A hybrid ICN cache coordination scheme based on role division between cache nodes

Guoqiang Zhang*,†‡§∥, Xiaohui Wang†, Qian Gao†¶ and Zhen Liu‡¶
* State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
† School of Computer Science and Technology, Nanjing Normal University, Nanjing, China
‡ Beijing Jiaotong University, Beijing, China
§ Department of Computer Science and Engineering, University of California, Riverside, USA
¶ University of Massachusetts Boston, USA
∥ Corresponding author’s email address: guoqiang@ict.ac.cn

Abstract—Information-Centric Networking (ICN) is a paradigm shift from traditional TCP/IP’s host-centric approaches, which relies on self-certifiable content, built-in pervasive in-network caching and flexible request routing to provide native support for efficient and scalable content distribution from the architectural layer. Caching in ICN diverts from traditional Web Caching due to its arbitrary network topology and the requirement for line-rate operation, which makes cache coordination a crucial task. However, most of the cache coordination schemes proposed to date use uniform caching coordination policies within the same cache network, which is either hard to achieve good balance between user QoE and resource utilization or hard to scale. This paper proposes a hybrid cache coordination scheme that bases on role division between cache nodes

I. BACKGROUND

Recently, the usage pattern of the Internet has shifted from sender-driven end-to-end communication to receiver-driven many-to-one content retrieval. The mismatch between TCP/IP’s unicast communication primitive and the application’s usage model is one of the root causes for today’s traffic explosion. Information Centric Networking (ICN) is a paradigm shift from the host-centric approaches [1] [2] [3] [4]. ICN relies on the decoupling between content providers and consumers, self-certifiable content, pervasive in-network caching and flexible request routing to satisfy the massive, asynchronous and heavy-tailed user requests, which provides native support for efficient and scalable many-to-one content distribution from the architectural layer. In addition, it is also believed that ICN is a good candidate to reduce energy consumption in massive content dissemination [5] [6].

Pervasive in-network caching is regarded as a fundamental building block of ICN. The pervasiveness makes ICN’s caching mechanism different from traditional Web caching. Unlike Web caching, the cache network topology in ICN is no longer hierarchical, but arbitrary [7]. Meanwhile, as a basic service, ICN’s caching should support line-speed operation, which limits the cache capacity of each cache node [8] [19]. Hence, cache coordination between different nodes becomes very crucial to reduce cache redundancy and improve the utility of cache resources.

Given a cache network, the primary goals of caching are mainly two-folded: to improve the user’s quality of experience (QoE) from users’ perspective, and to improve network resource utilization and reduce cross-domain traffic from the network’s perspective [12] [10]. Unfortunately, these two objectives are sometimes contradictory. In order to improve the user’s QoE, one tendency is to place as many popular content as possible to those cache nodes near users. This tendency, just as the on-path caching policy LCE does [18], will make many duplicated content copies of popular content, and thus reducing the cache diversity within the network. On the other hand, to fulfill the latter goal, one tends to increase the cache diversity. One extreme of this approach is the HASH-based cache coordination [9] [11]. This off-path cache coordination policy leaves at most one copy of each content in the cache network regardless of the content popularity. This, however, may degrade the user’s experience. So, a good cache coordination scheme needs to balance between these two goals.

Besides the tradeoff between QoE and network resource utilization, there is a growing concern of how to make cache coordination between different networks [20] and how to make a cache coordination scheme scalable so that it can be used in large ISPs. This requires the ability for different cache networks or different network portions of the same large network to apply different cache coordination policies rather than to adopt a universal cache coordination policy. Consequently, a good cache coordination scheme needs to support the enforcement of flexible or even autonomous policies.

This paper proposes a hybrid cache coordination scheme based on role division between cache nodes. It divides a network into one core area and several edge areas. The scheme can make use of existing partition of large ISPs to establish the partition, such as OSPF areas. And, for networks without pre-existing partition, it applies the k-core decomposition to partition the cache nodes. Cache nodes in the core area applies
an explicit as well as off-path cache coordination policy, e.g., the HASH-based cache coordination policy, while the edge areas can apply any effective implicit and on-path cache coordination policies such as LCE and LCD [18]. By this separation, our hybrid cache coordination scheme honors both user’s QoE and network’s resource utilization, and can be easily extended to the environment of large networks.

The organizational structure of the remainder paper is as follows: section II describes related work; section III details our hybrid cache coordination scheme based on role division between cache nodes; section IV evaluates its performance by simulation and compares it with several other cache coordination schemes; finally, section V concludes this paper.

II. RELATED WORK

Research on cache network performance mainly focuses on optimizing network resource utilization and improving user’s QoE. The former is mainly measured by metrics such as cache hit ratio and number of hops for a request to be satisfied, while the latter is mainly measured by the user perceived latency. The behavior of a cache network can be summarized as a four tuple policies (DM, DS, FW, RP), where DM means static cache capacity dimension policy, DS means cache decision policy, FW means request forwarding policy, and RP means cache replacement policy [13].

Cache dimension policy is a static cache network planning policy that determines the cache capacity allocation under the restriction of total budget. Mostly, the decision depends on the network topology and network traffic pattern. Generally, it is believed that it is more beneficial to improve the capacity of edge nodes rather than core nodes [14] [19] [15]. Since this is not a dynamic policy we can adjust, we will simply adopt a uniform cache dimension policy in our study.

Cache decision policy is a hot topic in ICN research, which aims to make efficient use of the cache resources in the network through cache coordination between cache nodes. Typically, a solo cache coordination policy is applied to the whole cache network. There are various ways to classify different cache decision policies. One way to classify different policies is based on whether they use on-path or off-path caching [21] [32]. On-path cache coordination only caches data in some(or all) nodes along the path that the data follows to reach the requester, so that when the same data is requested by some requestor along the path later, it may be served on its way to the server [18] [27] [28] [25]. While in off-path caching, the data can be cached anywhere in the network. In this case, some mechanisms should be provided to direct the request to appropriate cache nodes so that the cached data can take effect [29]. Another way to classify cache decision policies is based on whether they use explicit or implicit cache coordination [7]. Explicit cache coordinations make their decisions on information such as network topology, request pattern, partial or global cache status [27] [29] [11] [9] [32], whereas implicit ones can independently make their own cache decisions [18] [28] [25]. Often, explicit cache coordinations can make better utilization of network resources, and armed with the knowledge of content popularity and global network topology such as the Adaptive Off-Path Caching [32], they can also reduce the hop delay to access content. But, explicit cache coordination either needs a centralized controller or incurs more computation and communication overhead. There is one exception within the explicit cache coordination family, i.e., the HASH-based cache coordination [9] [11], which requires little information and incurs little overhead. For this reason, we choose the HASH-based cache coordination in the core area of our hybrid scheme.

Request forwarding decides how to forward a request when a cache miss occurs, which can be either static or dynamic. Static policy makes its request forwarding decision solely based on network topology and the cache status information exchanged through the control plane [16] [26]. While dynamic policy can also rely on the response feedback from data plane, e.g., the successful rate and response quality of forwarding on a given interface, to dynamically adjust its request forwarding decision [17] [30]. Sometimes, the use of request forwarding policy is tightly coupled with the cache coordination policy being used. For example, when an off-path cache coordination such as HASH-based cache coordination is used, the same HASH function should also be used to route the request to the responsible cache node for the requested content.

Cache replacement policy decides which data is to be evicted from the cache when it is out of capacity. LRU is the most frequently used one in the literature, although it is observed that simple random cache replacement can also approximate the performance of LRU [22].

III. HYBRID CACHE COORDINATION BASED ON ROLE DIVISION BETWEEN CACHE NODES

A. Overview

In this section, we describe our hybrid cache coordination scheme in detail. The main objectives of our cache coordination scheme are: (1) to balance between the QoE and network resource utilization, and (2) to allow the enforcement of different cache coordination policies in different network parts. In general, our scheme is both a hybrid of on-path and off-path cache coordination, as well as a hybrid of explicit and implicit cache coordination.

In our scheme, the cache network is partitioned into a core area and several edge areas. Each of the edge areas can apply any of the implicit and on-path cache coordination policies. In this way, popular content can be pulled to the edge cache nodes and cached there, so future requests for these popular content can be quickly satisfied. In this paper, we evaluate two implicit cache coordination policies in the edge area, i.e., LCE and LCD. LCE is the default cache coordination in many ICN architectures, e.g., CCN [2], which leaves a copy everywhere along the data’s path to the requester regardless of its popularity. LCD is simple, but it leads to high cache redundancy and high cache replacement frequencies. LCD caches the data one hop further from the hit node, so only popular content will be pulled to the edge of the network quickly. LCD implicitly increases cache diversity between different cache nodes, so it is expected to provide better cache performance than LCE. In fact, more intelligent cache coordinations can be used in the edge area, e.g., MCD [18], Prob [18], WAVE [25], and better performance could be expected. We choose LCE and LCD simply because of their...
simplicity and it suffices to demonstrate the advantages of our hybrid approach.

The core area applies some kind of explicit and off-path cache coordination to maximize its cache diversity and reduce outbound/inbound traffic to/from other networks. Essentially, the core area is collectively regarded as a large logical cache node. In our scheme, we use the HASH-based cache coordination in the core area also because of its simplicity and its ability to maximize the cache diversity. The rational that only one copy is stored in the core area is that requests for popular content have high probability being absorbed in the edge areas, so that requests arriving at the core area are typically those requests for not-so-popular content. Since they are not so popular, the overhead incurred by off-path caching can be overlooked.

**B. Role Division between Cache Nodes**

For a large ISP network that has natural partitioning mechanism, e.g., with OSPF area support, we can establish a direct mapping between the OSPF areas and our cache areas. In this case, the backbone area serves the core area, and each non-backbone area is mapped to an edge area.

For those networks that do not have an existing partition, we adopt the $k$-core decomposition [23] [24] to perform the partition task. $K$-core decomposition is a common algorithm used to discover hierarchical structure in arbitrary network topologies. To extract the $k$-core in a network, the algorithm starts by removing all nodes in the network with less than $k$ neighbors. This process is repeated until no node can be removed. Fig. 1 illustrates the $k$-core decomposition process. In our scheme, we set $k = 2$ so the $k$-core decomposition process will give one 2-core which serves as our core area. We then group those nodes hanging from the 2-core based on their connectedness. That is, each maximally connected component is regarded as an edge area.

After the core nodes and edge nodes are identified, a level tag can be assigned to each node as follows: first, all the core nodes are assigned the same level tag, e.g., 1; then, for each edge node, the shortest path to the core area is identified and its upstream node to the core area is recorded. If the edge node is directly connected to a core node, then its level tag is set to be the level tag of the core node plus one; otherwise, if the edge node’s upstream node is also an edge node, its level tag is set to be the level tag of its upstream node plus one. After the level tag assignment, all nodes in the core area have the same level tag, while nodes in an edge area may have different level tags, which reflects their hierarchical positions in the topology. Fig. 2 illustrates the role division process. In this example, the partition process identifies a core area (nodes 11-17 with green color) and 3 edge areas.

**C. Request and Data Processing**

A content request sent by a user first reaches an edge area where the request will be served in a hierarchical hop-by-hop manner. Upon a cache miss, the request will be forwarded one hop further to an upstream node until it leaves the edge area. When the edge area can not satisfy the user request, the request will be forwarded to the core area, where it will be routed to a node in the core area that is in charge of the requested content according to the underlying hash function. For example, the core nodes can form a DHT and intermediate nodes forward the Interests based their requested content identifiers. When the core area can still not satisfy the request, finally the request will be forwarded to the content source according to the FIB.

If the request is satisfied in the edge area, then the data packet is returned and cached along its way if necessary according to the cache coordination policy in the specific edge area. If the request is satisfied in the core area, then the data packet is returned to the corresponding edge area, where it is returned back to the requester and cached along its way in the edge area if necessary. If the request is satisfied out of the whole cache network domain, there are two ways to deal with the response data packet. One is to send the response data packet to the HASH node first, where it can get cached, and then the HASH node forwards the data packet to the edge area. The other is to send the response data packet to the HASH node and the edge area simultaneously. Both methods have advantages and disadvantages. The former has smaller traffic overhead within the domain, but increases the delay experienced by users because of triangular routing. The latter one has shorter response time but it incurs more data transmission overhead. In this paper, we use the latter one.
IV. PERFORMANCE EVALUATION

We developed our simulation based on the ccnSim [31], which in turn is based on the OMNet++ simulation platform. ccnSim is an open source CCN simulator developed in ParisTech, which has implemented basic features of the CCN forwarding engine and caching components.

In our simulation, we set the number of content files to be 10000, and the file popularity obeys Zipf distribution. We assume there is only one content store located outside of the domain, which is connected to a core node. The link delay between each two cache nodes are the same, but the link delay between the content store and the core node is larger in order to simulate the fact that inter-domain links are often the performance bottlenecks. In ccnSim, each client represents a group of users. The request arrival from each client obeys Poisson distribution. We generated a different request sequence for each client in one simulation run, and feeded the same request sequences in the following simulation runs. We set the cache size of each node to be able to cache $0, 10^2, 10^2.5$ files in different simulation runs. LRU is used as the cache replacement policy for all nodes.

We compared the following cache coordination schemes: pure LCE in the whole domain (black line in all figures), pure LCD in the whole domain (red line), pure HASH-based cache coordination in the whole domain (blue line), Hybrid-HASH-with-LCE (green line) and Hybrid-HASH-with-LCD (pink line). In the following, we use four metrics to evaluate the performance: global hit rate, source load, average download time, and average download distance. All the results are gathered and calculated after the warm-up stage, i.e., when the cache system becomes stable. The global hit rate measures the ratio of requests satisfied by the whole cache network, i.e., the ratio of requests without having to go to the content store. Fig. 3 shows the global hit rates of the five cache coordination schemes. In most cases, the pure HASH-based cache coordination has the highest hit rate, which is due to the highest cache diversity of pure HASH-based coordination scheme. Both LCE and LCD have much lower global hit rates than the other three schemes because of relative low cache diversity. The hit rates of our hybrid schemes are only slightly lower than the pure HASH-based one, and the gap shrinks as the per-node cache size increases. The reason is that the collective cache size in the core area reserved for diversifying the cache content within the domain increases as well.

However, it is also observed that when the per-node cache size reaches $10^2.5$, the pure HASH-based cache coordination scheme is no longer superior in the global cache hit rate compared with our hybrid schemes. This is because that although pure HASH-based cache coordination maximizes the number of content items cached in the whole cache network domain, it does not guarantee to cache all popular contents. Depending on the hash function and the content name space, it is possible that a lot of popular content items be hashed to the same node, exceeding its cache capacity. Hence, some of these popular items has to be evicted from the whole cache network domain.

Fig. 4 reports the server load for different cache coordination schemes. Server load measures the total number of requests served by the content repository. As expected, the server load has a direct negative correlation with the global hit rate.

Fig. 5 shows the average download time of the five cache coordination schemes. The download time is measured by the duration from when the request is sent to when the file downloading is finished. It can be observed that although the pure HASH-based cache coordination has the highest global hit rate, it also has the highest download time, which is even higher than pure LCE. The reasons are two-folded. On reason is that the pure HASH-based cache coordination is an off-path cache coordination scheme, in which each request should be forwarded to the responsible HASH node first before being forwarded to the repository. In case of a cache miss, the
Fig. 5: Average download time of different cache coordination schemes.

Fig. 6: Average download hops of different cache coordination schemes.

overhead of triangular routing is high. The other reason is that each content only has at most one copy in the whole domain even if the content is very popular, so even with a cache hit in the whole domain, the content may still needs to be fetched from a node far from the user. Our hybrid based schemes, on the other hand, outperform either the pure implicit or pure explicit schemes. This demonstrates that hybrid schemes can make a good balance between the two somewhat contradictive goals. Essentially, in our hybrid scheme, the cache network is partitioned into two levels, with each level fulfilling different objectives. The upper level (i.e., the core area) is to increase the cache diversity and reduce inter-domain traffic, whereas the lower level (i.e., the edge areas) is to improve users’ QoE.

Fig. 6 reports the average download hops (for data only) for the five cache coordination schemes. As expected, it has a direct positive correlation with the average download time, the higher the download time, the longer the average download hops. Comparing with Fig. 3 and Fig. 4, where Hybrid-HASH-with-LCD and Hybrid-Hash-with-LCE show similar global hit rates and source load, the two schemes behaves differently in the average downloading time and average download hops. As expected, the Hybrid-HASH-with-LCD outperforms the Hybrid-HASH-with-LCE in users’ QoE, which is due to the fact that LCD takes content popularity into account, hence can more accurately identify popular content and pull them to the edge. Regarding the global hit rate and source load, although the Hybrid-HASH-with-LCD scheme shows higher diversity in the edge areas than the Hybrid-HASH-with-LCE scheme, the much larger logical cache in the core area smoothes this discrepancy.

V. Conclusion

This paper proposed a hybrid cache coordination scheme based on role division between cache nodes. It is a hybrid of on-path and off-path cache coordination, as well as a hybrid of explicit and implicit cache coordination. A large cache network is divided into one core area and several edge areas. In the core area, the HASH-based cache coordination is used to diversify the cached content within the whole domain, whereby in the edge area, implicit cache coordination policies are used to allow multiple copies of popular content to be cached near users. Simulation results show that our proposed hybrid cache coordination schemes have a good tradeoff between user experience and network resource utilization. The global hit rate as well as server load are comparable to that of the pure HASH-based cache coordination scheme, but the QoE is much better. The fact that different network parts can apply different cache coordination policies also makes this scheme potentially able to scale to large networks.

ACKNOWLEDGMENT

This work is partly supported by the National Natural Science Foundation of China under Grant Nos: 61100178, 61174152, 61303243, the Hi-Tech Research and Development Program of China under Grant No. 2013AA013503, the Natural Science Foundation of Jiangsu Province under Grant No. BK20141454 and the Jiangsu Future Networks Innovation Institute Prospective Research Project on Future Networks.

REFERENCES
