차량 엣지 컴퓨팅에서 로드 밸런싱을 고려한 강화학습 기반의 마이그레이션

문성원, 임유진
숙명여자대학교 IT 공학과
sungwon268@sookmyung.ac.kr, yujin91@sookmyung.ac.kr

Migration with Load Balancing Based on Reinforcement Learning in Vehicular Edge Computing

Sungwon Moon, Yujin Lim
Dept. of IT Engineering, Sookmyung Women's University

요 약

최근 실시간 응답 및 처리에 민감한 서비스들이 급증하면서 멀티액세스 엣지 컴퓨팅(MEC)이 차세대 기술로 주목받고 있다. 사용자들의 잦은 이동성 때문에 MEC 서버들 사이에서의 마이그레이션은 중요한 문제로 다뤄진다. 본 논문에서는 이동성이 많은 차량 엣지 컴퓨팅 환경을 고려하였으며, 강화학습 기법인 Q-learning 을 사용하여 마이그레이션 여부 및 대상을 결정하는 기법을 제안하였다. 제안 기법의 목적은 지연 제약조건을 만족시키면서 차량 엣지 컴퓨팅 서버(VECS) 사이의 로드 밸런싱을 최적화하는 것이다. 제안 기법의 성능 비교를 통하여 다른 기법들보다 로드 밸런싱 측면에서 약 22-30%, 지연 제약조건 만족도 측면에서 약 20-31%로 더 좋은 성능을 보임을 확인하였다.

1. 서론

최근 사물인터넷의 확산에 따라 실시간 응답 및 처리에 민감한 서비스들, 예를 들어 자율주행, AR, VR 과 같은 서비스들이 급증하고 있다. 이러한 서비스들은 많은 양의 컴퓨팅 자원과 높은 QoS(Quality of Service)를 요구한다. 그러나 데이터 처리와 연산이 중앙에 집중된 기존 클라우드 컴퓨팅(cloud computing)은 기하급수적으로 늘어나는 데이터들을 전송하고 처리하는 응답 시간이 길어진다. 또한, 한꺼번에 방대한 양의 작업이 집중되면 정체 현상이 발생할 수 있고 이로 인하여 연산 결과를 빠르게 받기 어려울 수 있다. 그래서 사람 및 사물과 근접한 네트워크 종단에 컴퓨팅 자원을 분산 배치하는 멀티액세스 엣지 컴퓨팅(Multi-access Edge Computing, MEC) 컴퓨팅이 차세대 기술로 주목받고 있다[1]. 서비스 근접한 곳에서 바로 데이터를 처리하고, 빠른 연산 결과를 받을 수 있을 뿐만 아니라 네트워크 대역폭에 대한 부담을 줄여 기존의 클라우드 컴퓨팅의 단점을 보완하기 때문이다.

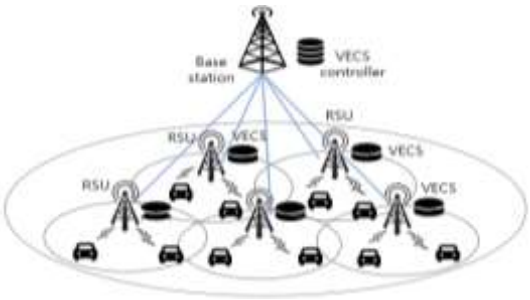
하지만, MEC 환경에서 해결해야 할 하나의 큰 과제는 사용자들의 이동성이다. 제한적인 커버리지를 갖는 MEC 서버들 사이에서 사용자의 높은 이동성은 서비스 중단과 QoS 저하를 초래한다. 또한, 사용자들의 높은 이동성으로 인해 특정 MEC 만 혼잡해지는 경우가 발생할 수 있으며, 이는 MEC 서버들의 로드들이 불균형해질 수 있다. MEC 서버사이의 로드 불균형은 MEC 시스템의 처리량이 낮아질 뿐만 아니라 지연 제약조건에 충족하지 못할 수도 있다. 이러한 문제는 MEC 서버들간 로드 밸런싱(load balancing)을 통해 특정 MEC 서버들만 혼잡해지는 경우를 줄임으로써 시스템의 처리량과 서비스의 QoS 만족도를 높일 수 있다. 그래서 사용자의 서비스를 이동성에 맞게 MEC 서버간 마이그레이션(migration) 진행 여부와 대상을 결정하는 서비스 마이그레이션을 실현하는 것이 중요하다.

기존 마이그레이션 연구는 실시간 서비스를 위해 QoS 기반의 마이그레이션 여부를 결정하는 기법[2]과 지연시간, 에너지 그리고 네트워크 대역폭을 고려한 비용을 최소화하는 마이그레이션 여부 및 대상을 결정하는 기법[3]이 제안되었다. 그리고 시스템 유틸리티를 최대화하기 위해 모바일 디바이스와 MEC 서버의 로드 밸런싱을 고려한 오프로딩(offloading) 기법[4]

과 모바일 디바이스와 MEC 서버에서의 서비스 지연 시간을 이용하여 로드 밸런싱을 고려한 오프로딩 기법[5]이 제안되었다. 이는 모바일 디바이스와 MEC 서버 사이의 로드 밸런싱만 고려하고, MEC 서버들 사이는 고려하지 않았다는 한계점이 있다.

본 연구에서는 차량 엣지 컴퓨팅(Vehicular Edge Computing, VEC) 환경에서 차량의 이동에 따라 강화학습 기법인 Q-learning 을 적용한 마이그레이션 기법을 제안한다. 본 기법은 VECS(VEC Server)들의 로드 밸런싱을 고려하여 서비스의 QoS 만족도와 시스템의 처리량을 높이고자 제안하였다.

2. 시스템 모델



(그림 1) 차량 엣지 컴퓨팅 모델

본 논문에서 제안된 시스템은 그림 1 처럼 M개의 VECS 가 각 RSU 에 배치된 환경이며, 이는 하나의 클러스터를 구성하고 있다고 가정한다. N개의 차량들이 주행하고 있으며, 차량에서 발생된 태스크는 인접한 VECS 에 오프로딩한다. 차량에서 발생한 태스크 $s_n = \{I_n, A_n, T_n^{max}\}$으로 정의되며, I_n는 태스크의 크기, A_n는 태스크를 연산하기 위해 요구되는 컴퓨팅 자원이고, T_n^{max}는 태스크의 최대 허용 지연시간이다. VECS 들은 서로 백홀(backhaul) 채널을 통해 연결되어 있고, 각 VECS 들은 VECS 컨트롤러와 연결되어 있다. VECS 컨드롤러는 차량과 VECS 들에 대한 정보를 수집하며, 마이그레이션에 관한 결정을 한다. 마이그레이션 결정 시점은 기존의 핸드오버 기법을 사용하여, 차량이 인접한 VECS 들간 서비스 커버리지가 중첩되어 있는 곳에 위치하게 될 때 이뤄진다. VECS 들의 로드 밸런싱을 고려하여, 현재 태스크를 수행하고 있는 VECS 에서 새로운 VECS 로 마이그레이션을 진행할지 그리고 진행한다면 클러스터안에 어떤 VECS 로 할지 결정한다.

차량에서 태스크가 발생하였다면, 차량의 위치에서 지리적으로 가장 가까운 VECS 로 오프로딩을 한다. 이때, 차량 $n \in N$ 의 태스크가 VECS m로 오프로딩 할 때 전송 지연시간은 다음과 같다.

$$T_n^{trans} = \frac{I_n}{R_{n,m}} \tag{1}$$

연산 결과는 태스크의 크기보다 작아 다운 지연시간은 고려하지 않는다. 차량 n과 VECS m 사이의 전송 속도인 $R_{n,m}$은 다음과 같다.

$$R_{n,m} = B_n \log_2(1 + \frac{P_n \cdot H_{n,m}}{\sigma^2}) \tag{2}$$

B_n는 전송 대역폭, P_n는 차량의 송신전력 그리고 σ^2는 잡음 이득이다. $H_{n,m}$는 차량과 VECS 사이의 채널 이득이다.

마이그레이션 지연시간은 백홀 채널을 통해 VECS $m \in M$에서 $m' \in M$로 전송된 시간이며 다음과 같다.

$$T_n^{mig} = \begin{cases} 0, & if\ \ vecs_m = vecs_{m'} \\ \frac{I_n}{R_b}, & if\ \ vecs_m \neq vecs_{m'} \end{cases} \tag{3}$$

R_b는 백홀 채널의 전송 속도이다. VECS m에서 태스크를 수행하는 연산 지연시간은 다음과 같다.

$$T_n^{comp} = \frac{A_n}{f_{n,m}} \tag{4}$$

$f_{n,m}$는 VECS m에서 차량 n에 할당된 컴퓨팅 자원이다. 따라서 태스크의 총 지연시간은 다음과 같다.

$$T_n^{total} = T_n^{trans} + T_n^{mig} + T_n^{comp} \tag{5}$$

본 연구에서의 목표는 VECS 들 사이의 로드 밸런싱을 최적화하는 것이다. 각 VECS 의 로드는 VECS 에서 수행되고 있는 태스크들의 워크로드의 합과 같다.

$$\zeta_m = \sum_{m \in M} \sum_{n \in N} \theta_{n,m} \gamma_n \tag{6}$$

태스크의 워크로드는 지연 제약 조건안에 태스크를 완료하기 위해 필요한 연산 작업량이다. 이미 오프로딩된 상황이므로, 지연 제약조건에서 전송 지연시간을 제외한 시간으로 태스크의 워크로드를 측정하였다.

$$\gamma_n = \frac{A_n}{T_n^{max} - T_n^{trans}} \tag{7}$$

$\theta_{n,m}$는 태스크가 현재 $vecs_m$에서 수행되고 있는지를 나타내는 변수이다.

$$\theta_{n,m} = \begin{cases} 1, & if\ \ s_n\ is\ running\ on\ vecs_m \\ 0, & otherwise \end{cases} \tag{8}$$

VECS 들의 로드의 전체 평균은 다음과 같다.

$$\zeta = \frac{1}{M} \sum_{m=1}^{M} \zeta_m \tag{9}$$

VECS 들 사이에 로드 밸런싱이 얼마나 잘 되었는지 측정하기 위해 각 VECS 들의 로드 편차의 평균으로 측정한다.

$$Load_{diff} = \frac{1}{M} \sum_{m=1}^{M} |\zeta_m - \zeta| \tag{10}$$

본 논문에서 제안한 기법의 목표는 차량들이 발생한 태스크들의 지연 제약조건을 만족시키면서 VECS 들간 로드 밸런싱를 최적화함으로써 VEC 시스템 처리량을 최대화하는 것이다. 이를 식으로 표현하면 다음과 같다.

$$min. \quad Load_{diff} \tag{11}$$

$$s.t. \quad T_n^{total} \leq T_n^{max},\ n \in N \tag{11a}$$

제약조건(11a)는 태스크 총 지연시간이 태스크의 최대 허용 지연시간을 초과해서는 안된다는 것을 의미한다. 차량과 VECS 의 개수가 증가할수록 VECS 간

로드 편차가 점점 커진다는 문제점이 있다. 기존의 최적화 기법 대신, 본 논문은 강화학습 기법인 Q-learning 을 사용하여 이 문제를 해결하고자 한다.

3. 제안하는 알고리즘

본 논문은 Q-learning 을 사용하여 마이그레이션 여부와 대상을 결정하는 알고리즘을 제안한다. VECS 컨트롤러는 모든 차량과 VECS 정보들을 관찰하여 최적의 로드 밸런싱을 위한 마이그레이션을 결정한다.

Q-learning 은 상태(State), 행동(Action) 그리고 보상(Reward)로 구성되어 있으며, Q 함수(Q function)을 사용하여 최적의 정책을 학습한다. 최적의 정책을 찾기 위해 Q 테이블(Q-table)에 저장된 Q 값(Q value)이 가장 큰 행동을 선택하는 Q 함수는 다음과 같다.

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \cdot \max_{a} Q(s',a') - Q(s,a)] \quad (12)$$

현재 상태 s, 행동 a, 보상 r 그리고 새로운 상태 s' 와 행동 a' 에 대한 최댓값을 구한다. 학습율(Learning rate)는 α 이고, 할인율(Discount factor)은 Υ 이다. Q-learning 은 다음 표 1 과 같고, ϵ-탐욕적 알고리즘을 사용하여 가끔 보상에 따라 탐색하는 것이 아닌 무작위성을 가진 탐사를 하는 방법을 사용하였다.

Q-learning 에서 상태 s, 행동 a 그리고 보상 r의 정의는 다음과 같다. VECS 들의 로드 밸런싱을 최적화하기 위해 차량이 발생한 태스크 정보와 해당 태스크를 수행중인 VECS 에 대한 정보를 반영해야 한다. 따라서 상태 $s = \{I_n, A_n, T_n^{max}, vecs_m\}$으로 표현 가능하다.

VECS 컨트롤러는 현재 상태 s를 기반으로 행동 a을 선택하고, 선택한 행동을 취한다. 행동은 마이그레이션 여부와 진행 대상을 결정할 수 있도록 정의하였다. 행동 $a = \{vecs_1, vecs_2, \dots, vecs_m\}$으로 표현 가능하다.

VECS 컨트롤러는 행동 a을 취함으로써 보상 r을 얻는다. 본 논문에서 제안한 기법의 목표인 VECS 들의 로드 밸런싱을 최적화하기 위해 식 (11)처럼 로드 편차의 평균을 최소화하는 것이다. 따라서 부정적인 보상으로 $r = Load_{diff}$으로 표현이 가능하다.

<표 1> Q-learning 기법

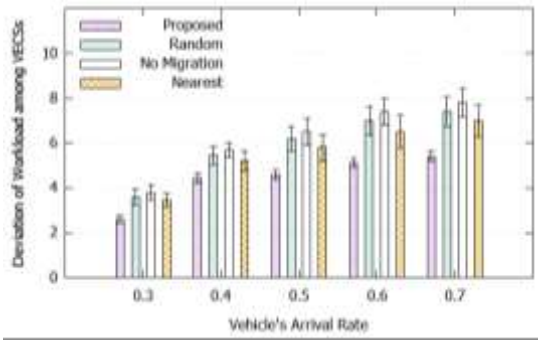
Algorithm 1. Q-learning method

Initialize $Q(s,a)$
for each episode **do**
  Initialize initial state s_0, reward r_0
  For time slot $t = 1$ to $t = T - 1$ **do**
    The controller acquires information about vehicles' tasks and VECSs by interacting with the environment
    If the random number < ϵ:
      $a_t = argmax_a Q(s_t, a_t)$
    else:
      Randomly select action a_t
    Execute a_t at VECS controller, observe r_t and s_{t+1}
    Compute the Q value according to (12)
    Agent obtains an optimal policy
    Terminate when all the vehicles are out of simulation region
  end for
end for

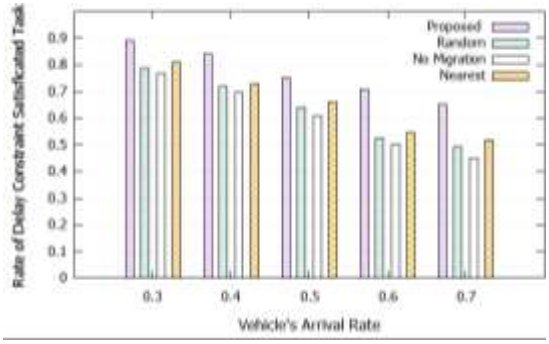
4. 실험 및 결과 분석

제안한 마이그레이션 알고리즘에 대한 성능을 측정하기 위해 실험을 진행했다. 총 5 개의 VECS 가 RSU 에 배치된 환경을 고려하였다. VECS 의 반경은 $250m$ 이고, 각 VECS 의 컴퓨팅 능력은 10 GHz이다. 차량은 푸아송 분포를 사용하여 발생시켰다. 태스크의 크기 I_n 과 이에 요구되는 CPU 사이클 A_n 은 각각 [50, 75] MB, [0.2,0.3] $Gcycles/MB$ 범위 내에서 무작위로 선택하였다. 그리고 각 차량은 하나의 태스크를 발생시킨다. 차량과 VECS 사이의 경로 손실은 $140.7 + 36.7\log_{10}(distance(km))$ 이고, B_n 은 10 MHZ R_b 는 20MHZ 이다. 차량의 송신전력 P_n은 1 W 이고, 잡음 이득 σ^2은 $10^{-11}mW$이다. Q-learning 을 위해 α는 0.01, γ 는 0.9, ϵ는 0.9 로 설정하였다. 본 실험은 샌프란시스코의 이동성 데이터를 사용하였다[6].

본 논문에서 제안한 Q-learning 기반의 마이그레이션 기법인 *Proposed* 의 성능 비교를 위하여 *Random, No Migration* 그리고 *Nearest* 총 3 가지 기법과 비교하였다. *Random* 은 차량이 이동할 때마다 VECS 를 랜덤으로 선정하여 마이그레이션을 진행한다. *No Migration* 은 태스크가 완료될 때까지 마이그레이션을 진행하지 않는 기법이다. *Nearest* 은 차량이 이동할 때마다 지리적으로 가장 인접한 VECS 로 마이그레이션을 진행한다.



(그림 2) 차량 도착율에 따른 VECS 의 워크로드 편차 비교



(그림 3) 차량 도착율에 따른 지연 제약조건 만족도 비교

그림 2 은 차량의 도착율에 따라 VECS 사이의 로드 편차를 이용하여 로드 밸런싱을 비교한 실험이다. 차량의 도착율이 증가할수록 전체적으로 로드의 편차가 커지는 것을 볼 수 있다. 차량이 많아질수록 특정 VECS 로 치우치는 경향이 강하여 로드 밸런싱이 깨지기 때문이다. 하지만 본 논문에서 제안한 알고리즘은 비교적 차량이 많아져도 큰 차이가 없는 것을 볼 수 있다. *Proposed* 알고리즘을 *Random, No Migration* 그리고 *Nearest* 기법과 비교하였을 때 약 27%, 30% 그리고 22%만큼 차이가 난다. 그리고 마이그레이션을 진행하는 기법이 진행하지 않는 기법보다 특정 VECS 로 치우치는 경향이 덜하여 로드 밸런싱이 더 잘되는 것을 확인할 수 있다.

그림 3 은 차량의 도착율에 따른 태스크 지연제약 만족도이다. 만족도는 지연제약조건을 만족한 태스크 개수를 전체 완료 태스크 개수로 나누어 계산하였다. 차량이 많아질수록 특정 VECS 만 혼잡해지는 경우가 있어 로드의 불균형이 생기고, 이에 서비스 거절(blocking)되는 태스크들이 많아지면서 지연 제약조건을 만족하는 태스크의 비율이 낮아진다. 로드 밸런싱을 고려한 *Proposed* 기법은 고려하지 않은 *Random, No Migration* 그리고 *Nearest* 와 비교했을 때 약 25%, 31%, 20% 정도 나은 성능을 보인다. 또한, 로드 밸런싱이 잘 이뤄지지 않았던 *No Migration* 기법이 제일 낮은 만족도를 갖는 것을 확인할 수 있다.

5. 결론

본 논문에서는 차량 엣지 컴퓨팅 환경에서 Q-learning 을 이용하여 마이그레이션 여부 및 대상을 결정하는 기법을 제안하였다. 제안한 기법은 태스크의 지연 제약조건을 만족시키면서 VECS 사이의 로드 밸런싱를 최적화함으로써 VEC 시스템 처리량을 최대화하는 것이다. 실험을 통하여 VECS 사이의 로드 밸런싱와 지연 제약조건 만족도가 다른 기법들에 비해 우수한 성능을 보임을 확인하였다. 하지만 본 논문은 멀티 유저 환경에서 싱글 에이전트를 고려한 상황이다. 더 동적이며 복잡해지는 환경에 있어 효율적으로 학습하기 위해 멀티 에이전트를 고려한 연구가 필요하다.

사사문구

이 성과는 2018 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2018R1A2B6002505)

참고문헌

[1] S. Wang, J. Xu, N. Zhang and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," IEEE Access, vol. 6, pp. 23511-23528, Apr. 2018.

[2] J. Li, X. Shen, L. Chen, D. Pham, J. Ou, L. Wosinska and J. Chen, "Service Migration in Fog Computing Enabled Cellular Networks to Support Real-Time Vehicular Communications," IEEE Access, vol. 7, pp. 13704-13714, 2019.

[3] S. Maheshwari, S. Choudhury, I. Seskar and D. Raychaudhuri, "Traffic-Aware Dynamic Container Migration for Real-Time Support in Mobile Edge Clouds," 2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Indore, India, pp. 1-6, 2018.

[4] J. Zhang, H. Guo, J. Liu and Y. Zhang, "Task Offloading in Vehicular Edge Computing Networks: A Load Balancing Solution," IEEE Transactions on Vehicular Technology, vol. 69, no. 2, pp. 2092–2104, 2020.

[5] Y. Dai, D. Xu, S. Maharjan and Y. Zhang, "Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks," IEEE Internet of Things Journal, vol. 6, no. 3, pp. 4377–4387, 2019.

[6] M. Piorkowski, N. Sarafijanovic-Djukic and M. Grossglauser, CRAWDAD DataSet Epfl/Mobility, Feb. 2009, Available: http://crawdad.org/epfl/mobility.