셀룰러 네트워크에서 딜레이드 오프로딩 스키마를 적용한 사용자 이동성 고려 캐싱 기법

최윤정*, 임유진*
*숙명여자대학교 IT 공학과
potatochips@sookmyung.ac.kr, yujin91@sookmyung.ac.kr

Caching Strategy Adopting Delayed Offloading Scheme with User Mobility in Cellular Network

Yoonjeong Choi*, Yujin Lim*
*Dept. of IT Engineering, Sookmyung Women's University

요 약

비디오 컨텐츠 사용이 증가하면서, 사용자가 요구한 파일을 제시간 안에 전달하는 문제가 중요해졌다. 사용자와 가까운 곳에 파일을 캐싱해 두고 필요할 때 다운받으면 파일을 보다 빨리 전달할 수 있는데 사용자가 움직일 경우 이동성을 고려해야 한다. 본 논문에서는 사용자의 이동 경로와 파일의 인기도를 함께 고려해 딜레이드 오프로딩(delayed offloading) 스키마를 적용한 환경에서 마이크로 기지국(micro base station, MBS)에서 다운받는 데이터 크기를 최소로 만들어 비용을 최소화 하는 캐싱 기법을 제안한다. 실험을 통해 타알고리즘에 비해 MBS 로부터 다운받는 양을 줄이고 스몰 셀 기지국(small cell base station, SBS)에서 다운받을 성공 확률을 높이는데 효과가 있다는 것을 보였다.

1. 서론

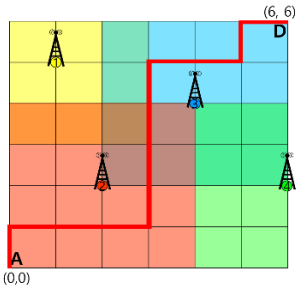
전 세계적으로 모바일 기기의 사용이 증가하면서 모바일 데이터 트래픽이 증가하고 있다. 모바일 데이터 트래픽은 2017 년부터 2022 년까지 연평균 성장률이 46%를 육박할 것으로 예상하고 있다. 특히 2022 년 모바일 데이터 트래픽 77EB/month 중에서 비디오에 대한 양이 61EB/month 으로 예상되는 만큼, 비디오에 대한 수요가 폭발적으로 늘 것으로 예측하고 있다[1]. 많은 사용자에게 빠른 속도로 컨텐츠를 전달해 주는 것이 중요해지면서 사용자의 이동성을 고려해 가장 가까운 네트워크에 미리 컨텐츠를 캐싱해 두는 방법들이 주목받기 시작했다. [2]는 사용자의 이동 경로를 미리 아는 상황에서, deep deterministic policy gradient 을 사용해 road side unit (RSU)와 MBS 로부터 다운받는 비용과 파일 업데이트 비용 모두를 최소화하는 캐싱 정책을 제안했다. [3]은 연결된 차량(connected vehicles)의 끊김 없는 컨텐츠 다운로드를 위해 이동에 관한 개인정보의 유출 없는 캐싱 기법을 제안했다. [4]는 셀룰러 네트워크 환경에서 파일의 인기도와 이동 경로를 알았을 때 MBS 로부터 파일을 최소로 다운받도록 하는 coded caching 을 제안했다.

SBS 와 MBS 로 이루어진 셀룰러 네트워크에서 요구한 파일을 주어진 시간 안에 SBS 로부터 모두 다운받지 못 할 경우 MBS 로부터 다운받는 딜레이드 오프로딩 스키마를 적용할 때, 사용자가 요구한 컨텐츠를 보다 낮은 비용으로 다운받기 위해 MBS 보다 SBS 로부터 가져오는 것이 효과적이다[4]. 따라서 본 논문은 이동하는 사용자가 요구한 파일을 SBS 가 아닌 MBS 로부터 다운받는 크기를 줄이기 위한 캐싱 기법을 제안하고자 한다.

2. 제안기법

2.1 시스템 모델

하나의 MBS 와 다수의 SBS 가 이종 네트워크를 구성하며 SBS 의 범위는 겹칠 수 있다고 가정한다. SBS 의 범위는 반경 100m 로, 사용자의 이동성은 사람의 걷는 속도를 기반으로 1.25m/s 로 설정했다. 그림 1 은 SBS 가 설치된 공간을 나타낸다.



(그림 1) SBS 설치 및 사용자 이동 공간

사용자는 A 지점에서 D 지점까지 이동하면서 각 위치에 해당하는 SBS 에 캐싱된 파일을 다운받을 수 있다. 한번 지나간 곳은 다시 되돌아 가지 않으며 오른쪽 혹은 위쪽 방향으로만 이동한다고 가정했다. 예를 들면 사용자의 이동 경로는 그림에 굵게 표시된 붉은 선과 같다. A 지점 (0,0)에서 D 지점 (6,6)까지 가는 모든 경로에 대한 확률과 캐싱할 파일에 대한 인기도는 사전에 알 수 있다고 가정한다. 경로는 좌표의 배열로 나타낸다.

2.2 문제 정의

본 논문에서는 사용자가 요구한 컨텐츠에 대하여 A 에서 출발해 D 에 도착할 때까지 발생하는 다운로드 비용을 낮추기 위해, MBS 로부터 다운로드하는 데이터 양을 최소화하는 것을 목표로 한다. SBS 보다 MBS 로부터 다운받을 때 더 많은 비용이 들기 때문이다.

파일의 크기를 B 라 하고 모든 파일은 크기가 같다고 가정한다. 전체 파일은 K 개로 인기도 순으로 정렬돼 있고, 파일의 인기도를 P_k, SBS 개수를 N, SBS 용량을 C_n, SBS 의 전송 속도를 R_n, 사용자의 출발지부터 목적지까지 가능한 모든 경로 수를 M, 각 경로의 확률을 q_m, 주어진 시간을 T, 알고리즘을 통해 찾고자 하는 캐싱 기법을 X 라 하면 MBS 로부터 다운받는 평균 파일 크기 d_{av}는 다음 식과 같이 표현 가능하다[4].

$$d_{av} = \sum_{m=1}^{M}\sum_{k=1}^{K} q_m p_k d_{k,m} \quad (1)$$

$$d_{k,m} = \max\{B - (\sum_{n=1}^{N} min\{x_{n,k}, R_n S_{m,n}\}), 0\} \quad (2)$$

$x_{n,k}$는 SBS n 에 파일 k 를 얼마만큼 저장할지 나타내고 $S_{m,n}$는 경로 m 이 SBS n 에 머무르는 총 시간을 의미한다. 따라서 MBS 로부터 다운로드 받는 양을 최소화하는 문제는 다음 식으로 정의할 수 있다[4].

$$\mathbf{P1}: \min_{X_T} d_{av}$$

$$\text{subject to: } \sum_{k=1}^{K} x_{n,k} \le C_n, \ \forall n. \quad (3)$$

$$x_{n,k} \ge 0, \ \forall n,k. \quad (4)$$

2.3 알고리즘

알고리즘은 크게 두 단계로 나뉜다. 첫번째 단계는 각 SBS 의 서비스 범위가 겹치지 않는다고 가정한 상태에서 캐싱을 하는 것이고 두번째 단계는 SBS 범위가 겹치는 것을 고려해 캐싱된 파일을 재배치 해주는 것이다. 첫번째 단계에서 전송 속도 R_n 으로 파일을 다운받을 때, 한 파일의 크기인 B 를 모두 다운받을 수 있는 가장 짧은 시간을 T_{min} 이라 하면 T_{min} 이내에 SBS 에서 다운받는 파일 k 의 크기 $\tilde{d}_{av,n}^{k}$는 다음과 같다[4].

$$\tilde{d}_{av,n}^{k} = \sum_{t=1}^{T}\sum_{m:S_{m,n}\ge t} p_k q_m \ max\{min\{x_{n,k} - (t-1)R_n, R_n\}, 0\} \quad (5)$$

MBS 에서 최소로 다운받는 것은 SBS 에서 최대로 다운받는 것과 같은 의미이기 때문에 위 식의 최댓값을 구하면 된다. $\sum_{m:S_{m,n}\ge t} p_k q_m$ 이 커질수록 $\tilde{d}_{av,n}^{k}$이 같이 증가하기 때문에 최대로 만들기 위해서 항상 가장 큰 $\sum_{m:S_{m,n}\ge t} p_k q_m$ 인 파일 k 를 R_n 만큼 SBS 에 저장하면 된다. 하지만 이 방법은 전체 시간 T 가 T_{min} 보다 작거나 같을 때만 $\tilde{d}_{av,n}^{k}$을 최적으로 만들기 때문에 전체 시간 T 가 T_{min} 보다 크다면 최적의 방법이 아닐 수 있다. 따라서 추가적인 요소를 고려했다. 경로 확률 q_m 보다 파일 인기도의 중요도를 높이기 위해 파일 k 의 인기도 p_k 에 α 만큼의 가중치를 더해 $\sum_{m:S_{m,n}\ge t}(p_k+\alpha)q_m$ 이 가장 큰 파일 k 를 선택하도록 했다. 인기있는 파일 위주로만 캐싱했을 경우 사용자가 요구한 파일을 B 만큼 모두 다운받을 가능성이 낮아지기 때문에 τ 확률로 랜덤하게 파일을 선택하고 $(1-\tau)$ 확률로 가장 큰 $\sum_{m:S_{m,n}\ge t}(p_k+\alpha)q_m$ 인 파일 k 를 선택해 R_n 크기만큼 캐싱하도록 한다.

두번째 단계에서 SBS 의 겹치는 부분을 고려해 캐싱할 파일을 재배치 하는데, 이를 결정하기 위해서 각 SBS 를 지날 확률을 사용했다. 인기도가 높은 파일은 어디서나 요구를 많이 하는 반면, 인기도가 낮은 파일은 몇몇의 특정한 장소에서만 요구하는 경향이 있다[5]. 범위가 겹치는 SBS 사이에 인기도가 낮은 동일한 파일이 중복적으로 캐싱됐을 경우 더 많은 모바일 사용자가 지나가는 SBS 에 해당 파일을

제거하고 동일한 크기만큼 인기가 높은 파일을 캐싱한다. 여기서 파일 인기도의 기준은 클러스터 H 로 한다. 클러스터 H 는 각 SBS 에 인기도가 높은 순으로 파일을 정렬했을 때 연속적으로 캐싱된 파일들을 의미한다. 이를 기준으로 H 에 속한 파일을 인기가 있는 파일, H 에 속하지 않는 파일은 비교적 인기가 없는 파일이라고 판단한다. 범위가 겹치는 SBS 에서 H 에 속하지 않은 파일이 동일하게 캐싱되어 있다면, 더 많은 사용자가 지나가는 SBS 에서 해당 파일을 제거하고 H 에 속하지 않으면서 가장 인기도가 높은 파일을 캐싱한다. 이렇게 캐싱을 재조정 했을 때 d_{av} 의 크기가 줄지 않는다면 다시 원래 상태로 되돌아가고, 감소했을 경우 변경된 캐싱으로 진행한다.

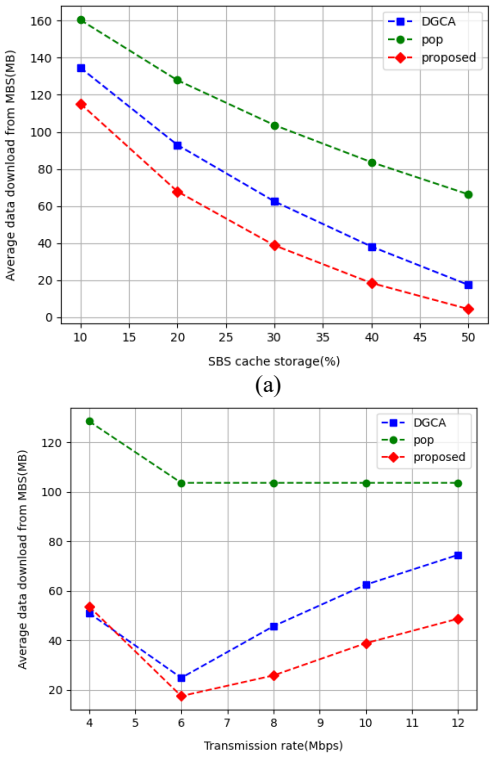
3. 성능평가

제시한 알고리즘과 비교할 알고리즘은 [4]를 참고했다. [4]에서 제시한 분산 탐욕 캐시 할당 알고리즘(Distributed Greedy Cache Allocation, DGCA)은 이동 경로와 파일의 인기를 고려하되 가중치를 두지 않고 캐싱한 후 캐싱된 크기가 동일한 파일끼리 그룹을 나눠 각 그룹에서 캐싱된 파일을 R_n 만큼 늘리거나 줄였을 때 d_{av} 가 감소한다면 캐싱된 파일을 재배치 해주는 방법을 사용한다. 이 외에도 가장 인기있는 파일부터 차례로 캐싱하는 방법(popularity first, pop)과 함께 비교했다.

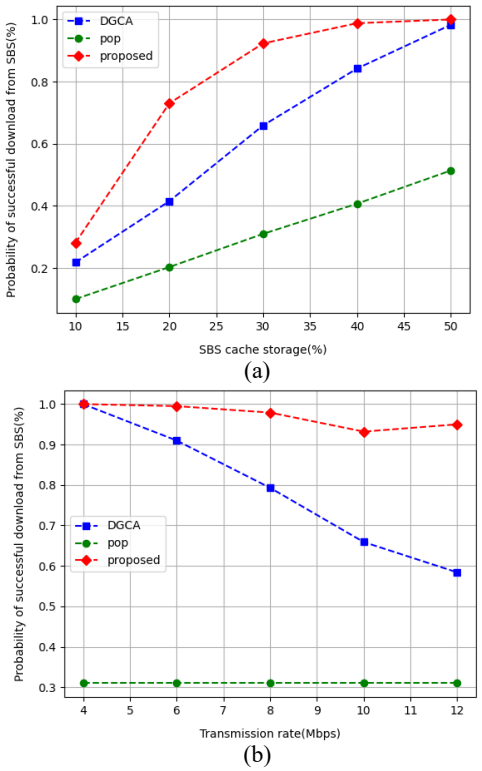
시간단위(time slot)는 20 초로 설정해 사용자가 이동 공간의 한 칸을 움직였을 경우 25m 를 이동한다고 가정했다. A 에서 D 까지 이동하는 최소 시간은 13 time slot 으로 제한한다. 사용자가 오른쪽으로 이동할 확률은 0.7, 위로 갈 확률이 0.3 이라 가정하고 모든 이동 경로의 확률을 계산했다. 파일의 인기도는 파라미터가 0.56 인 지프 분포(zipf distribution)를 따른다고 설정했다. 실험은 그림 1 과 같이 4 개의 SBS 가 반경 100m 의 범위를 가지고 있을 때 사용자가 요청한 파일을 출발 지점 A 에서 도착 지점 D 까지 이동하면서 13 time slot 동안 다운받는 시나리오다. 실험에 사용한 파라미터들은 표 1 에 정리했다.

parameter	value	**parameter**	value
Number of files (K)	500	**SBS cache storage(C_n)**	[10, 50]%
File size(B)	240 MB	**α**	0.5
Transmission rate(R_n)	[4, 12]Mbps	**τ**	0.05
One time slot	20s	**request**	1000

<표 1> 실험에 사용한 파라미터 값



(그림 2) MBS 로부터 다운받은 평균 데이터 크기



(그림 3) SBS 로 모두 다운받기 성공한 비율

그림 2 는 SBS 용량과 전송 속도에 따라 MBS 로부터 다운받는 데이터 양을 표시했다. (a)는 SBS 용량을 캐싱 대상 파일 전체 크기 K×B 의 10%, 20%, 30%, 40%, 50%로 바꾸면서 실험한 결과다. 제안한 알고리즘이 MBS 로부터 다운받는 데이터 크기가 가장 작은데 그 이유는 DGCA, pop 에 비해 우선순위가 높은 파

일에 몰려있기 보단 캐싱된 파일이 골고루 분포하기 때문이다. (b)는 전송 속도를 4Mbps 에서 12Mbps 로 2Mbps 씩 증가하면서 측정한 결과다. 전송 속도가 작을 때는 DGCA 와 크게 차이 나지 않는데 그 이유는 제시한 알고리즘과 DGCA 모두 여러 파일을 조금씩 저장하기 때문이다. 반면에 전송 속도가 커지면 DGCA 는 특정 파일에 캐싱이 집중되고, 제시한 알고리즘은 몰리는 부분이 있더라도 비교적 고루 캐싱이 분포돼 있기 때문에 MBS 로부터 다운받아야 할 평균 데이터 크기의 차이가 벌어진다.

그림 3 은 사용자가 요구한 파일 크기 B 를 제한 시간 동안 모두 다운받았는지 나타내는 성공 비율을 1000 번의 요구에 대해서 계산한 결과다. (a)는 SBS 용량을 전체 파일 용량의 10%, 20%, 30%, 40%, 50%로 변경하며 실험을 진행했다. 제안한 알고리즘이 다른 방법에 비해 높게 나온 이유는 파일이 작은 크기로 비교적 여러 SBS 에 흩어져 있어 경로를 지나가면서 요구한 파일 크기 전체를 다운받을 확률이 높아지기 때문이다. 다른 두 알고리즘은 비교적 큰 용량으로 인기있는 파일에 대해서만 SBS 에 캐싱되어 있기 때문에 상대적으로 낮은 값을 보인다. (b)는 전송 속도를 4Mbps 에서 12Mbps 로 2Mbps 씩 증가하면서 실험한 결과이다. 제안한 알고리즘과 DGCA 모두 전송 속도가 증가하면 성공 비율이 떨어지는 것을 볼 수 있는데 이는 저장 단위가 커지면서 인기도가 높은 파일의 저장 용량이 커지는 반면, 다른 파일의 용량이 줄어들어 결국 성공률이 감소하게 되기 때문이다.

4. 결론

본 논문에서는 셀룰러 네트워크 환경에서 사용이 증가하는 모바일 비디오 컨텐츠를 사용자에게 낮은 비용으로 전달해주기 위한 캐싱 기법을 제안했다. 딜레이드 오프로딩 스키마를 적용한 상황에서 MBS 로부터 다운받는 평균 데이터 양의 최소화를 목표로 알고리즘을 설계했다. 모바일 사용자의 경로와 파일 인기도를 사전에 알고 있으며 SBS 의 범위가 겹치지 않는다고 가정한 환경에서 캐싱하는 첫번째 단계와, SBS 의 범위가 겹치는 환경에서 사용자가 더 많이 지나가는 SBS 에 대해 인기있는 파일을 재배치하는 두 번째 단계로 구성했다. 실험 결과 제안 기법이 SBS 캐싱 용량에 따라 MBS 로부터 다운로드 측면에서 DGCA 알고리즘보다 평균 40% 좋아졌고, 성공 비율 측면에서 최대 1.75 배 향상된 결과를 보였다.

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1F1A1047113).

참고문헌

[1] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022," Cisco, February, 2019.

[2] W. Liu, H. Zhang, H. Ding, D. Li and D. Yuan, “Mobility-Aware Coded Edge Caching in Vehicular Network with Dynamic Content Popularity,” IEEE Wireless Communications and Networking Conference (WCNC), Nanjing, China, 2021

[3] A. Mahmood, C. E. Casetti, C. Fabiana Chiasserini, P. Giaccone and J. Härri, “The RICH Prefetching in Edge Caches for In-Order Delivery to Connected Cars,” IEEE Transactions on Vehicular Technology, vol. 68, no.1, pp. 4-18, 2019

[4] E. Ozfatura and D. Gündüz “Mobility and Popularity-Aware Coded Small-Cell Caching,” IEEE Communications Letters, vol. 22, no. 2, pp. 288-291, 2018.

[5] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen and W. Zhu, “Understanding Performance of Edge Content Caching for Mobile Video Streaming,” IEEE Journal on Selected Areas in Communications, vol. 35, no. 5, pp. 1076 – 1089, 2017