The Fractional Cooperative Caching in Content-Oriented Networks

Phuong Luu Vo and Choong Seon Hong
Department of Computer Engineering, Kyung Hee University
Email: {phuongvo, cshong}@khu.ac.kr

Abstract

Solving the large-scale linear programming is a challenge in the cooperative caching problem due to the large amount of contents in the network. We propose a decomposition approach that decomposes the large-scale problem into many subproblems which can be solved parallelly. Each of them is associated with one content.

1. Introduction

In the content-oriented networks, the contents are cached in the storage nodes placed in the service provider's network. The users can get the contents at the intermediate nodes instead of downloading from their original sources. Thus, the performance of content delivery to the end users is improved. Idealistically, all the contents should be cached in each storage node. However, there are a large amount of contents in the network whereas the storage capacity is limited. Hence, the nodes usually cache the most popular contents. When a node receives a request for a particular content, if it already has the content in its storage, the content is sent back to the end user directly. Otherwise, the content is fetched from other nodes in the network or from its repository.

It has been known that the nodes in the network should cooperate in caching in order to reduce the bandwidth cost of the traffic transferred within the network. Given the demands of the contents, the optimal content placement and fetching strategy are proposed by solving a integer linear program that minimizes the total bandwidth cost of the network [1,2,3]. In case the integral constraints are relaxed, the problem models the fractional cooperative caching which is considered in our paper.

However, the cooperative caching problem is a large-scale linear program because of a large amount of contents in the network. Utilizing the block-angular structure of the problem, we decompose the problem into many subproblems which can be solved parallelly. Each subproblem is associated with one content.

2. The bandwidth-expense minimization problem

Consider a network with M contents and S storage nodes providing the caching service to the customers. The nodes collaborate in caching in order to reduce the total bandwidth cost. When a storage node i receives a request for content m. If it already has x_i^m fraction of content m in its storage, node i continues to forward the content m's request to all the other nodes. In return, it receives y_{ij}^m fraction of content m from each node j and z_i^m fraction from content m's repository.

Usually, the content would rather be downloaded from the storage nodes than from the repository. The cost of transferring one unit of content m from its repository to serve the request at node i, c_i^m, is higher than that from the other node j, c_{ij}, i.e., $c_i^m > c_{ij}$.

In this paper, we assume that the demands for the contents on the nodes are given and constants during the time period of interest.

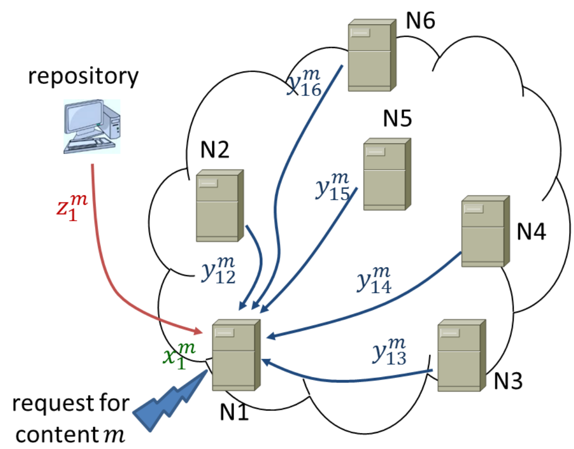

Fig. 1. An example of cooperative caching in a content-oriented network with six nodes. To serve the request for content m at node 1 which already has x_1^m fraction of the content, node 1 receives y_{1i}^m fraction of content from node i, $i \in S$, and z_1^m fraction of content from m's repository.

The amount of bandwidth cost in the network for serving the demand of content m at node i is the total of amount bandwidth cost of content m from the other storage nodes and the repository, i.e., $s^m d_i^m(\sum_{j\in S,j\neq i} c_{ij} y_{ij}^m + c_i^m z_i^m)$ where s^m is the size of content m and d_i^m is the demand of content m at node i. Hence, the total amount of bandwidth cost of the network is $\sum_{m\in M}\sum_{i\in S} s^m d_i^m(\sum_{j\in S,j\neq i} c_{ij} y_{ij}^m + c_i^m z_i^m)$. We need to choose a content placement and fetching strategy in order to minimize the bandwidth expense of the network.

LP:

$$\text{Min.}\sum_{m\in M}\sum_{i\in S} s^m d_i^m\left(\sum_{j\in S,j\neq i} c_{ij} y_{ij}^m + c_i^m z_i^m\right)$$

$$\begin{aligned}
\text{st. } & \textstyle\sum_{m\in M} s^m x_i^m \le B_i,\ \forall i \in S, && (1)\\
& y_{ij}^m \le x_j^m,\ \forall i,j \in S, j \neq i, m \in M && (2)\\
& x_i^m + \textstyle\sum_{j\in S,j\neq i} y_{ij}^m + z_i^m \ge 1,\ \ \forall i \in S, m \in M && (3)\\
& x_i^m \ge 0, y_{ij}^m \ge 0, z_i^m \ge 0,\ \forall i,j \in S, j \neq i, m \in M && (4)
\end{aligned}$$

The constraint (1) is the storage capacity constraint of each node, i.e., the total storage of all contents/fraction of contents on node i must be less than its storage capacity B_i. The constraint (2) implies that the fraction of content m from node j serving the request at node i must be less than the amount of content m stored at node j. Finally, the constraint (3) guarantees the amount of content m stored at node i and the total amount of m received together must be larger than the full content in order to serve the request. At the optimal point, the equality actually holds in constraint (3). Hence, $x_i^m \le 1$, $y_{ij}^m \le 1$, and $z_i^m \le 1$ are implied implicitly.

3. Solving the large-scale linear program

Although a linear program can be solved in a polynomial time, LP is actually a large-scale problem since there are a large number of contents in the network. However, LP has a special structure, the block-angular form (see Fig.2) which can be decomposed into many subproblems. To solve the problem, we first relax the non-separable constraint (1). Then, we decompose the relaxed problems into many subproblems according to the contents. Each subproblem is a linear program associated with each content.

Particularly, the relaxed problem of LP has the following form

$$\text{Min. } \sum_{m\in M}\sum_{i\in S} s^m d_i^m\left(\sum_{j\in S,j\neq i} c_{ij} y_{ij}^m + c_i^m z_i^m + \alpha_i \frac{x_i^m}{d_i^m}\right) - \sum_{i\in S} B_i \alpha_i$$

$$\begin{aligned}
\text{st. } & y_{ij}^m \le x_j^m,\ \forall i,j \in S, j \neq i, m \in M\\
& x_i^m + \textstyle\sum_{j\in S,j\neq i} y_{ij}^m + z_i^m \ge 1,\ \ \forall i \in S, m \in M\\
& x_i^m \ge 0, y_{ij}^m \ge 0, z_i^m \ge 0,\ \forall i,j \in S, j \neq i, m \in M
\end{aligned}$$

where $\boldsymbol{\alpha}$ is the Lagrange multiplier associated with constraint (1). Therefore, the subproblem associated with content m has the following form

LP(m):

$$\text{Min.}\sum_{i\in S} s^m d_i^m\left(\sum_{j\in S,j\neq i} c_{ij} y_{ij}^m + c_i^m z_i^m + \alpha_i \frac{x_i^m}{d_i^m}\right)$$

$$\begin{aligned}
\text{st. } & y_{ij}^m \le x_j^m,\ \forall i,j \in S, j \neq i,\\
& x_i^m + \textstyle\sum_{j\in S,j\neq i} y_{ij}^m + z_i^m \ge 1,\ \ \forall i \in S,\\
& x_i^m \ge 0, y_{ij}^m \ge 0, z_i^m \ge 0,\ \forall i,j \in S, j \neq i,
\end{aligned}$$

Giving $\boldsymbol{\alpha}$ in each iteration, M subproblems are solved parallelly. We apply Dantzig-Wolfe decomposition approach to find the optimal multipliers in each iteration by solving the weighting problem as described in [4, chap.12]. The objective of the weighting problem is indeed a potential function that monotonically decreases in each iteration.

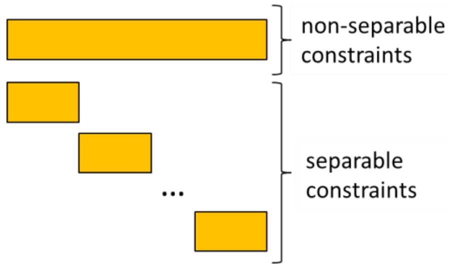

Fig. 2. The block-angular structure of LP.

For instance, with 20 storage nodes and 10,000 contents in the network, by using a normal PC (processor AMD Athlon IIx2 and 2GB memory), we cannot solve LP because of out of memory. However, by decomposition, the problem is solved after 1 hour 45 minutes on the same PC.

4. Numerical results

In the experiment, the demands of the contents follow Zipf-Mandelbrot distribution

$$d_i^m = r\frac{(q+m)^{-a}}{\sum_{m=1}^{M}(q+m)^{-a}}$$

with shift parameter $q = 10$ and shape parameter $a = 1$ [5]. The request rate to each node is 1 request per minute.

The costs of the path between two nodes $c_{ij} = 2$ GB per minute for all i,j. Suppose that the cost to transfer one unit of content m from the original source to the node i, c_i^m, is 5 GB per minute for all $i \in S$ and m$\in M$. Assume that the network has 10,000 contents and 50 storage nodes. All the contents have the equal sizes, i.e., $s = 2$ GB. The storage capacity of each node is 1000 GB.

Without cooperation, the contents are fetched from the repositories in case the contents are not stored on the nodes. The cost of the network is 216.92 GB per minute for noncooperation. By collaborating 05 nodes (in 50 nodes of the network), the cost drops down to

212.99 GB per minute. By collaborating all 50 nodes, the cost of the network is only 98.68 GB per minute, decreases by 54.51% the cost of non-cooperation (see Fig.3).

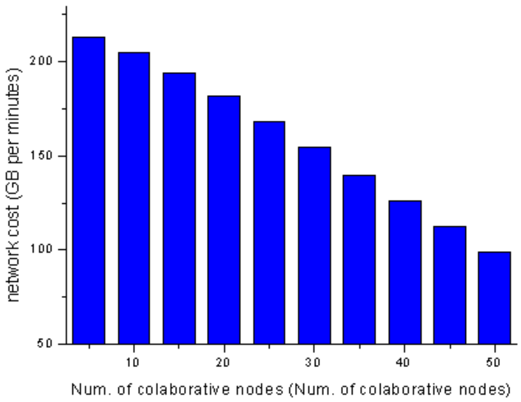

Fig. 3. The cost of the network as increasing the number of nodes collaborating.

5. Conclusions

Utilizing the block-angular structure of the primal problem, we apply Dantzig-Wolfe decomposition approach to solve the large-scale linear program. The main problem is decomposed into many subproblems which can be solved parallelly.

Acknowledgment

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2010-0020728). Dr. CS Hong is corresponding author.

Reference

[1] S. Borst, V. Gupta, and A. Walid, “Distributed caching algorithms for content distribution networks,” in Proceedings IEEE INFOCOM 2010, march 2010, pp. 1–9.

[2] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, “Optimal content placement for a large-scale vod system,”
in Proceedings of the 6th International COnference, ser. Co-NEXT ’10. New York, NY, USA: ACM, 2010, pp. 4:1–4:12.

[3] H. Xie, G. Shi, and P. Wang, “Tecc: Towards collaborative in-network caching guided by traffic engineering,” in Proceedings IEEE INFOCOM 2012, march 2012, pp. 2546–2550.

[4] S. P. Bradley, A. C. Hax, T. L. Magnanti, Applied mathematical programming. Addison-Wesley, 1977.

[5] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and zipf-like distributions: evidence and implications,” in Proceedings IEEE INFOCOM ’99, vol. 1, mar 1999, pp. 126–134 vol.1.