KIPS Trans. Comp. and Comm. Sys.
Vol.9, No.11 pp.250~255
ISSN: 2287-5891 (Print), ISSN 2734-049X (Online)
https://doi.org/10.3745/KTCCS.2020.9.11.250

Service Image Placement Mechanism Based on the Logical Fog Network

Jonghwa Choi† · Sanghyun Ahn††

ABSTRACT

For the resolution of the latency problem of the cloud center-based cloud computing, fog computing was proposed that allows end devices to offload computations to nearby fog nodes. In the fog computing, virtualized service images are placed on fog nodes and, if service images are placed close to end devices, the duplicate service image placement problem may occur. Therefore, in this paper, we propose a service image placement mechanism based on the logical fog network that reduces duplicate service images by considering the pattern of collected service requests. For the performance evaluation of the proposed mechanism, through simulations, we compare ours with the on-demand mechanism placing a service image upon the receipt of a service request. We consider the performance factors like the number of service images, the number of non-accommodated service requests, and the network cost.

Keywords : Fog Computing, Service Image Placement, Service Provisioning, Computation Offloading, Cloud Computing

논리적 포그 네트워크 기반의 서비스 이미지 배치 기법

최 종 화† · 안 상 현††

요 약

클라우드 센터 기반 클라우드 컴퓨팅 방식의 지연시간 문제를 해결하기 위해, 단말 장치에서 가까운 포그 노드에게 컴퓨테이션 오프로딩(Computation offloading)을 하는 포그 컴퓨팅 방식이 제안되었다. 포그 컴퓨팅에서는 포그 노드에 가상화된 서비스 이미지가 배치되며, 단말 장치와 가까운 포그 노드에 서비스 이미지를 배치하는 경우 동일한 서비스 이미지가 여러 포그 노드에 중복 배치되는 문제가 발생할 수 있다. 따라서 본 논문에서는 단말 장치로부터 수집된 서비스 요청 패턴을 고려해서 서비스 이미지의 중복 배치를 최소화하는 논리적 포그 네트워크 기반의 서비스 이미지 배치 기법을 제안한다. 제안 기법의 성능 평가를 위해 시뮬레이션을 통해 서비스 요청이 있을 때 동적으로 서비스 이미지를 할당하는 기법과 제안 기법의 성능을 비교하며, 성능 분석 요소로서 서비스 이미지 배치 수, 수용되지 못한 서비스 요청 수, 네트워크 비용을 고려한다.

키워드 : 포그 컴퓨팅, 서비스 이미지 배치, 서비스 프로비저닝, 컴퓨테이션 오프로딩, 클라우드 컴퓨팅

1. 서 론

에너지 관리, 공장 관리, 스마트 팜, 차량 흐름 제어 등과 같이 다양한 분야에서 사물인터넷(Internet of Things, IoT) 서비스가 클라우드 컴퓨팅 기술을 기반으로 제공된다. 그러나 중앙 클라우드 센터에서 모든 사용자의 요청을 처리하기 때문에 응답시간이 길어지는 단점이 있으며, 단말 장치(End device)와 가까운 위치에 서비스를 분담 처리해주는 포그(Fog) 노드를 배치해서 응답시간을 줄이는 포그 컴퓨팅이 제안됐다[1]. 포그 컴퓨팅은 클라우드 센터와 단말 장치 사이에 포그 노드로 구성된 가상화 플랫폼을 추가하며, 클라우드 센터는 서비스를 가상화된 이미지로 만들어서 보관하고, 단말 장치의 서비스 요청에 따라 해당 서비스의 가상화 이미지를 포그 노드에 배치한다. 포그 노드에서는 클라우드 센터로부터 받은 서비스 이미지를 실행해서 단말 장치에게 서비스를 제공한다. 포그 컴퓨팅 환경에서의 가상화 이미지는 서비스 실행 환경에 적합한 경량 가상화 이미지로, 신속한 서비스 이미지 배치와 효율적인 자원 사용을 가능하게 해준다.

포그 컴퓨팅 환경에서의 서비스 이미지 배치에 대한 기존 연구로는 동적으로 요구된 서비스에 대해 주어진 제약조건을 만족하는 서비스 이미지 배치 기법들이 있다[2, 3]. 요구되는 서비스 요청들에 대해 서비스 요청마다 각 시점에 적합한 서비스 이미지를 배치하게 되면, 이미 할당된 자원 환경 하에서 서비스 이미지 배치를 고려하기 때문에 전체 자원이 비효율적으로 사용될 수 있다[4]. 특히 서비스 요청이 많으면 더 비효율적이 될 수 있다.

※ 이 논문은 2019년도 서울시립대학교 연구년교수 연구비에 의하여 연구되었음.
† 준 회 원 : 코인플러그 연구원
†† 종신회원 : 서울시립대학교 컴퓨터과학부 교수
Manuscript Received : September 16, 2020
Accepted : October 22, 2020
* Corresponding Author : Sanghyun Ahn(ahn@uos.ac.kr)

다수의 서비스 유형과 서비스 요청이 있는 상황에서 포그 노드에 서비스 이미지를 배치할 때 발생할 수 있는 문제로는, (1) 수요가 적은 동일 서비스 이미지가 다수의 포그 노드에 중복으로 배치되어 자원을 비효율적으로 사용하는 것, (2) 서비스 이미지가 사용자 분포와 무관하게 상위 계층의 포그 노드에 배치되어 응답시간이 길어지는 것이 있다. 이 문제들은 서비스 이미지 배치 시 서비스별 사용자 수요와 사용자의 지역 분포 등을 고려하지 않았기 때문에 생긴다. 서비스 이미지 배치를 최소화할 경우 첫 번째 문제는 해결할 수 있으나 응답시간이 길어질 수 있는 반면, 서비스 응답시간 최소화를 할 경우 두 번째 문제는 해결할 수 있으나 서비스 이미지가 더 배치될 수 있다.

본 논문에서는 앞서 언급한 두 가지 문제 중 서비스 이미지의 중복 배치 문제 해결에 초점을 맞추어, 다수의 서비스 유형과 요청 관련 정보가 수집된 상황에서 [4]에서 제안한 것과 같이 그물망(Mesh) 형태의 물리적 네트워크를 포그 노드 중심의 논리적 포그 네트워크로 간략화하고, 논리적 포그 네트워크를 기반으로 서비스 이미지 배치가 최소화될 수 있도록 하는 기법을 제시한다. 제안 기법의 성능 분석을 위해 개별 서비스 요청 시 동적으로 서비스 이미지를 배치해주는 방식과 비교하며, 총 서비스 이미지 배치 수, 수용되지 않은 서비스 요청 수, 네트워크 비용 측면에서 시뮬레이션을 통해 성능을 분석한다.

본 논문의 구성은 다음과 같다. 2절에서 포그 컴퓨팅, 서비스 이미지 배치 등과 관련된 기존 연구에 대해서 소개하며, 3절에서 제안하는 서비스 이미지 배치 기법에 대해서 설명한다. 4절에서는 시뮬레이션을 통해 제안 기법의 성능을 분석한다. 끝으로, 5절에서 결론을 맺는다.

2. 관련 연구

컴퓨테이션 오프로딩(Computation offloading)은 서비스 작업을 단말 장치에서 실행하지 않고 클라우드 센터나 포그 노드에게 위탁하는 것을 의미하며, 서비스 프로비저닝(Service provisioning)은 클라우드 센터나 포그 노드로부터 서비스를 제공받기 위해 CPU, RAM, 저장공간 등의 자원을 미리 적절하게 할당해서 서비스 이미지를 배치 운영하는 것이다[5]. 컴퓨테이션 오프로딩과 관련해서 서비스 프로비저닝에 대한 연구가 선행되어야 하며, 본 논문에서는 서비스 프로비저닝에 대한 연구 중 서비스 이미지를 클라우드 센터 혹은 포그 노드에 적절하게 배치하는 것에 대해서 다룬다.

포그 컴퓨팅을 위해 서비스 이미지를 할당하기 위한 포그-클라우드 아키텍쳐에 대한 연구[6, 7], 이미 할당된 서비스 이미지를 다른 포그 노드에 할당하는 마이그레이션에 대한 연구[8], 그리고 서비스 품질(Quality of Service)과 체감 품질(Quality of Experience)을 기반으로 한 연구[2, 3, 9, 10] 등이 있다.

[6, 7]에서는 포그 컴퓨팅 환경을 한 개의 클라우드 센터와 포그 콜로니들로 구성하며, 포그 콜로니는 컨트롤 노드, 포그 셀들로 구성된다. 포그 콜로니의 컨트롤 노드는 모든 서비스에 대한 서비스 이미지를 갖고 있으며, 서비스 제약사항을 만족하고 응답시간을 최소화하는 포그 셀을 선정해서 컨트롤 노드에 있는 서비스 이미지를 해당 포그 셀에 배치한다. [8]은 서비스 이미지 저장 및 포그 노드 탐색 등 서비스 프로비저닝을 위한 API를 제공하는 Foglets을 제안했으며, 이미 배치된 서비스 이미지에 대해서 동적으로 포그 노드를 탐색하고 서비스 이미지를 재배치시키는 마이그레이션 작업을 수행한다.

[2]는 하나의 클라우드 계층과 두 개의 포그 계층으로 구성된 포그 컴퓨팅 환경을 제안하고 모든 서비스 이미지가 모든 포그 노드에 배치되는 것을 가정했다. 포그 노드의 자원 공간을 시간 슬롯으로 표현하고 서비스 제공을 위해 일정 시간동안 슬롯을 할당해준다. [3]은 서비스 제약사항을 만족하는 포그 노드를 빠르게 탐색해서 서비스 이미지를 배치한다. [9]는 포그 노드들을 지역별로 나누고 해당 지역에서 요구된 서비스에 한해 해당 지역의 포그 노드들에게 서비스 이미지를 배치하며, 이때 서비스가 요구하는 제약조건을 만족하는 포그 노드를 찾는다. [10, 11]은 서비스 종류를 하나 또는 여러 포그 노드로부터 서비스를 제공받는 것으로 분류하며, 서비스 제약조건을 만족하는 포그 노드에게 서비스 이미지를 배치한다.

앞서 언급한 연구 [2]는 해당 서비스를 단말 장치에게 제공하기 위해서 사전에 미리 서비스 이미지가 배치됨을 가정했고, [3]은 단순히 서비스 이미지를 배치할 포그 노드를 빠르게 찾는 것을 목적으로 했다. [9]는 해당 지역에 있는 포그 노드만 서비스를 제공할 수 있으며 상위계층에 있는 포그 노드를 활용하지 않는다. [10, 11]은 사용자의 서비스 요청과 무관하게 단순히 수행 가능한 포그 노드에게 서비스 이미지를 배치한다. 이들 연구에서는 서비스 이미지를 포그 노드에 배치함에 있어서 서비스 유형 수, 서비스 요청 수에 대한 고려를 하지 않는다.

[4]에서는 동적으로 요구되는 서비스 요청들 각각에 대해서 요청 시점에 적합한 서비스 이미지들을 배치하는 경우 자원 활용 측면에서 비효율적인 점을 언급하고 이를 해결하기 위한 확장 가능한 서비스 이미지 배치 기법의 필요성을 제기하였다. 따라서 본 논문에서는 다수의 서비스 유형과 서비스 요청에 대해 포그 노드의 자원을 효율적으로 사용할 수 있도록 하는 서비스 이미지 배치 문제를 다룬다. 서비스 제약조건을 만족하면서 포그 노드 자원을 효율적으로 사용할 수 있도록 하기 위해서 서비스 이미지가 포그 노드에 중복 배치되는 문제를 해결한다.

3. 논리적 포그 네트워크 기반 최적 서비스 이미지 배치 기법

일반적인 물리적 네트워크는 그물망(Mesh) 형태의 토폴로지를 형성하며, 클라우드 센터를 중심으로 포그 노드들이 배치된다. 단말 장치는 클라우드 센터보다 포그 노드에 더 가까우며, 포그 노드마다 서비스 이미지 할당 공간이 상이하게 구성된다. 클라우드 센터는 단말 장치에서 요청한 서비스에 대해 서비

스 이미지를 배치할 포그 노드를 결정하며, 포그 노드는 클라우드 센터에 의해서 배치된 서비스를 단말 장치에게 직접 제공하고, 하위 계층의 포그 노드는 상위 계층의 포그 노드로부터 서비스를 제공에 대해 데이터 전달 경로로서의 역할을 수행한다.

실제로는 서비스마다 요구하는 자원 양이 상이하지만, 편의상 각 서비스 이미지 배치에 필요한 자원 양이 동일하다고 가정한다. 클라우드 센터는 포그 노드의 물리적 위치, 단말 장치가 요청하는 서비스 등에 대해서 알고 있으며, 서비스 제공 응답시간은 서비스 이미지가 배치된 포그 노드와 단말 장치 간 거리에 비례한다고 가정한다.

서비스 이미지의 배치는 포그 컴퓨팅 환경에서 서비스 응답시간에 영향을 미치며, 짧은 서비스 응답시간만을 목적으로 할 경우 서비스 이미지를 단말 장치와 가장 가까운 포그 노드에 우선적으로 배치하면 된다. 그러나 동일한 서비스를 넓게 분포된 다수의 단말 장치가 요청하면 동일한 서비스 이미지가 다수의 포그 노드에 중복 배치되어 자원 사용이 비효율적으로 된다. 따라서 포그 노드에 배치되는 서비스 이미지 수를 최소화할 필요가 있으며, Table 1에 관련 기호들이 정리되어 있다.

Table 1. Notations

Notation	Description
S	The set of service types, $S=\{s_0, s_1, \ldots, s_\alpha\}$
F	The set of fog nodes, $F=\{f_0, f_1, \ldots, f_\beta\}$
U	The set of end devices, $U=\{u_0, u_1, \ldots, u_\gamma\}$
X	The matrix of service image placement, $x_{ij}=\begin{cases}1, \text{ if } s_i \, is \, placed \, on \, f_j \\ 0, \, otherwise\end{cases}$
D	The matrix of service requests, $d_{ij}=\begin{cases}1, \text{if } u_i \, requests \, s_j \\ 0, otherwise\end{cases}$
c_i	The maximum capacity for service images in f_i
M	The connectivity matrix of the logical fog network $m_{ij}=\begin{cases}1, \text{if } u_i \, is \, a \, descendant \, of \, f_j \\ 0, otherwise\end{cases}$
R	The matrix of the number of service requests per service type, $R=D\times M$

본 논문에서는 논리적 포그 네트워크의 연결 행렬 M, 서비스 요청 행렬 D가 주어졌을 때 최소한의 서비스 이미지를 포그 노드에 배치하는 서비스 이미지 배치 행렬 X를 찾고자 한다. 이때 다음 제약조건을 고려해야 한다. 첫째, 단말 장치는 서비스 요청에 대한 응답을 특정 지연 시간 이내에 제공받아야 한다. 둘째, 클라우드 센터에서 서비스 이미지를 배치할 때 해당 포그 노드에 서비스 이미지를 배치할 수 있을 만큼 가용 자원이 있어야 한다. 본 논문에서는 서비스 이미지 배치 수의 최소화에 중점을 맞추며 응답시간은 제약조건으로 고려하지 않는다.

노드들이 그물망 형태로 복잡하게 연결된 경우, 서비스 이미지 배치를 위한 고려 요소가 많아서 계산 복잡도가 높아진다. 계산 복잡도를 낮추기 위해 그물망 형태의 물리적 네트워크를 단순화시키며, 이를 위해 [4]에서는 서비스 이미지 배치 전에 물리적 네트워크를 클라우드 센터와 포그 노드들로 구성된 트리 형태의 논리적 포그 네트워크로 단순화한다. 주변 포그 노드에 비해 서비스 이미지 할당 공간이 많은 몇 개의 포그 노드들을 클라우드 센터에 논리적으로 연결함으로써 클라우드 센터가 중심이 되는 방사형 토폴로지를 구성한다. 할당 공간이 많은 포그 노드들을 먼저 클라우드 센터의 차하위 계층에 두는 이유는 많이 요청되는 서비스의 이미지를 상위 계층 포그 노드에 배치함으로써 서비스 이미지를 적게 배치할 수 있기 때문이다. 그 뒤 방사형 토폴로지에 연결된 포그 노드들을 루트로 하는 최단 경로 서브 트리를 형성한다. 논리적 포그 네트워크의 연결 행렬 M을 구하며, M의 원소 m_{kj}는 단말 장치 u_k가 포그 노드 f_j의 하위 계층에 존재하는 경우 1, 그렇지 않은 경우 0의 값을 갖는다. 예를 들어, Fig. 1의 물리적 네트워크는 Fig. 2와 같은 논리적 포그 네트워크로 단순화된다. Fig. 1에서, 편의상, 클라우드 센터를 포그 네트워크에 포함시키지 않았으며, f_0가 최상위 포그 노드(즉, 클라우드 센터의 차상위 포그 노드)이다.

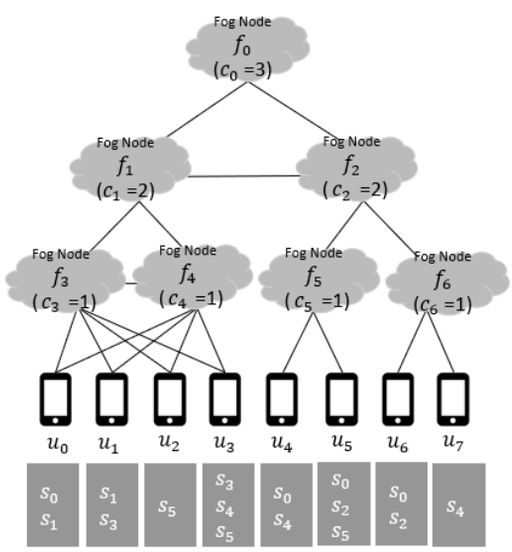

Fig. 1. An Example of a Physical Network

단말 장치가 많이 요청하는 서비스 이미지를 상위 계층 포그 노드들에게 배치하면 적은 수의 서비스 이미지 배치로 서비스 제공이 가능한 반면, 많이 요청되는 서비스의 이미지를 하위 계층 포그 노드들에게 배치하면 동일 서비스에 대한 서비스 이미지가 여러 포그 노드에 중복으로 배치될 수 있다. 따라서 논리적 포그 네트워크를 기반으로 단말 노드로부터의 서비스 요청 수를 고려해서 서비스 이미지를 포그 노드에 배치하는 SR-SIP(Service Request based Service Image Placement) 기법을 제안한다.

SR-SIP 기법에서는 먼저 클라우드 센터에서 각 포그 노드에게 요청한 서비스 유형별 서비스 요청 수를 의미하는 행렬 R을 구하기 위해 단말 장치가 요청한 서비스를 나타내는 행렬 D와 논리적 포그 네트워크에서 단말 장치가 서비스를 요청할 수 있는 포그 노드들을 나타내는 행렬 M을 곱한다. 그

다음 R을 이용해서 상위 계층부터 하향으로 포그 노드 f_j에 대해서 서비스 유형별 총 서비스 요청 수를 기준으로 서비스 유형을 내림차순 정렬해서 서비스 요청이 많은 것부터 차례대로 서비스 s_i가 제약조건을 만족하면 s_i의 이미지를 f_j에 배치한다(즉, x_{ij}는 1이 된다). s_i의 이미지가 배치된 f_j의 자식 포그 노드들에게는 더 이상 s_i의 이미지를 배치할 필요가 없기 때문에 r_{ij} 값을 0으로 설정한다. 예를 들어, Fig. 2와 같이 논리적 포그 네트워크가 구성되었고 그에 해당하는 M과 D가 있으면 Fig. 3과 같이 R을 구할 수 있다. R의 원소 r_{ij}는 f_j에 요청되는 s_i 요청 건 수이다.

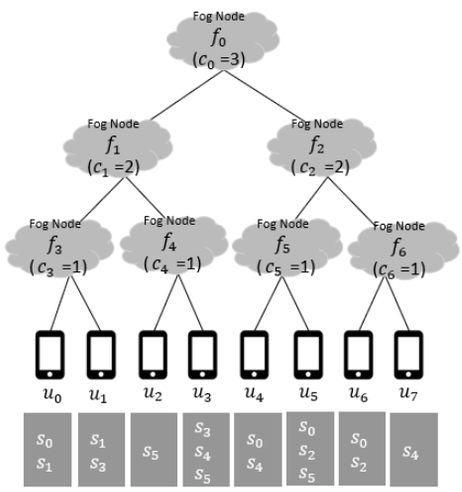

Fig. 2. The Logical Fog Network Corresponding to (Fig. 1)

$$D \times M = R$$

$$\begin{bmatrix} 1&0&0&0&1&1&1&0 \\ 1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&1&1&0 \\ 0&1&0&1&0&0&0&0 \\ 0&0&0&1&1&0&0&1 \\ 0&0&1&1&0&1&0&0 \end{bmatrix} \times \begin{bmatrix} 1&1&0&1&0&0&0 \\ 1&1&0&1&0&0&0 \\ 1&1&0&0&1&0&0 \\ 1&1&0&0&1&0&0 \\ 1&0&1&0&0&1&0 \\ 1&0&1&0&0&1&0 \\ 1&0&1&0&0&0&1 \\ 1&0&1&0&0&0&1 \end{bmatrix} = \begin{bmatrix} 4&1&3&1&0&2&1 \\ 2&2&0&2&0&0&0 \\ 2&0&2&0&0&1&1 \\ 2&2&0&1&1&0&0 \\ 3&1&2&0&1&1&1 \\ 3&2&1&0&2&1&0 \end{bmatrix}$$

Fig. 3. The Computation of R for the Example in (Fig. 2)

Fig. 4에서 R의 첫 번째 열은 f_0에 대한 서비스 유형별 서비스 요청 수를 의미한다. 서비스 요청 수를 내림차순 정렬해서 차례대로 f_0에 해당 서비스 이미지를 배치할 경우 제약조건이 만족되는지 검사한다. 이 예에서는 응답시간 제약조건이 항상 만족된다고 가정하고 가용 자원 제약조건만 고려한다. f_0의 서비스 이미지 할당 공간 c_0은 3이며(즉, 최대 3개의 서비스 이미지가 배치될 수 있다), 서비스 s_0, s_4, s_5의 이미지를 배치한다(x_{00}, x_{40}, x_{50}는 1이 된다). s_0, s_4, s_5의 이미지가 f_0에 배치되었기 때문에 f_0의 자식 포그 노드들인 f_1, f_2, f_3, f_4, f_5, f_6에 서비스 s_0, s_4, s_5의 이미지를 배치할 필요가 없기 때문에 r_{01}, . . . , r_{06}, r_{41}, . . . , r_{46}, r_{51}, . . ., r_{56}을 0으로 설정한다. f_0에 여유 공간이 없기 때문에 다음 포그 노드에 대해 위 작업을 반복한다. R의 두 번째 열은 f_1에 대한 서비스 유형별 서비스 요청 수를 의미한다. f_0와 동일하게 내림차순으로 f_1에 서비스 유형별로 제약조건이 만족되는지 검사한다. f_1의 서비스 이미지 할당 공간 c_1은 2로 최대 2개의 서비스 이미지 배치가 가능하며 s_1, s_3의 이미지가 배치된다(x_{11}, x_{31}은 1이 된다). 그리고 f_1의 자식 포그 노드들인 f_3, f_4에 s_1, s_3의 이미지를 배치할 필요가 없기 때문에 r_{13}, r_{14}, r_{33}, r_{34}를 0으로 설정한다. 이렇게 해서 구한 서비스 이미지 배치 행렬 X가 Fig. 5에 나타나 있다.

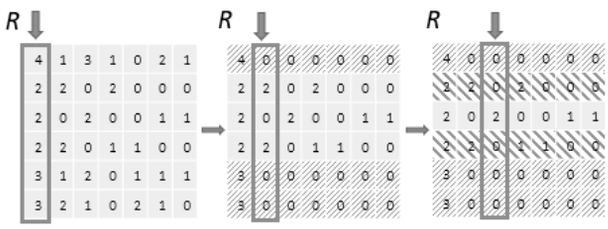

Fig. 4. The Procedure of Service Image Placement on Fog Nodes in (Fig. 2)

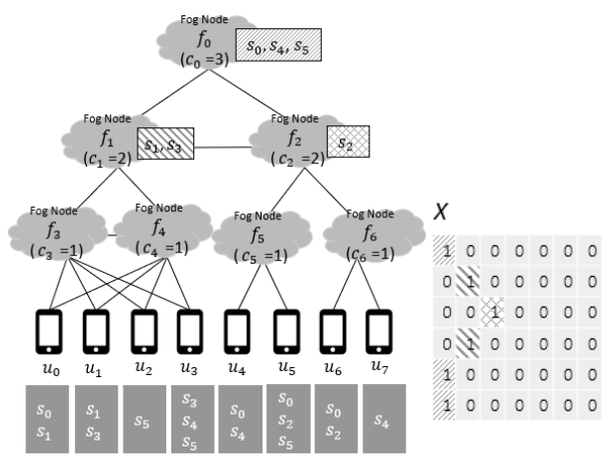

Fig. 5. The Result of Service Image Placement of (Fig. 2)

4. 성능 분석

제안 기법의 성능 분석을 위해 Python 언어와 NetworkX 패키지를 사용했으며, 시뮬레이션 네트워크 환경을 [4]와 동일하게 서울시 기반의 클라우드 센터 1개, 포그 노드 449개로 구성했으며, 서울 열린데이터 광장[12]에서 제공하는 행정구/행정동 세대 수에 비례해서 포그 노드의 서비스 이미지 할당 공간 크기를 정했다. 네트워크의 연결성은 특정 범위 안에 있는 행정구, 행정동 간에 설정했으며, 포그 노드 간 엣지의 가중치는 거리에 비례해서 설정했다. 서비스 이미지 배치에 앞서 물리적 네트워크에 해당하는 논리적 포그 네트워크

를 3절에서 설명한 방식에 의해서 구성하였다.

본 논문에서 제안하는 SR-SIP 기법의 성능을 비교하기 위해서, 서비스 요청이 있을 때마다 논리적 포그 네트워크를 기반으로 해당 서비스 이미지를 동적으로 배치해주는 On-Demand 기법을 설계 및 구현하였다. On-Demand 기법에서는 서비스 요청을 받은 포그 노드가 논리적 포그 네트워크 상에서 자신의 2홉 이내 포그 노드들이 서비스 이미지를 갖고 있는지 확인해서 없으면, 자신의 조상 포그 노드들이 서비스 이미지를 갖고 있는지 검사한다. 만일 조상 포그 노드에도 없으면 해당 서비스 이미지를 자신에게 배치한다. 만일 자신의 가용 자원이 부족하면, 자신의 2홉 이내 포그 노드 중 가까운 것부터 차례대로 할당 공간이 가능한 포그 노드에게 배치한다. 만일 2홉 이내 이웃 포그 노드에도 배치할 수 없으면, 조상 포그 노드들 중에서 자신으로부터 가까운 것부터 차례대로 확인 후 배치한다. SR-SIP 기법을 On-Demand 기법과 비교함으로써 서비스 요청이 있을 때마다 동적으로 서비스 이미지를 배치해주는 On-Demand 방식의 포그 노드 자원 활용 측면에서의 성능에 대해서 살펴보고, 다수의 서비스 요청을 고려한 서비스 이미지 배치 기법의 필요성을 확인하고자 한다.

성능 평가를 위해서 클라우드 센터의 서비스 이미지 할당 공간의 최대 크기를 300, 500, 750(하나의 서비스 요청은 포그 노드의 서비스 이미지 할당 공간 1을 사용한다고 가정한다)으로 변경하면서 배치된 서비스 이미지 수, 수용되지 못한(미수용) 서비스 요청 수, 네트워크 비용을 측정하였다. 이때 네트워크 비용은 서비스 제공에 소요되는 물리적 거리에 해당하며 지연시간에 비례한다. 서비스 유형은 총 1,000개이며, 총 서비스 요청 수는 42,210이며, 서비스 요청의 분포 형식은 대략 10% 수준의 주요 서비스들에 대한 요청이 많고 나머지 90% 서비스들에 대한 요청은 적은 롱테일(long-tailed) 분포 형식을 갖도록 하였다.

Fig. 6~8은 클라우드 센터의 서비스 이미지 할당 공간의 최대 크기에 따른 SR-SIP 기법과 On-Demand 기법의 성능을 보여준다. Fig. 6에서 클라우드 센터의 할당 공간 크기가 늘어나면서 두 기법 모두 클라우드 센터에 배치되는 서비스 이미지가 많아져서 총 서비스 이미지 배치 수가 감소하는 것을 볼 수 있다. 특히 On-Demand 기법의 경우 SR-SIP 기법보다 훨씬 더 많은 서비스 이미지를 배치하며, 클라우드 센터의 할당 공간이 증가해도 서비스 이미지 배치 수가 별로 감소하지 않는다. 이것은 On-Demand 기법의 경우 서비스 요청이 있는 단말 근처의 포그 노드에 서비스 이미지를 많이 배치하는 것을 의미하며, 따라서 On-Demand 기법이 서비스 이미지를 중복 배치할 가능성이 높음을 알 수 있다.

Fig. 7은 클라우드 센터의 할당 공간 크기가 늘어나면서 두 기법 모두 미수용 서비스 요청 수가 감소하는 것을 보여주며, 전반적으로 두 기법 간 차이가 근소한 편이나 클라우드 센터의 할당 공간 크기가 작을수록 SR-SIP 기법이 On-Demand 기법보다 미수용 서비스 요청이 적음을 알 수 있다. 즉, 클라우드 센터 자원이 충분하지 않을 때 SR-SIP 기법이 On-Demand 기법보다 서비스 요청을 수용하는 데 있어서 우수함을 알 수 있다.

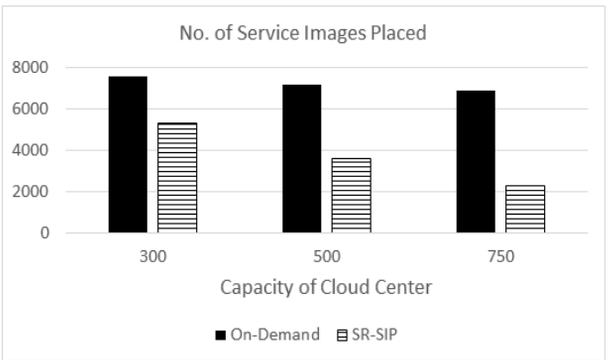

Fig. 6. The Number of Service Image Placements for Various Maximum Resource Capacity of the Cloud Center

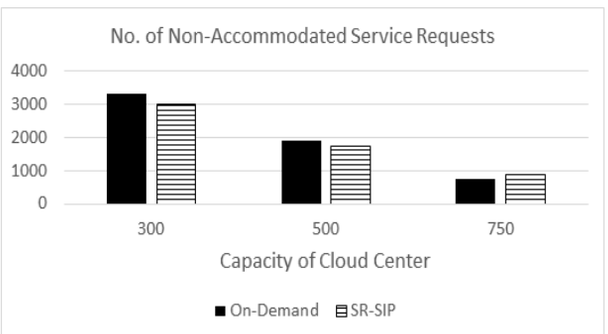

Fig. 7. The Number of Non-accommodated Service Requests for Various Maximum Resource Capacity of the Cloud Center

Fig. 8은 클라우드 센터의 할당 공간 크기에 따른 서비스 요청별 평균 네트워크 비용을 보여준다. 클라우드 센터의 할당 공간 크기가 커지면 SR-SIP 기법의 경우 클라우드 센터에 배치되는 서비스 이미지가 많아져서 네트워크 비용이 증가한다. 반면 On-Demand 기법은 클라우드 센터의 할당 크기가 증가해도 네트워크 비용이 크게 증가하지 않고 모든 경우에 있어서 상대적으로 매우 작게 유지된다. 이것은 On-Demand 기법이 서비스 요청을 하는 단말 노드에 가까운 포그 노드에 서비스 이미지를 배치함으로써 네트워크 비용을 확연하게 줄임을 의미한다.

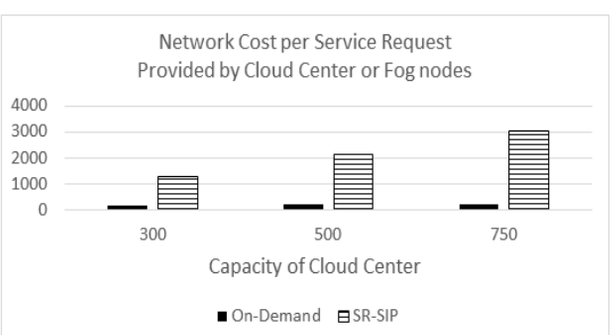

Fig. 8. The Average Network Cost per Service Request for Various Maximum Resource Capacity of the Cloud Center

5. 결 론

본 논문에서는 포그 컴퓨팅 환경에서 서비스 요청 패턴을 고려해서 서비스 이미지의 중복 배치를 최소화함으로써 포그 노드 자원을 효율적으로 사용하는 SR-SIP 기법을 제안했다. 제안 기법의 성능 분석을 위해 단말 노드로부터의 서비스 요청 시 마다 해당 단말로부터 가까운 포그 노드에 서비스 이미지를 배치하는 On-Demand 기법을 설계 및 구현하고, SR-SIP 기법과 성능을 비교했다. SR-SIP 기법이 On-Demand 기법에 비해 서비스 이미지를 덜 배치하고 더 많은 서비스 요청을 수용함을 알 수 있었다. SR-SIP 기법의 경우 클라우드 센터에서 서비스 이미지를 배치할 최적의 포그 노드를 결정하더라도, 서비스 요청을 동적으로 처리하다 보면 서비스 이미지가 최적으로 배치되지 못할 수도 있기 때문에 최적화를 위해 서비스 이미지를 재배치 또는 마이그레이션해야 할 수도 있다.

References

[1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of Mobile Cloud Computing (MCC) Workshop*, 2012.

[2] V. B. C. Souza, W. Ramirez, X. Masip-Bruin, E. Marin-Tordera, G. Ren, and G. Tashakor, "Handling service allocation in combined fog-cloud scenarios," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2016.

[3] A. Brogi and S. Forti, "QoS-aware deployment of IoT applications through the fog," *IEEE Internet of Things Journal*, 2017.

[4] J. Choi and S. Ahn, "Scalable service placement in the fog computing environment for the IoT-based smart city," *Journal of Information Processing Systems*, 2019.

[5] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proceedings of Mobile Big Data (Mobidata) Workshop*, 2015.

[6] O. Skarlat, S. Schulte, M. Borkowski, and P. Leitner, "Resource provisioning for IoT services in the fog," in *Proceedings of IEEE International Conference on Service Oriented Computing and Applications (SOCA)*, 2016.

[7] O. Skarlat, M. Nardelli, S. Schulte, and S. Dustdar, "Towards QoS-aware fog service placement," in *Proceedings of IEEE International Conference on Fog and Edge Computing (ICFEC)*, 2017.

[8] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwalder, "Incremental deployment and migration of geodistributed situation awareness applications in the fog," in *Proceedings of ACM International Conference on Distributed and Event-based Systems (DEBS)*, 2016.

[9] F. Faticanti, F. D. Pellegrini, D. Siracusa, D. Santoro, and S. Cretti, "Cutting throughput with the edge: App-aware placement in fog computing," in *IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, 2019.

[10] R. Yu, G. Xue, and X. Zhang, "Application Provisioning in fog Computing-enabled Internet of Things: A Network Perspective," in *Proceedings of IEEE INFOCOM*, 2018.

[11] V. B. Souza, X. Masip-Bruin, E. Marín-Tordera, S. Sanchez-Lopez, J. Garcia, G.-J. Ren, A. Jukan, A. Jukan, and A. J. Ferrer, "Towards a proper service placement in combined fog to cloud (F2C) architectures," Future Generation Computer Systems, 2018.

[12] 서울 열린데이터광장 [Internet], https://data.seoul.go.kr/

최 종 화

https://orcid.org/0000-0001-7345-6517
e-mail : whdghk414141@naver.com
2017년 서울시립대학교 컴퓨터과학부(학사)
2019년 서울시립대학교 컴퓨터학과(석사)
2020년 ~ 현 재 코인플러그 연구원
관심분야 : 유무선 네트워크, 인터넷 등

안 상 현

https://orcid.org/0000-0001-7640-4480
e-mail : ahn@uos.ac.kr
1986년 서울대학교 컴퓨터공학과(학사)
1988년 서울대학교 컴퓨터공학과(석사)
1993년 미네소타대학교 컴퓨터학과(Ph.D.)
1994년 ~ 1998년 세종대학교 컴퓨터학과 교수
1998년 ~ 현 재 서울시립대학교 컴퓨터과학부 교수
관심분야 : 유무선 네트워크, 인터넷 프로토콜, IoT 등