때문에 이를 극복하고자 컴퓨팅 자원을 클라우드가 아닌 IoT 기기 및 사용자 근처로 가져오는 개념인 포그 컴퓨팅이 등장하였다[6].

포그 컴퓨팅 환경에서는 IoT 어플리케이션의 요청이 주로 BS(Base Station)에 설치된 포그 컴퓨팅 노드(Fog Computing Node, FN)에서 처리 된다[6]. 이러한 물리적인 특징으로 인해 지연을 최소화시키면서 서비스를 제공할 수 있다. 하지만 IoT 어플리케이션들의 요청이 증가하게 되면 FN으로 부하가 집중되기 때문에 병목현상이 발생할 수 있다. 이를 해결하기 위하여 이웃한 BS에 연결되어 있는 FN들을 고려하여 부하를 분배하거나 클라우드와 FN을 동시에 고려하여 부하를 분배하는 연구들이 활발히 진행되었다[6-8].

최근 기존의 정적으로 설치된 FN 뿐만 아니라 동적으로 움직이는 FN을 통한 테스크 처리에 대한 연구도 관심을 받고 있다[9-12]. 해당 연구들은 주로 차량 노드들 간 테스크를 분배하거나 포그 노드들을 함께 고려하여 테스크를 분배하는 방법을 제안하였는데 이렇게 동적인 FN을 통한 테스크 처리방법은 기회적 포그 컴퓨팅이라는 용어로 정의되었고[10] 이동특성을 갖고 있는 차량, 스마트폰, UAV(Unmanned Aerial Vehicle) 등 그 영역이 확장되고 있다[13].

하지만 기회적 포그 컴퓨팅 노드(Opportunistic Fog Node, OFN)들은 이동성 및 연결성 제약으로 인하여 항상 가용한 자원은 아니다. 또한 가용한 상황이라 하더라도 OFN과의 통신 및 컴퓨팅 지연이 발생할 수 있다. 그러므로 시간지연에 민감한 테스크의 경우 OFN을 활용할 경우 요청 손실 및 서비스 지연이 발생할 수 있다[10]. 기존 연구들은 이러한 OFN들을 이용하여 전체적인 시스템 입장에서 서비스 지연 요구사항을 만족시키면서 서비스 품질 저하 방지[9], 테스크의 의존성[12], 에너지 효율성[11] 등을 고려하였지만, 어플리케이션 요구사항의 특성을 기반으로 플로우를 분류하여 가용여부에 따라 OFN들을 활용하는 연구는 아직 진행되지 못하였다. 그러므로 본 연구에서는 어플리케이션 요구사항에 따라 어플리케이션 플로우를 분류하여 OFN들이 가용할 때 시간 지연에 민감하지 않은 테스크들을 OFN들을 통해 처리하고, 시간 지연에 민감한 테스크들은 정적인 포그 컴퓨팅 노드들을 통해 안정적으로 처리하도록 함으로써 각 요구사항을 만족시키면서 가용자원을 최대한 활용할 수 있는 방법을 제안하고자 한다. 또한 M/M/1 큐잉 모델 기반의 모델링 및 성능 분석을 수행하여 제안하는 방법과 기존 방법들에 대한 성능 비교를 진행하고자 한다.

2. 시스템 모델

2.1 시스템 아키텍처

본 논문에서는 Fig. 1과 같이 SN(Sensor Node)들과 같은 IoT 기기들이 설치되어 있고, 이 기기들은 물리적 자원 제약으로 인해 테스크를 BS에 연결되어 있는 FN으로 오프로딩 시키는 시스템을 가정하였다. 이 때, BS 영역 내에서 연결 가능한 OFN이 존재한다면 FN에 할당된 테스크를 OFN으로 전달하여 처리할 수 있다.

IoT 플로우의 경우 서비스 처리 지연시간에 대한 요구사항이 다양할 수 있다. 예를 들어 AR navigation 어플리케이션의 서비스 지연 요구사항은 최대 250ms이고, 스트리밍 어플리케이션의 경우 최대 1s로 측정된다[9]. 본 논문에서는 이러한 서비스 지연 요구사항에 따라 어플리케이션 플로우를 낮은 서비스 지연 시간을 필요로 하는 HP(High Priority) 플로우, 상대적으로 시간 지연에 덜 민감한 LP(Low Priority) 플로우로 구분하였다. HP 플로우의 경우 FN에서 처리하고자 하고 LP 플로우의 경우 OFN의 연결이 가능한 경우 OFN으로 분배시켜서 FN의 부하 집중을 방지하면서 분산적으로 처리하고자 하였다.

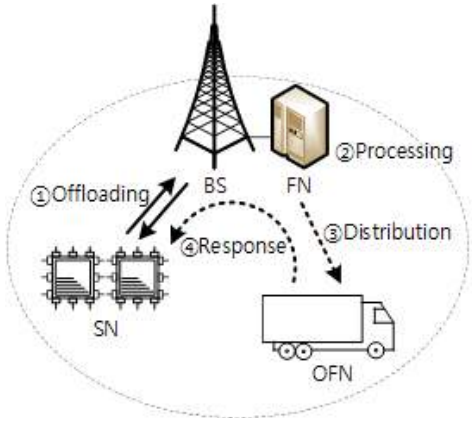


Fig. 1. Fog Computing Architecture.

2.2 제안하는 방법의 시스템 모델링

제안하는 방법의 모델링을 위해 본 논문에서는 Fig. 2에와 같이 FN과 OFN은 입력되는 어플리케이션 요청들이 Poisson 분포를 갖는 M/M/1 큐잉 모델을 따른다고 가정하였다[6,7]. Fig. 2의 상태 (i,j)에서 i는 FN의 가용여부에 따른 단계를 의미한다. 단계 A는 OFN이 SN이 속한 BS 영역 내에 있어서 OFN을 통한 부하분배가 가능한 단계를 의미하고, 단계 U는 OFN이 SN이 속한 BS 영역 내에 없어서 OFN으로의 부하분배 없이 FN에서의 테스크 처리만 진행되는 단계를 의미한다. Fig. 2에서와 같이 단계 A에서 BS의 영역 내에 OFN이 존재할 시간의 분포는 평균값이 1/η을 갖는 지수분포를 나타내고 단계 U에서의 시간 분포는 평균값이 1/ξ를 갖는 지수분포를 나타낸다[14,15]. 즉, 제안하는 방법은 FN에서 OFN으로의 부하분배가 가능한 단계 A에서는 Fig. 2(a)에서와 같이 HP 플로우는 FN에서 처리하고, Fig. 2(b)에서와 같이 LP 플로우는 OFN에서 처리하여 FN과 OFN의 부하분배 및 HP 플로우의 저지연 처리를 고려하였고, OFN으로의 부하 분배가 불가능한 단계 U에서는 Fig. 2(a)에서와 같이 FN에서 모든 플로우 요청들을 처리한다. 본 논문에서는 BS 영역 내의 SN들의 테스크는 BS에 직접적으로 연결된 FN에서만 처리되는 것을 가정하였고, 다양한 위치에 존재하는 FN들끼리의 협력적인 테스크 처리[6] 및 클라우드를 통한 테스크 처리를[8] 고려한 확장 방안은 본 연구의 향후 연구에서 진행하고자 한다.

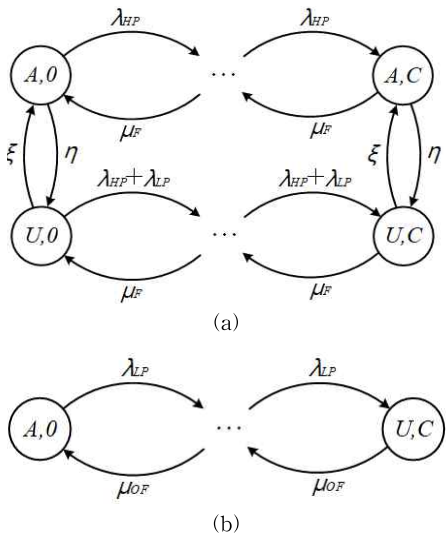


Fig. 2. Markov chain model for the (a) FN and (b) OFN in the proposed scheme.

Fig. 2(a)에서 보여주는 FN에서의 전이 확률을 나타내면 식(1)과 같다.

$$\begin{aligned}
&p(A,j;U,j)=\eta && (0\le j\le C)\\
&p(U,j;A,j)=\xi && (0\le j\le C)\\
&p(A,j;A,j-1)=\mu_F && (0<j\le C)\\
&p(A,j;A,j+1)=\lambda_{HP} && (0\le j<C)\\
&p(U,j;U,j-1)=\mu_F && (0<j\le C)\\
&p(U,j;U,j+1)=\lambda_{HP}+\lambda_{LP} && (0\le j<C)
\end{aligned} \tag{1}$$

Fig. 2(a)의 안정상태확률(πi,j)을 구하기 위하여 balance 수식을 구하면 식(2)와 같다.

$$\begin{aligned}
&1)\ i=A,j=0,\ (\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j}=\mu_F\cdot\pi_{i,j+1}+\xi\cdot\pi_{U,j}\\
&2)\ i=A,0<j<C,\ (\lambda_{HP}+\eta+\mu_F)\cdot\pi_{i,j}=\lambda_{HP}\cdot\pi_{i,j-1}\\
&\qquad +\mu_F\cdot\pi_{i,j+1}+\xi\cdot\pi_{U,j}\\
&3)\ i=A,j=C,\ (\eta+\mu_F)\cdot\pi_{i,j}=\lambda_{HP}\cdot\pi_{i,j-1}+\xi\cdot\pi_{U,j}\\
&4)\ i=U,j=0,\ (\xi+\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j}=\mu_F\cdot\pi_{i,j+1}\\
&\qquad +\eta\cdot\pi_{A,j}\\
&5)\ i=U,0<j<C,\ (\xi+\lambda_{HP}+\lambda_{LP}+\mu_F)\cdot\pi_{i,j}\\
&\quad =(\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j-1}+\mu_F\cdot\pi_{i,j+1}+\eta\cdot\pi_{A,j}\\
&6)\ i=U,j=C,\\
&\quad (\xi+\mu_F)\cdot\pi_{i,j}=(
\end{aligned} \tag{2}$$

본 논문에서는 수식(2)를 이용하여 iterative 알고리즘을 통해서 안정상태확률(πi,j)을 계산하였다[16]. HP 플로우의 요청 응답 시간을 구하기 위해서 FN에서의 평균 요청 개수를 구하면 수식(3)과 같이 나타낼 수 있다.

$$N=\sum_{i=A,U}\sum_{j=0}^{C} j\cdot\pi_{i,j} \tag{3}$$

또한 Fig. 2(a)에서의 각 단계를 고려한 effective 요청률은 수식(4)와 같이 계산될 수 있다.

$$\lambda_e=\sum_{j=0}^{C}\lambda_{HP}\cdot\pi_{A,j}+\sum_{j=0}^{C}(\lambda_{HP}+\lambda_{LP})\cdot\pi_{U,j} \tag{4}$$

수식(3)과 (4)를 이용해서 Little's 법칙에 따라 FN에서의 HP 플로우의 요청 응답 시간은 다음과 같이 계산될 수 있다.

$$W_{HP}=\frac{N}{\lambda_e} \tag{5}$$

2.3 기존 방법의 시스템 모델링

제안하는 방법과 비교하여 기존 방법들에서는 Fig. 3에서와 같이 입력되는 어플리케이션 요청들의 구분 없이 FN과 OFN에서의 처리가 진행된다[13].

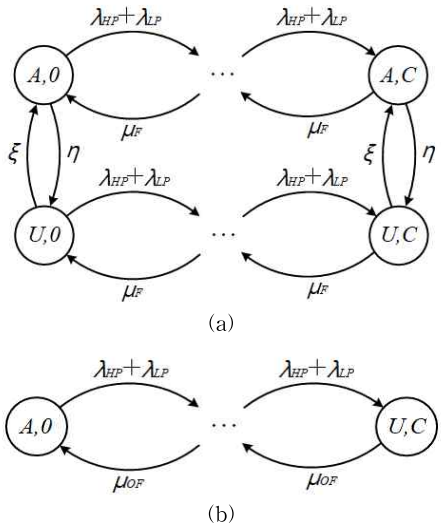


Fig. 3. Markov chain model for the (a) FN and (b) OFN in the conventional scheme.

입력되는 HP, LP 플로우의 구분 없이 차량 등의 OFN이 가용할 때는 FN과 OFN을 모두 활용하여 최대한 처리하고, OFN이 가용하지 않은 경우에는 제안하는 방법과 마찬가지로 FN만을 통해 처리하게 된다.

Fig. 3(a)에서 보여주는 FN에서의 전이 확률을 나타내면 식(6)과 같다. 식 (1)과 유사하고 단계 A에서 상태 j에서 j+1로 변경되는 확률만 차이가 있다.

$$\begin{aligned}
&p(A,j;U,j)=\eta && (0\le j\le C)\\
&p(U,j;A,j)=\xi && (0\le j\le C)\\
&p(A,j;A,j-1)=\mu_F && (0<j\le C)\\
&p(A,j;A,j+1)=\lambda_{HP}+\lambda_{LP} && (0\le j<C)\\
&p(U,j;U,j-1)=\mu_F && (0<j\le C)\\
&p(U,j;U,j+1)=\lambda_{HP}+\lambda_{LP} && (0\le j<C)
\end{aligned} \tag{6}$$

Fig. 3(a)의 안정상태확률(πi,j)을 구하기 위하여 balance 수식을 구하면 식(7)와 같다.

$$\begin{aligned}
&1)\ i=A,j=0,\ (\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j}=\mu_F\cdot\pi_{i,j+1}+\xi\cdot\pi_{U,j}\\
&2)\ i=A,0<j<C,\ (\lambda_{HP}+\lambda_{LP}+\eta+\mu_F)\cdot\pi_{i,j}\\
&\quad=(\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j-1}+\mu_F\cdot\pi_{i,j+1}+\xi\cdot\pi_{U,j}\\
&3)\ i=A,j=C,\ (\eta+\mu_F)\cdot\pi_{i,j}=(\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j-1}\\
&\quad+\xi\cdot\pi_{U,j}\\
&4)\ i=U,j=0,\ (\xi+\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j}=\mu_F\cdot\pi_{i,j+1}\\
&\quad+\eta\cdot\pi_{A,j}\\
&5)\ i=U,0<j<C,\ (\xi+\lambda_{HP}+\lambda_{LP}+\mu_F)\cdot\pi_{i,j}\\
&\quad=(\lambda_{HP}+\lambda_{LP})\cdot\pi_{i,j-1}+\mu_F\cdot\pi_{i,j+1}+\eta\cdot\pi_{A,j}\\
&6)\ i=U,j=C,\\
&\quad(\xi
\end{aligned} \tag{7}$$

제안하는 방법과 마찬가지로 iterative 알고리즘을 통해서 안정상태확률(πi,j)을 계산할 수 있고, 수식 (3), (4). (5)를 이용하여 HP, LP 플로우의 요청 응답 시간을 계산할 수 있다. 또한 OFN을 고려하지 않는 기존 방법들은 Fig. 3(a)의 단계 A만 고려되는 것으로 모델링이 가능하다.

3. 성능 분석 결과

본 장에서는 OFN을 사용하지 않고 FN만을 고려하는 방법(Basic)과 Fig. 3으로 모델링되는 플로우의 분류 없이 OFN을 사용하는 방법(No Differentiation, ND)[13], 그리고 2.2절에서 모델링한 제안하는 방법을 비교하여 서비스 응답 지연에 대해 성능 분석을 진행하고자 한다.

Fig. 4(a)는 η과 ξ이 모두 1/3이고 HP 플로우의 평균 요청률이 0.3일 때, LP 플로우의 증가에 따른 HP 플로우의 서비스 지연 시간을 보여준다. ND 방법은 플로우의 구분 없이 분배시키기 때문에 제안하는 방법과 비교하여 긴 서비스 지연 시간이 소요된다. ND 방법은 Basic 방법과 비교하여 볼 때, 입력되는 LP 플로우의 양이 적을 때는 Basic 방법보다 서비스 지연 시간이 더 소요되는데, 이는 FN의 부하가 적을 때에도 OFN이 가용할 때 FN보다 처리 용량 및 성능이 상대적으로 낮은 OFN을 통해 처리하기 때문이다. 본 장에서의 결과에서는 고려하지 않았지만 OFN으로의 분배 지연 및 응답 지연을 고려하게 되면 이 영향은 더 커질 수 있기 때문에 제안하는 방법과 같이 플로우의 구분에 따른 OFN 활용이 필요하다는 것을 알 수 있다. Fig. 4(b)는 η과 ξ이 각각 2/3, 1/3일 때, LP 플로우의 증가에 따른 HP 플로우의 서비스 지연 시간을 보여준다. Fig. 4(b)는 Fig. 4(a)와 비슷한 경향성을 보이면서 제안하는 방법이 가장 낮은 서비스 지연 시간을 보여주지만 HP 플로우의 양이 증가하면서 ND 방법과 제안하는 방법의 서비스 지연 시간 차이가 줄어드는 것을 볼 수 있다. 이는 OFN을 가용할 수 있는 확률이 높은 상황에서는 HP 플로우의 입력량이 고정되어 있을 경우, OFN을 통해 부하 분배가 원활히 이루어져 ND 방법에서 FN의 부하가 Fig. 4(a) 상황에서보다 더 줄어들기 때문이다. 해당 상황을 고려하여 네트워크 운영자는 OFN을 최대한 사용하면서 지연 요구사항은 만족시킬 수 있도록 플로우 분류 기준을 유연하게 변경하면

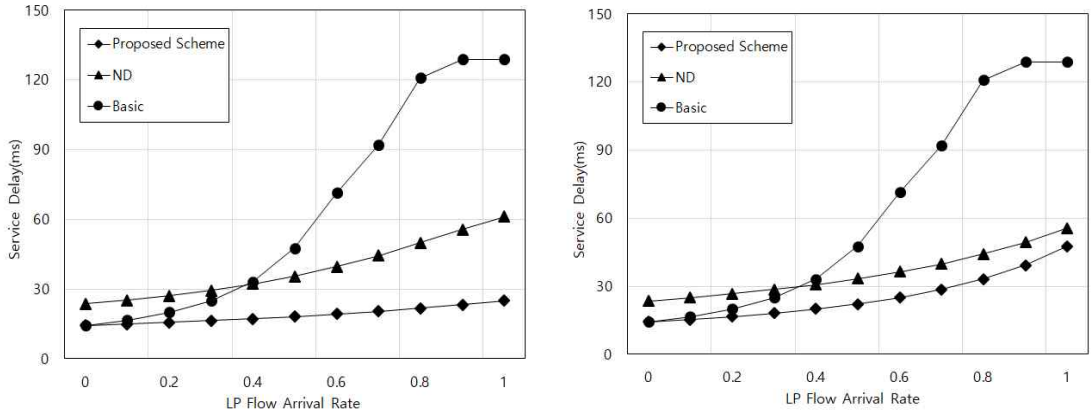


Fig. 4. Response delay time according to the LP flow. (a) η =1/3, ξ =1/3, (b) η =2/3, ξ =1/3.

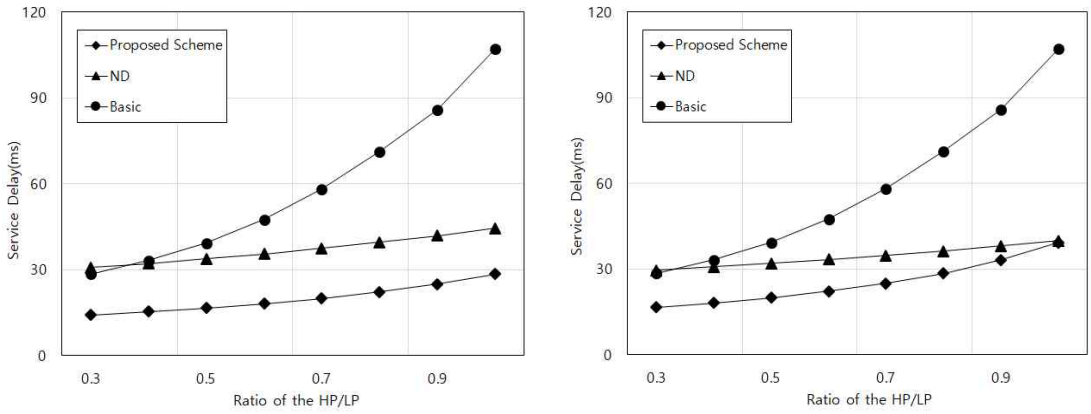


Fig. 5. Response delay time according to the ratio of the HP/LP. (a) η =1/3, ξ =1/3, (b) η =2/3, ξ =1/3.

서 HP 플로우를 동적으로 분류하게 되면 제안하는 방법의 효과가 더 커질 수 있을 것으로 기대된다.

Fig. 5(a)와 (b)는 각각 η과 ξ이 모두 1/3일 때와 2/3, 1/3일 때의 HP 플로우와 LP 플로우의 비율에 따른 HP 플로우의 서비스 지연 시간을 나타낸다. LP 플로우의 양을 0.5로 가정하고 HP 플로우의 양을 비율에 따라 증가시키면서 분석하였다. Fig. 5(a)에서는 HP 플로우와 LP 플로우의 비율이 같아지더라도 제안하는 방법과 ND 방법의 차이가 줄지 않는 것을 확인할 수 있는데, 이는 FN의 부하가 적을 때에도 OFN이 가용할 때 FN보다 처리 용량 및 성능이 상대적으로 낮은 OFN을 통해 처리하기 때문이다. 반면에 Fig. 5(b)에서는 HP 플로우와 LP 플로우의 비율이 같아질수록 제안하는 방법과 ND 방법의 차이가 줄어드는데, 이는 입력되는 부하가 커진 상황에서는 OFN을 가용할 수 있는 확률이 높은 경우, 제안하는 방법과 ND 방법 모두 FN과 OFN을 최대한 활용하여 처리하게 되기 때문이다. Basic 방법의 경우 Fig. 4와 마찬가지로 OFN을 고려하지 않기 때문에 비율에 상관없이 동일한 지연 시간 성능을 보인다.

4. 결 론

본 논문은 포그 컴퓨팅 환경에서 OFN들을 고려한 테스크 분배 방법을 제안하였다. 제안하는 방법은 서비스 플로우를 지연 요구사항에 따라 분류하여 지연에 민감한 플로우는 정적인 포그 컴퓨팅 노드를 통해 처리하도록 하고, 지연에 둔감한 플로우는 이동 특성

으로 인해 가용 여부가 동적으로 결정되는 OFN을 함께 활용하여 처리하도록 한다. 큐잉 모델 기반의 모델링 및 성능 분석을 수행하여 제안하는 방법이 서비스 플로우의 요구사항을 고려한 차등적인 OFN 자원 활용을 통해 기존 방법들에 비교하여 시스템 부하가 증가하더라도 지연에 민감한 서비스의 요구사항을 만족시킬 수 있음을 확인하였다. 본 연구는 이웃한 포그 컴퓨팅 노드들과 클라우드 컴퓨팅 노드를 함께 고려하여 다양한 지연 요구사항을 만족시킬 수 있는 최적의 테스크 분배 방법을 찾는 방향으로 확장시킬 예정이다.

REFERENCE

[1] H. Hwang and Y. Seo, "A Development of Real-time Energy Usage Data Collection and Analysis System Based on the IoT," *Journal of Korea Multimedia Society*, Vol. 22, No. 3, pp. 366-373, 2019.

[2] J. Ahn and B.M. Lee, "IoT Roaming Service for Seamless IoT Service," *Journal of Korea Multimedia Society*, Vol. 23, No. 10, pp. 1258-1269, 2020.

[3] S.H. Jung, J.Y. Kim, J. Park, S.M. Jang, and C.B. Sim, "A Study On Power Data Analysis And Rist Situation Prediction Using Smart Plug," *Journal of Korea Multimedia Society*, Vol. 23, No. 7, pp. 870-882, 2020.

[4] P. Si, Y. He, H. Yao, R. Yang, and Y. Zhang, "Deve: Offloading Delay-Tolerant Data Traffic to Connected Vehicle Networks," *IEEE Transactions on Vehicular Technology*, Vol. 65, No. 6, pp. 3941-3953, 2016.

[5] M. Li, P. Si, and Y. Zhang, "Delay-Tolerant Data Traffic to Software-Defined Vehicular Networks with Mobile Edge Computing in Smart City," *IEEE Transactions on Vehicular Technology*, Vol. 67, No. 10, pp. 9073-9086, 2018.

[6] Q. Fan and N. Ansari, "Towards Workload Balancing in Fog Computing Empowered IoT," *IEEE Transactions on Network Science and Engineering*, Vol. 7, No. 1, pp. 253-262, 2018.

[7] X. Sun and N. Ansari, "Latency Aware Workload Offloading in the Cloudlet Network," *IEEE Communications Letters*, Vol. 21, No. 7, pp. 1-22, 2017.

[8] J. Ren. G. Yu, Y. He, and Y. Li, "Collaborative Cloud and Edge Computing for Latency Minimization," *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 5, pp. 5031-5044, 2019.

[9] C. Zhu, J. Tao, G. Paster, Y. Xiao, Y. Ji, Q. Zhou, et al., "Folo: Latency and Quality Optimized Task Allocation in Vehicular Fog Computing," *IEEE Internet of Things Journal*, Vol. 6, No. 3, pp. 4150-4161, 2019.

[10] N. Fernando, S.W. Loke, I. Avazpour, F. Chen, A.B. Abkenar, and A. Ibrahim, "Opportunistic Fog for IoT: Challenges and Opportunities," *IEEE Internet of Things Journal*, Vol. 6, No. 5, pp. 8897-8910, 2019.

[11] Z. Ning, J. Juang, X. Wang, J.J.P.C. Rodrigues, and L. Guo, "Mobile Edge Computing-Enabled Internet of Vehicles: Toward Energy-Efficient Scheduling," *IEEE Network*, Vol. 33, No. 5, pp. 198-205, 2019.

[12] Y. Liu, W. Wang, Q. Zhao, S. Du, A. Zhou, X. Ma, et al., "Dependency-Aware Task Scheduling in Vehicular Edge Computing," *IEEE Internet of Things Journal*, Vol. 7, No. 6, pp. 4961-4971, 2020.

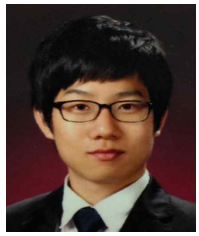
[13] Z. Zhou, P. Liu, J. Feng, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Computation Resource Allocation and Task Assignment Optimization in Vehicular Fog Computing: A Contract-Matching Approach," *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 4, pp. 3113-3125, 2019.

[14] J. Lee, G. Lee, and S. Pack, "Pseudonyms in IPv6 ITS Communications: Use of Pseudonyms, Performance Degradation, and Optimal Pseudonyms Change," *International Journal of Distributed Sensor Networks*, Vol. 11, No.

5, pp. 1-7, 2015.

[15] Y. Liu, W. Wang, Y. Ma, Z. Yang, and F. Yu, "Distributed Task Offloading in Heterogeneous Vehicular Crowd Sensing," *MDPI Sensors*, Vol. 16, No. 7, pp. 1481-1484, 2016.

[16] Y. Kim, H. Ko, S. Pack, W. Lee, and X. Shen, "Mobility-Aware Call Admission Control Algorithm with Handoff Queue in Mobile Hotspots," *IEEE Transactions on Vehicular Technology*, Vol. 62, No. 8, pp. 3903-3912, 2013.



경 연 웅

2011년 2월 고려대학교 전기전자 전파공학부(공학사)
2016년 8월 고려대학교 전기전자 전파공학부(공학박사)
2016년 9월~2020년 3월 삼성전자 무선사업부 책임연구원
2020년 3월~현재 한신대학교 컴퓨터공학부 조교수