Research on Cache Placement in ICN

Yu Zhang[1,2(✉)], Yangyang Li[1], Ruide Li[1], and Wenjing Sun[1]

[1] Beijing Institute of Technology, Beijing, China
{yuzhang, lyy_bl}@bit.edu.cn
[2] The Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory, The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang, China

Abstract. Ubiquitous in-network caching is one of key features of Information Centric Network, together with receiver-drive content retrieval paradigm, Information Centric Network is better support for content distribution, multicast, mobility, etc. Cache placement strategy is crucial to improving utilization of cache space and reducing the occupation of link bandwidth. Most of the literature about caching policies considers the overall cost and bandwidth, but ignores the limits of node cache capacity. This paper proposes a G-FMPH algorithm which takes into account both constrains on the link bandwidth and the cache capacity of nodes. Our algorithm aims at minimizing the overall cost of contents caching afterwards. The simulation results have proved that our proposed algorithm has a better performance.

Keywords: ICN · Steiner tree · Link cost · Cache placement · Group multicast

1 Introduction

Recently, with the rapid development of internet, network architecture based on contents gets the favor of researchers. The Palo Alto research center has put forward a landmark ICN network architecture CCN in 2007, which aims to provide an efficient and extensible content access application pattern for solving the insuperable internet traffic explosion problem [1]. ICN directly names the contents and doesn't focus on where the contents are causing the extensive concern of academic community [2]. Many research institutes carry out the research work of Information Centric Network, but the performance of many key technologies needs to be improved such as: caching strategy, routing mechanism, mobility and so on [3, 4].

A major feature of designing the ICN network architecture is in-network information caching, which has the advantages of improving the efficiency of contents distribution, distributing contents to the network edge, balancing the network bandwidth and load, etc. [5]. In recent years, researches on ICN network architecture have achieved substantial progress in the optimization methods, theoretical models and

Foundation Item: Science and Technology on Communication Networks Laboratory Foundation Project; Aerospace Field Pre-research Foundation Project (060501).

X. Gu et al. (Eds.): MLICOM 2017, Part II, LNICST 227, pp. 603–611, 2018.
https://doi.org/10.1007/978-3-319-73447-7_65

many other fields, but there are still a lot of problems to be solved. In ICN network, the redundant contents and the bandwidth consumption could be reduced through the content caching of nodes, this paper focus on how to select the cache nodes optimally, when requests returning to consumers. In the process of contents request returning to consumers, contents are stored on path nodes according to the cache management strategy. And the cache management strategy can be divided into two parts, i.e. cache placement policy and cache replacement policy. Appropriate cache placement is better support for content distribution, multicast, mobility, etc., therefore, the design of cache placement strategy is the key technic to performance of ICN. This paper mainly focuses on cache placement strategy.

Traditional algorithms generate a set of trees one by one, ignoring constrain of cache capacity which could lead to deterioration in performance, such as contents missing, larger delay, and network overload. In order to place contents effectively in the intermediate path nodes, we formulate cache placement as an extension of Group Steiner tree problem [6]. In our formulation, both the bandwidth and node's cache capacity are constrained, then meeting the needs of many-to-many data transmission and reception by establishing multiple Steiner tree.

2 Related Work

Recently, the way of communication in the network has been developed from one-to-one to one-to-many or many-to-many mode. So the research of multipoint communication has become an important topic in the field of network communication. Multipoint communication could be divided into two parts, i.e. "One-to-Many" and "Many-to-Many".

One-to-Many content distribution can be formulated as the minimum cost multicast tree problem, which is a typical NP-complete problem. There are some well-known algorithms such as MPH [7], Kou [8], Takahashi [9], Maxem-chuk [10] and Jingtao [11] algorithm, and their time complexity and the overall cost are much the same. Literature [12] proposes an improved MPH algorithm based on local search, which is called LSMPH (locally search minimum path cost heuristic), and its time complexity is low, but the total cost is generally greater than MPH algorithm. Literature [13] proposed a FMPH (Fast Minimum Path Cost Heuristic) algorithm aiming at solving the existing problems of MPH algorithm. The multicast tree established by FMPH algorithm is exactly the same as the tree by MPH algorithm, but FMPH algorithm improves the searching process of the shortest path node, so time complexity and the storage space will be reduced. Therefore, the Fast Minimum Path Cost Heuristic (FMPH) method can meet our need very well.

Many-to-Many content distribution problem is a typical group multicast routing problem (GMRP). At present, the researches on GMRP are still rare. Two methods can be generalized for solving the group multicast problem. The first method is to establish a tree for each set of multicast memberships, doing some coordination while building multiple trees so that the performance is optimal. Fei calls this method "Per-source-tree" [14]. Jia and Wang give a group multicast algorithm based on KMB algorithm called Jia and Wang's algorithm [6]. And then, Low and Wang give another

algorithm based on TM [15] called GTM, whose performance is better compared with Jia and Wang's algorithm, but its traffic distribution is not fair enough. The same author gives a FTM based on the TM [16], which can fairly distribute the traffic, but the cost is higher. The other method is CBT (Core Based Tree), which only constructs one tree, then the root of the tree will be the center for multicasting to all member nodes, the minimum cost is extended to the group, Fei calls this method STGM [14]. However, when using CBT, the source has to pass through some of the edges connected with the kernel, which can cause congestion at these edges. At the same time, the selection of multicast kernel is crucial to the performance of the established group multicast trees.

On the actual network environment, the data packet may be sent to multiple destination nodes when it returns, meantime, many consumers may request different contents. In other words, the source node may also be the destination node, therefore, we need to further study to solve our problem based on the group multicast routing. In group multicast routing, each established tree must contain the given nodes set, but our model is to assign some given nodes sets and each established tree must contain the corresponding nodes set. So the first step is to establish the source nodes set, and then each source node will be in the given collection. Finally, in the optimal tree sets we build, each tree needs to contain those nodes which are in the collection, and the extra nodes contained in those tree are intermediate nodes which is used to cache contents.

3 Problem Formulation

The network model is a graph $G = (V, E)$, and the bandwidth $b(i,j) \geq 0$ is asymmetric, i.e. $b(i,j) \neq b(j,i)$, and then the edge from node i to node j is e_{ij}, so if $\forall e_{ij} \in E$, then $\forall e_{ji} \in E$, each edge in G has a link cost $c_{ij} > 0$. We define Bf_i is the cache threshold of each node.

Let $D(D \subseteq V, |D| = m)$, $D = \{d_1, d_2, \cdots, d_m\}$ is a group of source nodes in G, and then define $D' = \{D'_1, D'_2, \cdots D'_m\} \in G$ (d_i is the root of D'_i) is the group multicast sets. The bandwidth requirement for the nodes in D'_i is defined as $BW = \{bw_1, bw_2, \cdots bw_m\}$, and then we need to find a set of directed routing tree $\{T_1, T_2, \cdots T_m\}$, $T_i = (V_i, E_i)$ Assuming that all nodes in the optimal sets are collected to P, $P = \{p_1, p_2, \cdots p_q\} p_q \subseteq V_i$,so the following requirements are the constraints:

$$\min \sum_{k=1}^{m} \sum_{(i,j) \in T_k} c_{ij} X_{ij}^k, \qquad i \in V, j \in V \tag{1}$$

$$\sum_{k=1}^{m} bw_k X_{ij}^k \leq b(i,j), \forall e_{ij} \in E, X_{ij}^k = \begin{cases} 1, & e_{ij} \in E_k \\ 0, & else \end{cases} \tag{2}$$

$$\sum_{k=1}^{m} Y_r^{T_k} bw_k \leq Bf_r, \forall r \in P \tag{3}$$

In formula (1), it is used to ensure that the overall cost of the group multicast tree sets is optimal. In formula (2), it is used to constrain the total bandwidth of each edge. Formula (3) is the paper's key, it is used to constrain the node's cache threshold. A set of trees $\{T_1, T_2, \cdots T_m\}$ $(T_i(1 \leq i \leq m))$ which satisfies those constrains is our feasible solution called G-FMRP.

For the sake of contract, we should deal with the overall link cost of the algorithm ignoring the cache overflow of nodes, the following are constrains:

$$\Delta x = (\sum_{k=1}^{m} Y_r^{T_k} bw_k - Bf_r)/Bf_r \tag{4}$$

The ratio of cache overflow is defined as Δx, if node r is belonging to the tree T_k, then $Y_r^{T_k} = 1$ else $Y_r^{T_k} = 0$, so the final overall cost for all trees is:

$$\cos t_{all} = \sum_{k=1}^{m} \sum_{(i,j) \in T_k} c_{ij} X_{ij}^k + \sum_{k=1}^{m} \sum_{r=1}^{q} \Delta x \cdot Y_r^{T_k} c_{T_k} \tag{5}$$

4 The Proposed Algorithm

According to the above analyzing process, the first step of our proposed algorithm is to establish a multicast tree using FMPH algorithm, then generating a set of trees cooperatively based on the FMPH algorithm. We need to build group multicast trees, so we call it G-FMPH algorithm. The procedure stops if some saturated edges occurs when we build tree T_i, and then the saturated edges make up a set defined as E'. All trees (except T_i) have the saturated edges make up a set defined as M. Finally we will compare the alternative link cost of tree T_i with the most recently built tree (or trees), the smaller one will be changed to the alternative tree. Simultaneously, each multicast tree needs to determine whether the node cache constraints are satisfied or not, in order to ensure that the cache of each node isn't overflowed. If the tree does not satisfy the cache constraint, we will delete the overflowed node. Note: no saturated edge is used during adjustment. The details of G-FMPH are given in Fig. 1.

We take the topology in Fig. 2 to establish the group multicast trees. The number at both ends of each arrow indicates the available bandwidth in the corresponding direction, and then the number in the middle indicates the cost. Figures 3 and 4 illustrate the procedure of creating group multicast trees by G-FMPH. To better describe the treatment processing for cache overflow, we assume that the available bandwidth of nodes in the Fig. 2 is abundant. We suppose a situation where three different requests appear, and then when data packets return to consumers, we need to choose appropriate caching nodes. The source nodes set is denoted as $D = \{A, B, C\}$, and the destination nodes sets are denoted as $d_1 = \{B, C\}$, $d_2 = \{A, C\}$ and $d_3 = \{A, B\}$. The bandwidth requirement for each tree is $BW = \{2, 2, 3\}$ and the cache threshold of each node is 5 units.

Input: graph $G=(V,E)$ **, a set of source nodes** D_k **,** bandwidth request $BW=\{bw_1,bw_2,\cdots bw_m\}$

Output: a multicast tree $T_k(V_t,E_t)$ **,** $k\in D_k$

```
1  if G is not connected then stop
2  for each node k ∈ D_k
3      if (min {bw_1,bw_2,⋯bw_m} >the available bandwidth of the edge)
4          delete the edges and update G
5  if G is not connected then stop
6  compute shortest paths V → D_i
7  for i = 1 to k
8      build multicast tree T_i using FMPH algorithm;
9      if there are overflowed nodes in T_i , delete the nodes and edges connecting with the nodes
10     if there are saturated edges in T_i
11         a set of saturated edges in T_i defined as E' , and then delete E' from G to get graph G'
12         compute shortest paths V' → D_i
13         build alternative multicast tree T_i' using FMPH
14         compute the overhead O_i = c(T_i') − c(T_i)
15         multicast trees contain edges in E' make up a set M
16         for each tree T_j ∈ M
17             compute shortest paths V' → D_i
18             build alternative multicast tree T_j' using FMPH
19             compute the overhead O_j = c(T_j') − c(T_j)
20             if O_i > Σ_{T_j∈M} O_j
21                 T_i use the saturated edges; all the other trees T_j ∈ M → T_j'
22             replace T_i by T_i' ; end if;
23         update the bandwidth status of all the edges and the cache of nodes;
24 end for;
25 end;  (Procedure G-FMPH)
```

Fig. 1. G-FMPH algorithm

Here we only consider cache overflows of nodes to simplify the process.

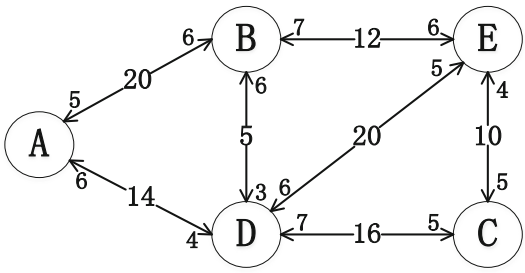

Fig. 2. A simple network topology

Starting from node A, the shortest path from A- > B is 19, which is smaller than A- > C (shortest path 30). The path is A- > D- > B and the path node is D, and then we need to update the available bandwidth of the used edges in the direction. The shortest path from D- > C is 16, which is smaller than the shortest path from A- > C (30). Therefore, the path D- > C is added to multicast tree. Finally, tree A is built shown in Fig. 3(a). The other two trees are built using the same method shown in Fig. 3(b) and 3 (c) respectively, ignoring the effect of cache overflow of node D.

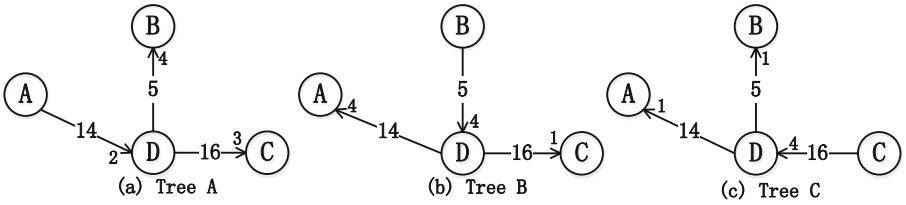

Fig. 3. An illustration of the traditional algorithm

The bandwidth requirement bw_1 and bw_2 is 4 units, and then bw_3 is 3 units, therefore, the cache of node D is overflowed when we build tree C. An alternative tree C' will be built using our proposed algorithm above. We will delete the node D and the edges connecting with node D when we build tree C', and then we build tree C' in graph G'. Finally, the group multicast trees are built in Fig. 4.

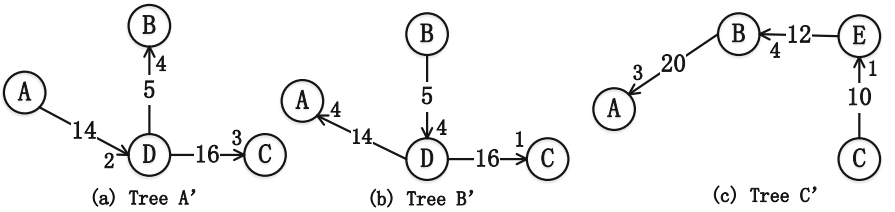

Fig. 4. An illustration of our proposed algorithm

Finally, compute the cost of trees established by using above two algorithms. The overall cost of traditional algorithm $\cos t_{tr}$ and our proposed algorithm $\cos t_{pr}$ are shown below calculated by formula (5) and formula (1).

$$\cos t_{tr} = \cos t_A + \cos t_B + \cos t_C = 119 \tag{6}$$

$$\cos t_{pr} = \cos t_A + \cos t_B + \cos t_{C'} = 112 \tag{7}$$

We can conclude that our proposed algorithm has smaller cost than the traditional one because of $\cos t_{pr} < \cos t_{tr}$, so our proposed algorithm has better performance. The difference of the overall cost between the two algorithms will increase as the number of group multicast trees grows.

5 Simulation

In order to assess our proposed algorithm's performance, the simulations are conducted. The topology is randomly generated, and the link between two nodes i and j is added by probability function:

$$P(i,j) = \lambda \exp(-d(i,j)/\rho L) \tag{8}$$

In formula (8), $d(i,j)$ is the distance between i and j, and then the maximum distance between any two nodes is defined as L. The range of the parameters λ and ρ is $0 < \lambda \leq 1$ and $0 < \rho \leq 1$. The average degree of nodes will be higher if we improve the value of λ, and then the density of shorter links compared with longer ones will be higher by decreasing the value of ρ, therefore, we can construct the network topology by modifying λ and ρ [17]. In our simulation, $\lambda = 0.3$ and $\rho = 0.15$. The cost from i to j is calculated by random integers (20, 50). The bandwidth is calculated by the formula (9):

$$b(i,j) = b_{\min} + r \bmod (b_{\max} - b_{\min}) \tag{9}$$

In formula (9), $b_{\max}$ is defined as the maximum bandwidth, $b_{\min}$ is defined as the minimum bandwidth. The bandwidth requirement of each tree is calculated by random integers (3, 5), and the cache threshold of each node is calculated by random integers (15, 20).

In the simulation, the overall cost is calculated by multicast trees. To insure the accuracy of the result, we simulate 10 times to get the average result.

Figure 5 shows the result of our proposed algorithm and traditional algorithm assigning $b_{\max}$ = 15, $b_{\min}$ = 5, $L = 200$ and the network size is 150. The abscissa and ordinate represent group size and network cost respectively. As the growth of the group multicast size, the gap becomes larger and larger. In the beginning, the group multicast size is small, therefore node cache may not be overflow, and then the overall cost of our proposed algorithm is the same as the traditional algorithm. But, many multicast trees are established as the growth of the group multicast size, this phenomenon may lead to cache overflow of partial nodes. Our proposed algorithm considers the cache overflow of nodes, but traditional algorithm ignores the cache overflow of nodes which will cause large additional overhead. Therefore, the overall cost of tradition algorithm is larger and the gap becomes larger as the growth of the group multicast size.

Figure 6 shows the result of our proposed algorithm and traditional algorithm assuming group size = 16. The abscissa and ordinate represent network size and network cost respectively. We can intuitively observe that the curve of traditional algorithm is higher than our proposed method. As the growth of network size, the gap becomes smaller. In the beginning, the network size is small, the group size that we assign is 16 which means we need to establish 16 trees, so we need to use many suboptimal paths and alternative trees because of constrains of bandwidth and cache, and then the overall cost is large. As the growth of the network size, node cache may not be overflowed, so the gap becomes smaller, finally two algorithms become nearly the same cost, which means there is no cache of nodes overflow.

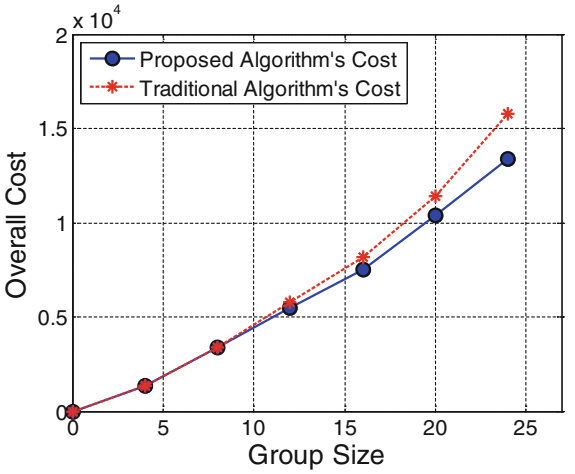

Fig. 5. Overall cost over group size

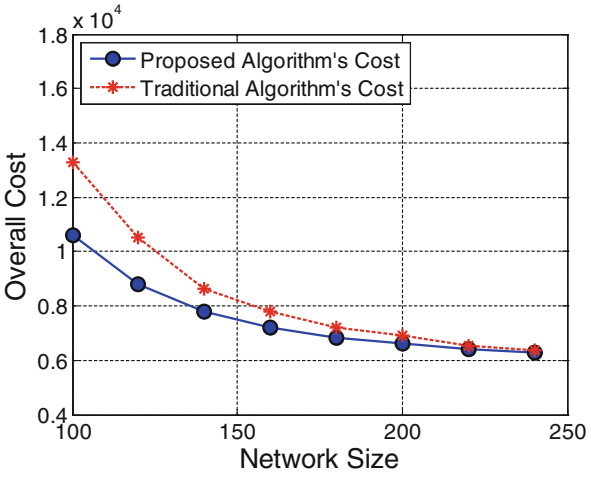

Fig. 6. Overall cost over network size

6 Conclusion

Traditional algorithms generate a set of trees one by one ignoring the limits of cache capacity which could lead to deterioration in performance, such as contents missing, larger delay, and network overload. Choosing appropriate caching placement policy is crucial to improving utilization of cache space and reducing the link cost when the data packet retrieval in ICN. In this paper, we use an extension of Steiner tree formulating the problem, and then we propose a G-FMPH algorithm which takes into account constrains of both available link bandwidth and the cache capacity limitation of nodes.

The simulation result shows that our algorithm has the superior performance over the traditional algorithm.

References

1. Jacobson, V., Smetters, D.K., Thornton, J.D., et al.: Networking named content. In: International Conference on Emerging Networking Experiments and Technologies, 55(1), pp. 1–12. ACM (2009)
2. Dannewitz, C., Golic, J., Ohlman, B., et al.: Secure naming for a network of information. In: INFOCOM IEEE Conference on Computer Communications Workshops, pp. 1–6. IEEE (2010)
3. Carofiglio, G., Gallo, M., Muscariello, L., et al.: Modeling data transfer in content-centric networking. In: Teletraffic Congress, pp. 111–118. IEEE (2011)
4. Muscariello, L., Carofiglio, G., Gallo, M.: Bandwidth and storage sharing performance in information centric networking. In: ACM SIGCOMM Workshop on Information Centric Networking, pp. 26–31. ACM (2011)
5. Shimizu, H., Asaeda, H., Jibiki, M., et al.: Content hunting for in-network cache: design and performance analysis. In: IEEE International Conference on Communications, pp. 3172–3177. IEEE (2014)
6. Jia, X., Wang, L.: A group multicast routing algorithm by using multiple minimum Steiner trees. Comput. Commun. **20**(9), 750–758 (1997)
7. Winter, P.: Steiner problem in networks: a survey. IEEE Netw. **17**, 129–167 (1987)
8. Kou, L., Markowsky, G., Berman, L.: A fast algorithm for Steiner trees. Acta Inf. **15**, 141–145 (1981)
9. Wang, B., Hou, J.C.: Multicast routing and its QoS extension: problems, algorithms, and protocols. IEEE Netw. **14**, 22–35 (2000)
10. Maxemchuk, N.F.: Video distribution on multicast networks. IEEE J. Sel. Areas Commun. **15**(3), 357–372 (1997)
11. Jingtao, S.U., Lin, F., Zhou, X., et al.: Steiner tree based optimal resource caching scheme in fog computing. China Commun. **12**(8), 161–168 (2015)
12. 李汉兵, 喻建平, 谢维信:局部搜索最小路径费用算法[J]. 电子学报, 28(5), pp.92--95 (2000)
13. Guang-Min, H.U., Le-Min, L.I., Hong-Yan, A.N.: A fast heuristic algorithm of minimum cost tree. Acta Electronica Sinica **30**(6), 880–882 (2002)
14. Fei, A., Duan, Z., Gerla, M.: Constructing shared-tree for group multicast with QoS constraints. In: Global Telecommunications Conference, pp. 2389–2394. IEEE (2001)
15. Low, C.P., Wang, N.: An efficient algorithm for group multicast routing with bandwidth reservation. Comput. Commun. **23**(18), 1740–1746 (2000)
16. Wang, N., Low, C.P.: On finding feasible solutions to the group multicast routing problem. In: Pujolle, G., Perros, H., Fdida, S., Körner, U., Stavrakakis, I. (eds.) Networking 2000. LNCS, vol. 1815, pp. 213–227. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45551-5_19
17. Waxman, B.M.: Routing of multipoint connections. IEEE J. Sel. Areas Commun. **6**(9), 1617–1622 (1988)