# DISTRIBUTED IN-NETWORK COOPERATIVE CACHING

**Xiaoyan Hu[1, 2], Jian Gong[1, 2]**

[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]Jiangsu provincial computer network technology key laboratory, Nanjing 210096, China
{xyhu,jgong}@njnet.edu.cn

**Abstract:** Named Data Networking (NDN) featuring in-network caching capability is a large effort that exemplifies information-centric approach to networking by shifting emphasis from hosts to data so as to meet growing demand on content. This work explores a scheme that enables a NDN domain to make full use of its in-network caches to enhance its performance, availability, and reliability. Currently, each NDN router independently determines what contents to cache and is unaware of content cached in nearby routers and thus their caches are not utilized in an efficient way. This paper proposes to have routers in a NDN domain share cached data and coordinate to make caching decisions (which is dubbed in-network cooperative caching) and formulates it into a constrained optimization problem. The Lagrangian relaxation and primal-dual decomposition method is applied to decompose the optimization problem into object placement subproblems and object locating subproblems, each of which can be solved in a distributed manner at each router, such that the in-network cooperative caching is addressed in a distributed way. Our simulation results, although preliminary, suggest that our scheme can benefit users, Internet Service Providers (ISPs) as well as content servers, and the improvement can be as much as *88%* compared to current NDN caching policy.

**Keywords:** In-network caching; Named data networking; Cooperative caching; Lagrangian relaxation

## 1 Introduction

To meet the growing demand of content[1], Information-Centric Networking (ICN) [1-3] shifts emphasis from hosts to data. Named data becomes addressable and routable, and are self-identifying and self-authenticating such that each data packet is potentially useful to many consumers and thus intermediate routers can cache data packets passing by (in-network caching) to serve future requests without resorting to original data sources.

Named Data Networking (NDN) [1] is a large effort that exemplifies the information-centric approach to networking. Currently, each NDN router independently determines what content to cache and is unaware of content cached in nearby routers. And thus caches at routers are not efficiently utilized. For example, there are two neighboring routers $R_A$ and $R_B$, and their caches both can store only one object. Both routers have received requests for two objects $o_1$ and $o_2$, and $o_1$ is more frequently requested. Due to the constraint of cache sizes and the unawareness of content cached at the other, they both employ Greedy Local (GL) cache strategy, i.e., choosing the more popular one, $o_1$, to cache, so that more requests can be locally served. In such case, the future requests for $o_2$ at both routers have to be routed to remote content publisher(s). Instead, if the two routers are aware of content cached by the other and coordinate in content caching and sharing, they can agree upon caching $o_1$ and $o_2$ separately so that all future requests for both objects can be served by their caches.

So how a NDN domain makes use of its in-network caches would play an important role in enhancing its performance, availability, and reliability. More specifically, a domain can utilize the caching capacity in its routers to cooperatively store inbound traffic so as to improve its content delivery performance and reduce its upstream bandwidth usage, which is dubbed in-network cooperative caching here. This work treats it as an optimization problem and applies Lagrangian relaxation and primal-dual decomposition to decompose it into subproblems so as to solve it in a distributed way. Our preliminary results suggest the effectiveness of our scheme and note that the concepts and algorithms proposed in this paper could apply to almost any other ICN [2, 3].

The rest of the paper is organized as follows. Section 2 presents the related work and Section 3 formulates the in-network cooperative caching problem. In Section 4, a conventional method, Lagrangian relaxation and primal-dual decomposition is applied to conduct the cooperative caching in a distributed way; and implementation specific issues are discussed. In Section 5, simulations are conducted to demonstrate the effectiveness of distributed in-network cooperative caching. And finally Section 6 concludes our work and closes with future work.

## 2 Related work

There has been a large body of literature on collaborative caching [4-6]. Most consider an overlay model where collaborative caching is treated as an overlay service independent from the underlay networks, and they have limitations to be applied in NDN directly. This work either focuses on special-purpose applications which put additional constraints on the design (e.g., P2P system), or requires the system to be constructed as a particular type of topology, e.g., a

1 We will use “content”, “data” and “object” interchangeably.

multicast tree. Extensive calculation is often required, which limits their usage in global environment.

At the end of the 20th century, the caching architecture called en-route [7, 8] was developed in which web caches are associated with routing nodes in the network and are referred to as en-route caches. In this respect, these en-route caches are similar to caches built in NDN routers and farsighted Wang *et al.* [9] proposed a scheme to dynamically place objects in the caches on the path from the server to the client in a coordinated fashion to maximize cost saving. It is content servers that make the caching decision and thus it is application specific such that it requires special configuration at applications. To the best of our knowledge, there are only a few works on the in-network cooperative caching problem. Ref. [10] formulated the cooperative in-network caching problem into Mixed-Integer Linear Programming problem, but the exact solution was not discussed at all. Li *et. al* [11] proposed to have nearby routers cooperate in caching to avoid these routers storing the same content which improves cache hit ratio. But the dissemination of content in this model considers nothing about content popularity at routers which may not maximize cost saving. Then Cho *et al.* [12] proposed popularity-based and collaborative in-network caching in which an upstream router recommends the number of chunks to be cached at its downstream router and the number exponentially increases as the request count (the indicator of popularity) increases. But there are still respectful redundancies and content servers are involved in the recommendation.

Based on these works, we assume that content caching is an inherent underlay capability and only routers are involved in the cooperative caching procedure such that existing and future applications would benefit from caching without requiring specific configurations. We consider an underlay, non-structured flat network model where any router can be the caching parent of any other router and propose a distributed cooperative caching decision making process taking content popularity and access costs between routers into account.

## 3 Problem Formulation

This section formulates the in-network cooperative caching problem and maps it into an optimization problem in a graph as follows.

**Graph construction:** the NDN router-level topology of a domain is represented as an undirected graph $G = (V, E, d)$. $V = \{v_1, v_2, \cdots, v_N\}$ is the set of routers ($|V| = N$). $E$ is the set of edges (links). Function $d: V \times V \rightarrow R$ defines the access cost between any two routers; the access cost $d(i, j)$ between routers $v_i, v_j \in V$, $v_i \neq v_j$, can be explained as the shortest hop count or the minimum access delay if there is a path between them (i.e., the two routers are connected), otherwise $d(i, j) = +\infty$; and $\forall v_i \in V$, $d(i, i) = 0$. The cache size of router $v_i \in V$ is denoted by $C_i$; the set of objects to be stored is represented by $O = \{o_1, o_2, \cdots, o_K\}$ ($|O| = K$); and the demand for object $o_k \in O$ (with size $s^k$) at router $v_i \in V$ is denoted by $r_j^k$. We assume that the origin of these objects is an upstream content server $v_{N+1}$ outside the domain and $\forall v_i \in V, d(i, N+1) = D$ where $D$ is some positive constant. And note that $\forall v_i, v_j \in V$ which are connected, $d(i, j) \ll d(i, N+1)$ i.e., the cost of accessing objects from the cache of a router in the domain is much smaller than that from the outside origin (e.g., fetching data from the cache of another router in the domain may reduce latency, load on potentially expensive upstream links, and so forth).

**Object placement**: we use a series of binary variables $x_j^k$ s to describe whether router $v_j$ caches object $o_k$.

**Object access/locating**: while NDN naturally supports multi-path routing, we assume each router chooses one path for each object access. An object access relies on the location of the object. More particular, upon receipt of a request for object $o_k$, router $v_i$ serves the request from its local cache if $o_k$ is locally cached, otherwise from some other router if the router caches $o_k$, otherwise ($o_k$ is not cached in the domain) from its data origin. We use a series of binary variables $y_{ij}^k$ s to describe whether router $v_i$ access object $o_k$ from router $v_j$.

**Objective**: the objective of in-network cooperative caching is to intelligently store these objects in the routers of the domain and share them, i.e., determine object placement $x_j^k$ s and object access $y_{ij}^k$ s, in such a way that the caching gain is maximized, i.e., maximizing the reduction in total access cost for these routers to fetch requested data as compared to that without caching. And the objective is formulated as follow:

$$\text{Maximize} \sum_{v_i \in V} \sum_{o_k \in O} \sum_{v_j \in V} y_{ij}^k (D - d(i, j)) r_i^k s^k \quad (1)$$

$$\text{Subject to: } y_{ij}^k = \{0,1\} \quad \forall v_i \in V, v_j \in V, o_k \in O \quad (2)$$

$$x_j^k = \{0,1\} \quad \forall v_j \in V, o_k \in O \quad (3)$$

$$\sum_{v_j \in V} y_{ij}^k \leq 1 \quad \forall v_i \in V, o_k \in O \quad (4)$$

$$y_{ij}^k \leq x_j^k \quad \forall v_i \in V, v_j \in V, o_k \in O \quad (5)$$

$$\sum_{o_k \in O} x_j^k s^k \leq C_j \quad \forall v_j \in V \quad (6)$$

The constraint in Eq. (4) characterizes the fact that if object $o_k$ is cached in the domain; if yes, it should be fetched from one of those routers hosting it ($\sum_{v_j \in V} y_{ij}^k = 1$), otherwise ($\sum_{v_j \in V} y_{ij}^k = 0$) from the origin. The constraint in Eq. (5) represents that router $v_i$ can access object $o_k$ from router $v_j$ if and only if $v_j$ caches $o_k$ ($v_i$ and $v_j$ can be the same router). And the constraint in Eq. (6) guarantees that the content caching at each router subjects to storage capacity constraint.

# 4 Distributed in-network cooperative caching

## 4.1 Distributed algorithm

The above in-network cooperative caching problem is difficult to solve in a centralized way due to its complexity. To achieve a distributed solution with heterogeneous settings of input parameters, this work applies a conventional method Lagrangian relaxation and primal-dual decomposition [13]. We firstly rewrite the constraint (5) to be as follow:

$$y_{ij}^k r_j^k s^k \le x_j^k r_j^k s^k \quad \forall v \in V, v \in V, o \in O \qquad (7)$$

which is then incorporated into the objective unction in Eq. (1) by associating a Lagrangian multiplier $\eta_{ij}^k$. Then the Lagrangian dual problem is represented as:

$$\begin{aligned} &\text{Minimize } L(\eta_{ij}^k) \\ &\text{Subject to: } \eta_{ij}^k \ge 0 \end{aligned} \qquad (8)$$

And the objective function $L\left(\eta_{ij}^k\right)$ in the dual problem is:

$$\begin{aligned} L(\eta_{ij}^k) &= \max \sum_{v_j \in V} \sum_{o_k \in O} \sum_{v_i \in V} \eta_{ij}^k r_j^k s^k x_j^k \\ &+ \sum_{v_j \in V} \sum_{o_k \in O} \sum_{v_i \in V} y_{ij}^k s^k \left( (D - d(i,j)) r_i^k - \eta_{ij}^k r_j^k \right) \end{aligned} \qquad (9)$$

The Lagrangian dual problem can then be decomposed into |V| object placement subproblems and |V| × |O| object locating subproblems which both can be solved in a distributed manner at each router. $\sum_{v_i \in V} \eta_{ij}^k$ reflects the interest of router $v_j \in V$ to store a particular content $o_k \in O$ and $\eta_{ij}^k$ reflects the interest of router $v_i \in V$ to access content $o_k \in O$ from router $v_j \in V$. At each iteration *t*, router $v_i \in V$ first solves the following object placement subproblem:

$$\begin{aligned} &\text{Maximize } \sum_{o_k \in O} \left( \sum_{v_i \in V} \eta_{ij}^k \right) r_j^k s^k x_j^k \\ &\text{Subject to: } x_j^k = \{0,1\} \ \forall o_k \in O \\ &\qquad \sum_{o_k \in O} x_j^k s^k \le C_j \end{aligned} \qquad (10)$$

Objects can be divided into equal sized NDN Data packets for convenience. This is the classical 0−1 knapsack problem and the optimal solution of this object placement problem is:

$$x_j^k(t) = \begin{cases} 1 \text{ for } k \in [1, z), \\ 0 \text{ for } k \in [z, |O|]. \end{cases} \qquad (11)$$

In solution (11) the object set *O* is sorted in descending order by the critical index $\left( \sum_{v_i \in V} \eta_{ij}^k \right) r_j^k$ and $z = \min\{h \mid \sum_{i=1}^{h} s^k > C_j\}$. Router $v_j$ is responsible to broadcast the |O| vector $X_j = \left( x_j^1, \cdots, x_j^k \cdots, x_j^K \right)$, i.e., its temporary placement decisions, to other routers in the network. Given the placement decisions at all routers, each router $v_i$ is able to solve the locating subproblem for individual object $o_k$ such that:

$$\begin{aligned} &\text{Maximize } y_{ij}^k s^k ((D - d(i,j)) r_i^k - \eta_{ij}^k r_j^k \\ &\text{Subject to: } y_{ij}^k = \{0,1\} \ \forall v_j \in V \\ &\qquad \sum_{vj \in V} y_{ij}^k \le 1 \\ &\qquad y_{ij}^k \le x_j^k(t) \end{aligned} \qquad (12)$$

We denote $\zeta_{ij}^k = \left( D - d\left(i,j\right) \right) r_i^k - \eta_{ij}^k r_j^k$. Then the optimal locating solution for object $o_k$ at router $v_i$ is as follow:

$$y_{ij}^k(t) = \begin{cases} 1 \text{ for } v_j \text{ with } \zeta_{ij}^k = \max\{\zeta_{ij}^k \mid v_j \in V, x_j^k(t) = 1\}, \\ 0 \text{ otherwise.} \end{cases} \qquad (13)$$

In solution (13), router $v_i$ firstly finds the router set $V^k = \{v_j \mid v_j \in V,\ x_j^k\left(t\right) = 1\}$ and then sorts routers of set $V^k$ in descending order by the critical index $\zeta_{ij}^k$.

With the temporary local object placement and locating decisions, each router $v_i$ is responsible to update values of its $\eta_{ij}^k$ s for the next iteration using sub-gradient method [13, 14]:

$$\eta_{ij}^k(t+1) = \eta_{ij}^k(t) - \theta(t)(x_j^k(t) - y_{ij}^k(t)) r_j^k s^k f(d(i,j)) \qquad (14)$$

where $\theta(t) = 1/t$ is the step-size and $f()$ is positively correlated to $d(i,j)$, the access cost between routers $v_i$ and $v_j$. From the update rule of $\eta_{ij}^k$, it can be seen that when $x_j^k(t) = y_{ij}^k(t)$, either 1 or 0, $\eta_{ij}^k$ does not change in the $t+1$ iteration indicating that the placement of object $o_k$ at router $v_j$ is useful to the access of object $o_k$ at router $v_i$ and thus in the next iteration, the chance of the decisions for object $o_k$ to change should be less. Otherwise if $x_j^k(t) > y_{ij}^k(t)$, then $\eta_{ij}^k(t+1)$ decreases (the decrease is proportional to the access cost between routers $v_i$ and $v_j$ ) such that in the next iteration, the chance of object $o_k$ being cached at router $v_j$ is less as not many routers rely on this copy. The above update rule of $\eta_{ij}^k$ ensures the quick convergence of the proposed algorithm.

The above mentioned Distributed In-networking Cooperative Caching (DICC) at router $v_i \in V$ is summarized in Algorithm 1 and Figure 1 shows how router $v_i$ interacts with others.

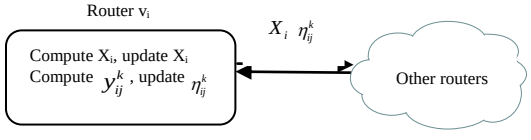



**Figure 1** Router $v_i$ interacts with other routers

Algorithm 1: Distributed In-network Cooperative Caching

Input:: $d(i,j), s^k, r_i^k, C_i$

Output: $x_i^{k*}, y_{ij}^{k*}$

1) Initiate t=0 and coefficient $\eta_{ij}^k(0)$ to some positive value, $\forall v_i, v_j \in V, o_k \in O$

2) Iterate until coefficient $\eta_{ij}^k$ converge to $\eta_{ij}^{k*}$:

(a) Calculate $x_i^k(t)$ according to equation (11), $\forall o_k \in O$

(b) Broadcast placement decision vector $X_i(t)$ snapshot from other routers

(c) Receive placement decision vectors $X_j(t)$ snapshot from other routers, and $\forall v_j \in V$ and $v_j \neq v_i$

(d) Calculate $y_{ij}^k(t)$ according to equation (13), $\forall v_j \in V, o_k \in O$

(e) Update coefficient $\eta_{ij}^k$ according to equation (14) and broadcast them to others, $\forall v_j \in V, o_k \in O$

3) Obtain the near optimal solution as $x_i^{k*} = x_i^k(t)$ and $y_{ij}^{k*} = y_{ij}^k(t)$

### 4.2 Complexity analysis and implementation specific issues

In the DICC procedure, when a router $v_i$ determines its object placement and object locating, it requires information of the access costs between other routers in the domain and itself, Lagrangian multiplier $\eta_{ij}^k$ and placement decision vector snapshot $x_j^k$ at other routers. The access cost information at each router should be relatively stable (updates only if there is network dynamic) and takes space *O(N )*. In each iteration, routers make the object placement or object locating decisions in parallel. For local object placement, router $v_i$ computes the critical index for each object (taking time and space *O(K)*), sorts objects to make caching decisions (taking time *O(KlgK)*), broadcasts $X_i$ and receives $X_j$*s* from others (taking time *O(2(N − 1)K)* and space *O((N − 1)K)*) which altogether consume time *O(K+KlgK+2(N−1)K)* and space *O(NK)*; for object locating and Lagrangian multiplier $\eta_{ij}^k$ update, the computation of $\zeta_{ij}^k$ and the pick of the largest $\zeta_{ij}^k$ (taking time and space *O(NK)*), the update of $\eta_{ij}^k$ (taking time and space *O(NK)*), the broadcast of $\eta_{ij}^k$ and the receipt of the Lagrangian multipliers at others (taking time *O(2NK(N−1))* and space *O(NK(N−1))*) altogether take time $O(2N^2K)$ and space *O(NK(N + 1))*. So at each router, the required space is upper bounded by *O(N+N K(N+2))*; at each iteration, the time is upper bounded by $O(KlgK+(2N-1)K+2N^2K)$.

Due to space limit, we only concisely mention the implementation of information exchange process. In DICC, while routers make decisions in parallel, when exchanging information, routers should be arranged in some order so that the information report (about object placement decision snapshot or object locating decisions) at these routers are sequentially processed. More specifically, routers request information at other routers one by one, i.e., each router firstly requests information at the router with order *1*, then that with order *2* and so on (the name of the Interest packet (request) for information at a specific router should include the ID of the asked router, iteration # and information type, either object placement decision snapshot or object locating decisions). In this way, the Interests for the information at the same router are likely to be aggregated in intermediate routers [1] or be served with copies in content stores of intermediate routers such that the asked router is less involved in the information exchange. Note that any $X_i$ can be succinctly represented as a space efficient (Delta) Bloom Filter [15, 16] to reduce communication cost.

## 5 Performance evaluation

In this section, we evaluate the DICC through simulations. We conduct simulations on two practical ISP topologies, the topologies of AS *209* and AS *7018* from Ref. [17], and the basic information of the two topologies are summarized in Table I (we set D, the delay for any PoP in the two ASes to access an object from original data sources, to be 130ms). The PoP-level topologies are different from the real router-level topologies; however, they still demonstrate the scope and the effectiveness of DICC. In our simulations, we treat PoP nodes in the underlying network as routers with both routing and caching capability, and have them make object placement and locating decisions using DICC strategy.

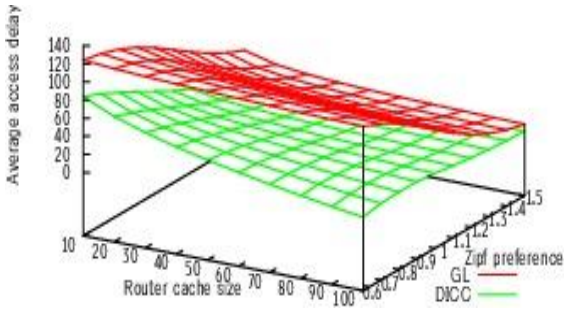
TABLE I: The topology information of ASes 209 and 7018.

| AS No. | # of PoPs | # of edges | Avg delay bw PoPs (ms) |
|---|---|---|---|
| 209 | 58 | 108 | 17.42 |
| 7018 | 115 | 148 | 11.69 |

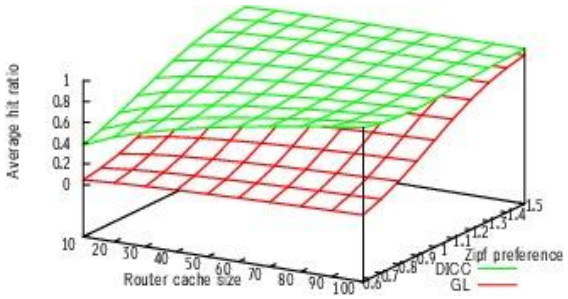
The numbers of objects that may be requested by routers are *1000* and *10000* in AS *209* and in AS *7018* respectively. We assume that routers in an AS are provisioned with caches of equal size and their demands on objects follow Zipf distribution with shape parameter $\alpha$ (Zipf preference), i.e. at router $v_i$, the object with rank $k$ is requested with rate $r_i^k = k^{-\alpha} / \sum_{l=1}^{|O|} l^{-\alpha}$.

We evaluate the effectiveness of DICC by comparing average access delay (the average delay for a router to fetch an object) and average hit ratio (the percentage of content served by caches of routers in a domain) under DICC with that if routers independently manage their own caches (i.e., under GL) when cache size and Zipf preference $\alpha$ are configured with different values. We

summarize the results and show them in Figure *2* and Figure 3. Figure *2* plots how the average access delay and average hit ratio vary with the change of cache size and Zipf preference; and due to space limit, we only report the results in AS 7018 (trends are consistent with results in AS 209). Figure 3 shows the average access delay and average hit ratio over all simulations (different simulations are with different cache sizes and Zipf preferences) in the two ASes.

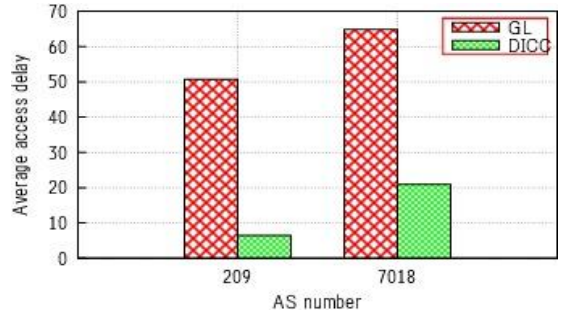



**(a)** Delay

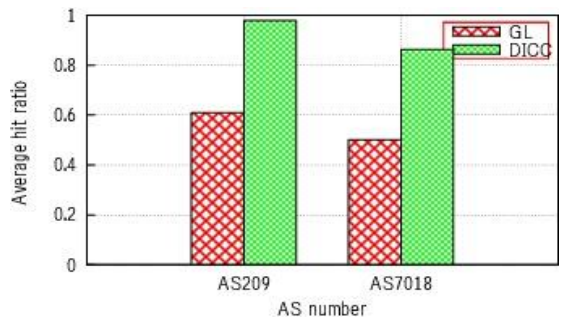



**(b)** Hit ratio

**Figure 2** Simulation results for AS 7018




**(a)** Delay




**(b)** Hit ratio

**Figure 3** Simulation results

From Figure 3, it can be seen that in AS 209, DICC reduces average access delay from *51ms* to *6ms* (about *88%* reduction) and improves average hit ratio from *0.61* to *0.98* (about *60%* improvement); and in AS *7018*, DICC reduces average access delay from *65ms* to *21ms* (about *68%* reduction) and improves average hit ratio from *0.50* to *0.86* (about *72%* improvement). The results demonstrate that with DICC, NDN domains can actually further improve its packet delivery performance (i.e., smaller delay and higher availability) and reduce its upstream bandwidth usage. Meanwhile, due to higher hit ratio, more request traffics are served within domains and thus the traffics in backbone network and the workload of content servers are reduced.

Figure *2* illustrates that DICC is especially effective when Zipf preference $\alpha$ is small. For example, when $\alpha$ = *0.7* (a typical Zipf preference in Web traffic popularity [18]) and cache sizes are set to as large as 100, DICC reduces average access delay from *103ms* to *14ms* and improves average hit ratio from *0.2*1 to *0.94*. But when Zipf preference increases, the demands on objects of routers become more heavy-tailed such that the objects additionally cached in the domain (due to cooperative caching) are accessed at smaller rates by routers in the domain and thus the effectiveness of the DICC caching strategy is less obvious.

## 6 Conclusions & future work

This work explores a scheme that enables a Named Data Networking (NDN) domain to make full use of its in-network caches to enhance its performance, availability, and reliability. Due to the growing demand on content, information-centric networking featuring routing by name and universal in-network caching capability is proposed as an alternative of IP and NDN is a large effort that exemplifies the information-centric approach to networking. Currently, each NDN router independently determines what content to cache and is unaware of content cached in nearby routers and thus their caches are not utilized in an efficient way. Based on the observation, this paper proposes to have routers in a NDN domain share their cached data and coordinate to determine what objects to cache at each router, which is called in-network cooperative caching. The in-network cooperative caching problem is formulated into a constrained optimization problem and the Lagrangian relaxation and primal-dual decomposition method is then applied to decompose the optimization problem into object placement subproblems and object locating subproblems, each of which can be solved in a distributed manner at each router, such that the in-network cooperative caching is addressed in a distributed way. Our simulation results, although preliminary, suggest that our scheme can benefit users, ISPs as well as content servers, and the improvement can be as much as *88%* compared to current NDN caching policy. And our next step is to develop more realistic models adapting to NDN, conduct comprehensive evaluations to quantify the

benefits, and efficiently implement our scheme in the realistic NDN environment, which is challenging.

# Acknowledgements

We are grateful to anonymous reviewers of the 2nd IEEE CCIS2012 International Workshop on Future Internet Technologies and look forward to getting feedbacks from you all. This work was sponsored by the National Grand Fundamental Research 973 program of China under Grant No.2009CB320505 and the National Nature Science Foundation of China under Grant No. 60973123.

# References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. Braynard, "Networking named content," in CoNEXT', 2009.

[2] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," SIGCOMM Comput. Commun.Rev.,vol.37, pp.181–192, 2007.

[3] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems - Volume 3, USITS'01, (Berkeley, CA, USA), pp. 4–4, USENIX Association, 2001

[4] J. Ni and D. H. K. Tsang, "Large-Scale Cooperative Caching and Application-Level Multicast in Multimedia Content Delivery Networks," IEEE Communications Magazine, vol. 43, pp. 98–105, May 2005.

[5] P. Sarkar and J. H. Hartman, "Hint-based cooperative caching," ACM Trans. Comput. Syst., vol. 18, pp. 387–419, Nov. 2000.

[6] H. Che, Z.Wang, and Y. Tung, "Analysis and design of hierarchical web caching systems," in INFOCOM, pp.1416–1424, 2001.

[7] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura, "Self-organizing wide-area network caches," in Proceedings of IEEE INFOCOM'98, (San Francisco, CA, USA), pp. 600–608, IEEE Press, March 1998.

[8] P. Rodriguez and S. Sibal, "Spread: scalable platform for reliable and efficient automated distribution," Comput. Netw., vol. 33, pp. 33–49, June 2000.

[9] X. Tang and S. T. Chanson, "Coordinated en-route web caching," IEEE Trans. Comput., vol. 51, pp. 595–607, June 2002.

[10] S. Wang, J. Bi, J. Wu, Z. Li, W. Zhang, and X. Yang, "Could in-network caching benefit information-centric networking?," in Proceedings of the 7th Asian Internet Engineering Conference, AINTEC '11, (New York, NY, USA), pp. 112–115, ACM, 2011.

[11] Z. Li and G. Simon, "Time-shifted tv in content centric networks: The case for cooperative in-network caching.," in ICC, pp. 1–6, IEEE, 2011.

[12] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "Wave: Popularity-based and collaborative in-network caching for content- oriented networks," in Proceedings of IEEE INFOCOM'12 Workshop on Emerging Design Choices in Name-Oriented Networking, NOMEN'12, (Orlando, Florida, USA), IEEE Press, March 2012.

[13] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," Selected Areas in Communications, IEEE Journal on, vol. 24, pp. 1439–1451, Aug. 2006.

[14] D. P. Bertsekas, A. Nedi´, and A. E. Ozdaglar, Convex Analysis and Optimization. Athena Scientific, 2003.

[15] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," Commun. ACM, vol. 13, pp. 422–426, July 1970.

[16] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," Internet Mathematics, vol. 1, no. 4, 2005.

[17] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," in Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '02, (New York, NY, USA), pp. 133–145, ACM, 2002.

[18] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in INFOCOM (1), pp. 126–134, 1999 .