A Study on Edge Proactive Caching Strategy with Graph-based Method

Ida Nurcahyani, Jeong Woo Lee*

Chung-Ang University

idanurcahyani@cau.ac.kr, *jwlee2@cau.ac.kr

Abstract

Edge caching technology in telecommunication networks plays a crucial role in optimizing content delivery services by storing content closer to end-users, thereby reducing delivery delays. Determining which contents should be kept in the edge server is an important issue that needs analyzing users' access patterns and predicting content popularity. Proactive caching strategy utilizes predictive analytics and machine learning that pre-caches contents based on forecasted demands. In this study, we implement a graph-based method in the caching strategy that captures the relationship between users and items based on users' past interactions. Simulation result shows that graph-based method performs better than the popularity method.

Ⅰ. Introduction

The edge caching in telecommunication networks is a crucial technology to optimize content delivery services. It allows content providers storing their services closer to end-users at the edge of the network. This is done by placing servers at the networks' edge node, thus improving the efficiency by reducing service delivery latencies. Furthermore, edge caching technology also alleviates the burden of the core network by reducing congestion at the backbone links.

The strategies to determine which services or contents that should be kept in the edge caching is an important issue that has been attracting interest from industries and academia. The algorithm and methods to select contents that align with users' interest can enhance the Quality of Service (QoS) of the network and improve customer's experience and satisfaction. This method usually involves analyzing users' access pattern, predicting contents' popularity, and determining caching policy. This data-driven approach enhances edge caching benefits by proactively storing content before it is requested by users.

Proactive caching utilizes predictive analytic techniques that often involve machine learning algorithms to forecast future content popularity and demand. Based on the prediction result, edge pre-caches content before actual requests. Pre-fetching contents can be done periodically to reduce computational and communication costs or in real-time to enhance services based on caching policies.

The main objective of edge caching is to minimize content delivery delays to end-users. Service delay is a crucial matrix to determine a network's performance. It also directly impacts users' experience that leads to customers' satisfactions and engagements. Designing caching mechanisms is an important task in delivering high performance networks that meet the needs and expectations of end-users. In this study, we want to investigate the impact of implementing proactive caching strategy in edge networks for the service delivery.

Ⅱ. Method

We consider a cellular network with one macro-cell Base Station (BS) that is connected to a Cloud Server and several small-cell BSs (s-BS) by backbone networks. Assume that s-BS uses orthogonal frequency division multiplexing (OFDM) to communicate with its end-users, so no interference is considered in this model.

For each request sent to edge server m, it first examines its local storage. If the item requested is available, then s-BS m sends it directly to the user. Otherwise, s-BS m fetches the item from BS, copies it to its local storage and sends it to the requester.

We denote s_i as item i file size and R_u^t as user u data rate at time t. Then the fetching delay experienced by user u in serving edge m as:

$$d_{ui}^t = \left({s_i}/{R_{u,m}^t}\right) + (1-\eta_i)\left({s_i}/{R_{m,BS}^t}\right) \quad (1)$$

Where $\eta_i \in \{0,1\}$ is the indicator whether of item i is found, $(\eta_i = 1)$, in the edge m storage or not, $(\eta_i = 0)$. $R_{u,m}^{dl}$ and $R_{m,BS}^{dl}$ are the achievable data rate for the downlink communication between client m and user u, and between s-BS m and BS respectively. To minimize fetching delays, edge m tries to predict the most demanded items in its by pre-fetching items before actual requests come.

The proactive caching mechanism considers users' preferences when selecting which items should be kept in edge local storages. When a user is connected to the edge client, it reports its past interaction data. Edge clients gather all its active users' historical data and leverage it to predict contents that are likely to align to

its users' interest by analyzing patterns in users' behaviors. Commonly, users with similar behavior would indicate similar preference on contents.

We can reconstruct historical interaction between users and contents by parameterizing them and extract information from their interaction to predict users' preferences based on parameterized relationship between them [1]. This process can be done by learning latent features that represent users and items past interactions. We can express users and items to vectorized representations, or embeddings, and then model their interaction by reconstructing their historical relations based on its embeddings.

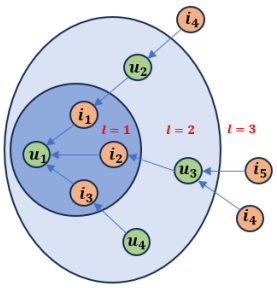

Fig. 1 Interaction Graph

In order to discover the users-item high order connectivity, we can implement a user-item bipartite interaction graph, shown in Fig.1. Where users and items are represented as nodes and their interaction as edges [2]. The presence or absence of an edge represents how a user interacts with an item. This graph structure is also where the collaborative signals between users-item are embedded.

In the learning process, the collaborative signals are utilized into user-item embeddings by propagating information through these graphs. Using this interaction graph, we allow information to pass through between connected users and items. Thus, enabling the model to learn the representation of users-items collaborative signals. The learned model shows information about users' preferences and items' characteristics.

We construct a connected user-item message pair (u, i) as:

$$\boldsymbol{m}_{u \leftarrow i} = f(\boldsymbol{x}_u, \boldsymbol{x}_i, \beta_{ui}) \tag{2}$$

where $\boldsymbol{x}_u$ and $\boldsymbol{x}_i$ are embedding vector for user u and item i respectively, and β_{ui} is a coefficient to control the decay factor on each propagation edge (u, i). The value of $f(\cdot)$ is written as:

$$f(\cdot) = \frac{1}{\sqrt{|H_u||H_i|}}\left(\boldsymbol{w}_1 \boldsymbol{x}_i + \boldsymbol{w}_2(\boldsymbol{x}_i \odot \boldsymbol{x}_u)\right) \tag{3}$$

where $1/\sqrt{|H_u||H_i|}$ as the graph Laplacian norm with H_u and H_1 are the first-hop neighbor of user u and item i respectively.

To investigate our system performance, we utilize MovieLens 100k dataset in our simulation. In order to minimize the system's complexity, we assume that each item has the same size. We use three edge clients with a maximum capacity of 1000 items. Each client performs its learning with its local dataset and generates local predictions. We compare the graph-based result with most popular method where items with most access gain most popularity in Fig. 2.

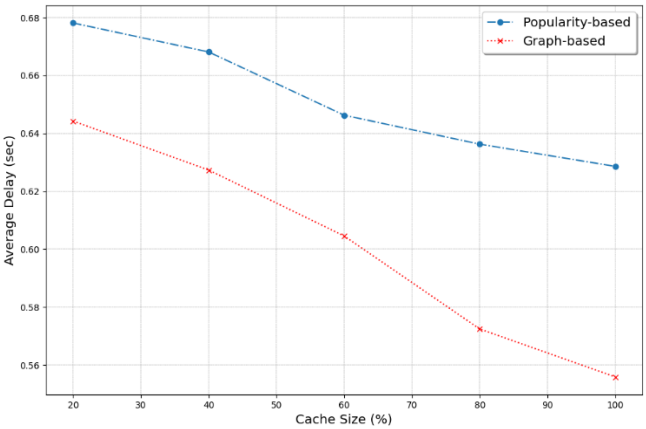

Fig. 2 Delay simulation result

Ⅲ. Conclusion

This study examines the impact of employing a caching strategy using graph-based approach to select caching items in edge networks. The result shows that the graph-based method performs better than the most popular method. This is due to the ability of graph-based methods that utilize the collaborative signals between users and items when learning users-item vector representations. Although the result is better than the popular-based method, further investigation is required to understand the relationship between end-delay and caching accuracy.

ACKNOWLEDGMENT

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-RS-2022-00156353) supervised by the IITP (Institute for Information & Communications Technology Planning &Evaluation).

REFERENCES

[1] Y. Cao, X. Wang, X. He, Z. Hu, and T. S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, pp. 151–161, 2019, doi: 10.1145/3308558.3313705.

[2] X. Wang, X. He, M. Wang, F. Feng, and T. S. Chua, "Neural graph collaborative filtering," *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 165–174, 2019, doi: 10.1145/3331184.3331267.