A Light-weight Content Distribution Scheme for Cooperative Caching in Telco-CDNs

Takuma Nakajima, Masato Yoshimi, Celimuge Wu, Tsutomu Yoshinaga
Graduate School of Information Systems
The University of Electro-Communications
Chofu-shi, Tokyo, Japan
Email: tnakajima@comp.is.uec.ac.jp, {yoshimi,clmg,yosinaga}@is.uec.ac.jp

***Abstract*—A key technique to reduce the rapid growing of video-on-demand's traffic is a cooperative caching strategy aggregating multiple cache storages. Many internet service providers have considered the use of cache servers on their networks as a solution to reduce the traffic. Existing schemes often periodically calculate a sub-optimal allocation of the content caches in the network. However, such approaches require a large computational overhead that cannot be amortized in a presence of frequent changes of the contents' popularities. This paper proposes a light-weight scheme for a cooperative caching that obtains a sub-optimal distribution of the contents by focusing on their popularities. This was made possible by adding color tags to both cache servers and contents. In addition, we propose a hybrid caching strategy based on Least Frequently Used (LFU) and Least Recently Used (LRU) schemes, which efficiently manages the contents even with a frequent change in the popularity. Evaluation results showed that our light-weight scheme could considerably reduce the traffic, reaching a sub-optimal result. In addition, the performance gain is obtained with a computation overhead of just a few seconds. The evaluation results also showed that the hybrid caching strategy could follow the rapid variation of the popularity. While a single LFU strategy drops the hit ratio by 13.9%, affected by rapid popularity changes, our proposed hybrid strategy could limit the degradation to only 2.3%.**

***Keywords*-cooperative caching, sub-optimal content placement, hybrid caching, dynamic content popularity.**

I. Introduction

Video-on-Demand (VoD) services generate enormous Internet traffic, where the video contents will contribute to more than 80 percent of the total traffic by 2020 [1]. Such tremendous traffic, not only degrades the user experience due to the congested links, but also increases the communication costs, such as the power consumption. Although Content Delivery Networks (CDNs) have reduced such traffic by caching videos [2], their corresponding cache servers are usually located outside of the Internet Service Provider (ISP) networks [3]. This implies that CDNs cannot reduce traffic on peering links between ISPs and CDNs, leading to many congested links. Even though CDN providers tried to locate their cache servers directly into the ISP networks, they still have no global knowledge about the network's physical properties, which is a key factor to effectively manage any given traffic. Several ISPs have considered building Telco-CDNs which are CDNs managed by ISPs rather than global CDN providers [3]. In Telco-CDNs, cache servers are directly located in the ISPs' backbone networks and managed by ISPs to efficiently reduce the traffic [3], [4]. In such scenario, ISPs can handle many cache servers with a full knowledge of the underlying networks, which makes possible the use of cooperative caching strategies. Such strategies usually allocate several groups of cache servers to aggregate storages and increase the availability of the contents cached in the network. However, it is considered as a challenging task to find the ideal mapping between several contents into several cache servers since such problem is NP-complete [3]. In fact, an inefficient allocation of contents may increase the internal traffic leading to many congested links [3].

Previously proposed cooperative caching schemes usually adopt an optimization approach and calculate a sub-optimal allocation of contents using heuristic techniques, such as a Genetic Algorithm (GA). However, the use of such techniques is always associated with a heavy computation time, even when using a cluster [3]. Moreover, the popularity of the contents changes frequently because of the insertion of new contents, viral activities, and influential news [5], [6]. Thus, there is a need to update the content allocations, which may still cause a massive traffic to the upstream CDN servers and often to the origin ones due to long update intervals.

Since 20% to 60% of the contents' popularities change every hour [6], the long calculation time causes mismatches in the content allocations that generates additional traffic. Hence, a light-weight scheme to find an efficient allocation is required. This paper focuses on the characteristics of the sub-optimal allocations of contents. The goal is to find an efficient allocation of contents with a minimum computation time overhead.

We first calculate a sub-optimal allocation that minimizes the traffic. We find out that the number of contents in the network decreases along with their popularities, which indicates that the density of contents is an important factor for its allocation. Starting from this fact, we propose a light-weight scheme for a cooperative caching that allocates contents taking into consideration their densities. The proposed

scheme adds colored tags to all cache servers, and attributes the same color to contents to be cached. In addition, we propose a hybrid caching strategy based on Least Frequently Used (LFU) and Least Recently Used (LRU) schemes, which efficiently manages the contents even with frequent changes in the popularity.

Evaluation results on a realistic topology showed that our light-weight scheme could reduce the traffic efficiently, reaching a sub-optimal performance. In addition, we demonstrate that this process would take only a few seconds of computation time, in contrast with conventional heuristic methods. Moreover, the adopted hybrid caching strategy could follow the rapid change in the content popularity.

The rest of the paper is organized as follows. Section II describes existing cooperative caching schemes and the video popularity's characteristics. In section III we first present a sub-optimal content allocations calculated by a GA (Genetic Algorithm), and then we describe how we allocate contents with the minimum computation time. The implementation of the proposed caching scheme is discussed in section IV. In section V, we evaluate the traffic reduction, cache hit ratio in rapid popularity change, and the computation time overhead using three different network topologies. Before we conclude the paper in section VII, we discuss in section VI the number of colors used in our scheme and how to further improve the contents' coloration.

II. Related Work

Existing cooperative caching strategies can be classified into centralized and distributed approaches.

A. Centralized Approaches for Cooperative Caching

Centralized approaches introduce a management server that computes an efficient content allocation using access logs gathered from cache servers [3], [7]. These approaches mainly focus on optimizing the network's metrics rather than find out the important factors of the optimal allocations. In fact, centralized schemes formulate an optimization approach based on multiple network constraints, such as the network topology, power consumption, link capacity, required bandwidth, and latency. Actually, it is hard to solve such complex problems including many constraints, since such problems are formed to Capacitated Facility Location Problem which is an NP-complete [3]. In centralized schemes, the sub-optimal allocation is found by using heuristic approaches, such as GA. Although such technique could find the sub-optimal allocations, it often takes more than 10 hours of computation overhead, even when using a cluster.

B. Distributed Approaches for Cooperative Caching

Distributed approaches [4], [8] share several lists of cached contents and access logs with adjacent servers. Each server decides whether to cache a content or not, based on the shared information, to increase the number of contents in the network. In [4], cache servers share content lists and search a requested content in a given network before fetching it from the origin server. In [8] cache servers search contents in k-hops before storing the requested contents. In [9], the authors propose to reform the optimization problem, and solve it in a distributed manner, by ignoring several constraints. Each cache server gathers access logs from nearby servers and estimates the resulting traffic reduction to decide whether to cache the contents or not.

Although these techniques can reduce the number of requests to the origin server, they often require a particular topology such as a ring [8] and a tree [4]. Unfortunately, these topologies are not compatible with the mesh-based network adopted by most of the ISPs' backbone networks [3]. Using overlay networks could solve the problem, but some direct links between cache servers cannot be covered. In terms of optimization, it is difficult to optimize the network taking into consideration all constraints, when using distributed approaches. Moreover, they often require complex operations for sharing information and calculations for content caching.

C. Following Rapid Changes in Content Popularity

Video popularity often change rapidly, influenced by the insertion of new contents, viral activities on Social Networking Service (SNS) such as Twitter and Facebook, and influential news [5], [6]. A hybrid caching strategy based on LFU and FIFO is proposed to follow rapid changes in popularity [10]. The introduction of the hybrid caching aims to increase hit ratio by using LFU caches and follow the variation in popularity by using FIFO caches. There are also other techniques like the prefetching technique [3], [4]. Such schemes push the popular contents to all cache servers just after their insertion into the origin server. Although using the prefetching approach could improve the caching performance, it is not always effective since a content's popularity occasionally increases due to viral communications on SNS or unexpected news.

Although we also propose a hybrid caching strategy, our caching scheme cooperates with nearby servers rather than working based only on servers' individual access logs [10]. Although existing hybrid strategies assume an environment with a single cache server, most of the ISPs intend to distribute cache servers in the network [3]. Moreover, since existing hybrid strategies do not take into consideration the collaboration between neighboring servers, many contents will be duplicated across the network, which will generate additional traffic. Therefore an efficient hybrid scheme is required that cooperates with other servers for further traffic reduction.

III. Proposed Light-weight Caching Strategy

A. Light-weight Approach for Cache Distribution

Our proposal is based first on a preliminary calculation of the sub-optimal allocation using a GA, followed by an anal-

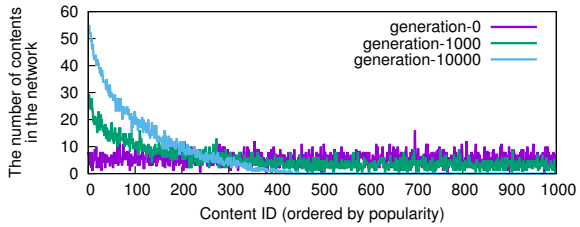

Figure 1. The number of contents in the network calculated by GA.

ysis of the obtained results. The configuration parameters of the calculation are shown in Table I. The content popularity is defined using a gamma distribution, which is a more realistic approach than the well-known Zipf for the VoD accesses [11]. The calculation follows the same approach as in [3], but without taking into consideration the bandwidth and latency constraints.

Fig. 1 shows the number of contents in the network along with different generations. The generation-0 distributes the contents randomly, which is an initial state of the calculation. The next generation is produced based on the current generation, to minimize the traffic generated by user accesses. In the generation-1000, the number of contents biases along with their popularities. In the generation-10000, which comes after the convergence, the bias gets stronger than the previous generation-1000. Some popular contents are cached in all servers, while unpopular ones are eliminated from the network. These results show that the important factor for the traffic reduction is the density of the content distribution.

In order to distribute contents with minimum computation overhead, our light-weight approach preliminarily colorizes both cache servers and contents. Each cache server in the network is preliminarily colorized with a specific color like the four-color theorem. It may take a long time to compute an efficient coloration of servers. However, this computation overhead can be amortized since the topology of the backbone network does not change frequently. Each server caches contents only when the content's color tag matches its color, which reduces duplicated contents in the network. In addition, to increase hit rates of each server, we added multiple colors to popular contents. These tags are updated periodically along with their latest popularities. Each tag has several bits, and its length corresponds to the number of available colors. The popularity classes are classified by the number of 1-bit in their tags' bits. For

Table I
GA SIMULATION PARAMETERS.

Total content	1000
Cache capacity	100
Popularity distribution	Gamma distribution
Gamma parameter k	0.475
Gamma parameter θ	170.6067
Topology	NTT core network [12]
Number of servers	55

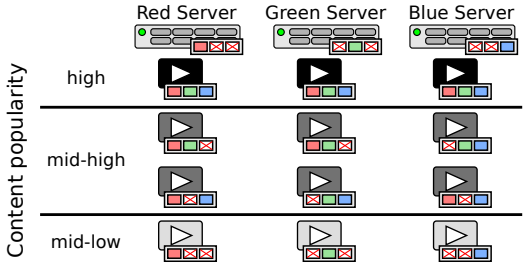

Figure 2. Example of contents cached in three servers according to their color tags and popularities.

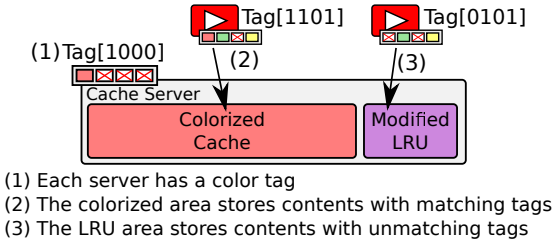

Figure 3. Proposed LFU-LRU Hybrid Cache Architecture.

example, contents with the highest popularity class will have tags having all bits equal to 1. The second class has tags with a single bit equal to 0. Fig. 2 shows an example with three colors. The red server caches only contents with red color tags. Blue and green servers do the same. If the servers' colors are uniformly distributed in the network, the density of content distribution decreases along with the popularity of each content, like the generation-10000 in Fig. 1.

B. Cooperative LFU-LRU Hybrid Caching Strategy

It is not easy to follow the rapid variation in content popularity by only using colored caches since the color tags of the contents are updated periodically. Each cache server has separated storage that is managed with LFU and LRU to follow such rapid changes. Fig. 3 shows the architecture of the proposed LFU-LRU hybrid caching strategy. The proposed algorithm for managing cache areas is shown in Algorithm 1. In contrast to the existing hybrid scheme, it checks the content colors before caching (lines 8-16). The LFU area caches the contents that have the same color tags as the servers for the efficient cache distribution, while the LRU area caches the rest of the contents to follow the rapid changes in the content popularity. Moreover, the LRU area increases the hit ratio by caching contents with relatively high popularity without matching the tags. In addition, we adopt a modified version of LRU [4] that improves hit ratio by inserting new contents into a few rank up from LRU position rather than Most Recently Used (MRU).

In contrast to the existing hybrid strategy, our scheme can efficiently distribute contents over the network since cache servers manage the LFU area with color tags that are calculated based on globally gathered access logs. Therefore, the wide variety of contents in the network improves hit

Algorithm 1 Request handling with hybrid caching scheme

```
1: if requested content exists in the LFU area then
2:     content ← fetchFromCache(key, LFU)
3: else if requested content exists in the LRU area then
4:     updateRank(key, LRU)
5:     content ← fetchFromCache(key, LRU)
6: else
7:     content ← fetchFromOrigin(key)
8:     if (colorbit(content) & colorbit(server)) ≠ 0 then
9:         while availableSize(LFU) < sizeof(content) do
10:            if LFU has a content without the same color then
11:                evict it from LFU area
12:            else
13:                evict the least popular content from LFU area
14:            end if
15:        end while
16:        insert(content, LFU)
17:    else
18:        while availableSize(LRU) < sizeof(content) do
19:            evict the oldest content from LRU area
20:        end while
21:        insert(content, LRU)
22:    end if
23:    return content
24: end if
```

ratios and reduces the traffic, not only between internal links but also between peering ones that connect the network to the upstream CDN servers.

IV. Implementation of the Proposed Caching Scheme

In order to implement the proposed caching scheme, we use four colors (R, G, B, Y) to colorize the cache servers and their contents. We first implemented a colorization algorithm for cache servers using the Welsh-Powell algorithm [13], which is a well-known algorithm for solving the four-color problem. Since the original algorithm is limited to a minimal number of colors, we modified it to use all available colors. The modified version is shown in Algorithm 2. The difference is that we added sorting procedure (line 8), just before deciding colors for each node. In our proposed algorithm we prefer to set a color assuming a long distance between the same color. Such operation could distribute colors in the network efficiently.

The origin server gathers access logs from cache servers periodically and colorizes the contents to define their popularity classes. Table II shows the different tags used to classify the popularity classes and the number of 1 bits. The number of contents in each popularity class is preliminarily defined according to the sub-optimal result of GA. The origin server first sorts the contents by their latest popularities, and then colorizes the contents from the most popular one. Contents with the same popularity class have tags in a cyclic fashion. In addition, new contents that are introduced between periodic updates, are initially tagged as low-popularity. However, they can still be cached in the LRU area to prevent the hit ratio's dropping.

In our scheme, cache servers have two separated areas that are managed by LFU and a modified version of LRU. For the LRU area, we adopt the modified LRU [4] which is an extension of the pure LRU that inserts a new object

Algorithm 2 Server colorization algorithm

```
Require: available colors C = {c_1...c_i}, Graph G(V, E) where nodes
    V = {v_1...v_j} and edges E = {e_1...e_k}
1: Initialization: sort V based on degree(v_j) descendingly
2: for v in V do
3:     adjacent_colors ← ∅
4:     for a in adjacent_nodes(v) do
5:         adjacent_colors = adjacent_colors ∪ {color(a)}
6:     end for
7:     candidate_colors ← C − adjacent_colors
8:     sort candidate_colors based on the minimal distance to the same
    color descendingly
9:     color(v) = candidate_colors[0]
10: end for
```

at a few rank up from the LRU position. Such behavior can decrease the probability of eviction of popular contents.

As a support to the proposed architecture, we also implemented a set of routing algorithms for the ring, 2D-mesh, and other mesh topologies. In the ring topology, the request is routed following a unidirectional scheme. In the 2D-mesh, we opted for the X-Y dimension order routing. For the other mesh topologies, the routing cost from the origin server is computed with the well-known Dijkstra algorithm [14], and the requests are routed with the shortest paths. In all topologies, first, cache servers search an adjacent server, with the same color as the requested contents, and causing the lowest cost. Second, when the adjacent server is found, and to increase hit ratio, the request is forwarded to the server rather than using the shortest path. Once the request reached the origin server or a cached server, the requested content is returned with backtracking the requesting path.

V. Evaluation

A. Evaluation Methodology

Our evaluations are performed using the unidirectional ring, 2D-mesh, and a mesh network from the NTT backbone network in Japan [12]. Figures 6, 7 and 8 show the adopted ring, 2D-mesh, and the NTT topologies and their colorations, respectively. For the ring and 2D-mesh topologies, shown

Table II
Popularity class and corresponding tags with four colors.

	Color bit				
Popularity class	R	G	B	Y	bit count
high	1	1	1	1	4
mid-high	1	1	1	0	3
	1	1	0	1	3
	1	0	1	1	3
	0	1	1	1	3
middle	1	1	0	0	2
	1	0	1	0	2
	1	0	0	1	2
	0	1	1	0	2
	0	1	0	1	2
	0	0	1	1	2
mid-low	1	0	0	0	1
	0	1	0	0	1
	0	0	1	0	1
	0	0	0	1	1
low	0	0	0	0	0

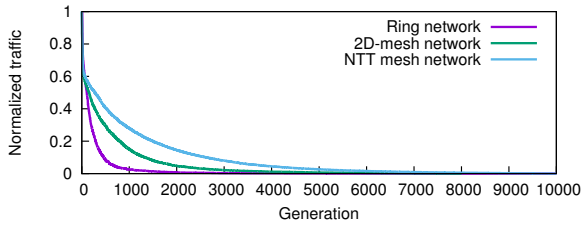

Figure 4. Comparison of convergence.

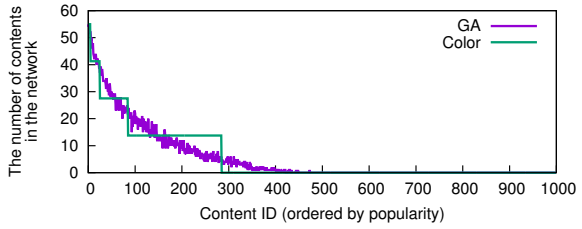

Figure 5. The number of contents in the NTT network calculated by the GA and the number of contents to be cached in the evaluation.

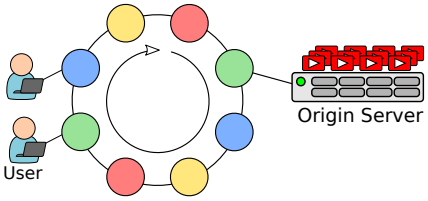

Figure 6. Unidirectional ring topology with 8 nodes with their corresponding colorations.

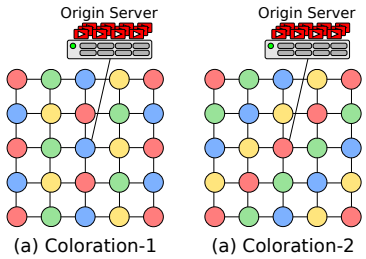

Figure 7. 2D-mesh topology with two different colorations.

in Figs. 6 and 7, respectively. We manually applied colors to the nodes since their topologies are quite simple, with two different colorations for the 2D-mesh topology. For the NTT network shown in Fig. 8, the nodes have been colorized using our modified version of Welsh-Powell algorithm [13], as shown in Algorithm 2. Moreover, each node in all the three networks has a client that generates content accesses.

The popularities of the contents are defined with the Gamma distribution, which generates more realistic content accesses than other distributions [11] (e.g., Zipf and Weibull distribution). The number of total contents, cache capacity, and gamma parameters are shown in Table I.

The number of contents in each popularity class is obtained from the result of the sub-optimal allocations computed by the GA using a single host with the configuration shown in Table IV. Fig. 4 shows the transitions of the normalized traffic generated by content accesses. Popularity classes are set just after the convergence, which occurs at the generation 3000, 8000, and 10000 for the ring, 2D-mesh, and NTT topology, respectively.

B. Content Popularity

Fig. 5 shows the sub-optimal curve and the number of cache servers for each content. Table III shows the number of contents in each popularity class in the evaluations. For example, the most popular five contents are cached on all servers in the NTT topology. The second most popular 20 contents are cached in 75% of all servers, since they have tags with three 1-bits out of 4-bit tags. It is important to mention that the number of contents in the popularity classes is adjusted to not overflow the cache capacities, and to cache the tagged contents efficiently. We notice that the sub-optimal curve of the proposed color scheme tends to follow the GA-induced curve in a phased fashion.

C. Traffic Reduction

We compared the traffic reduction in our proposed coloration-based strategy to a *no-cache*, Perfect-LFU, and GA strategies. The *no-cache* is simply a network without any cache servers only for the performance comparison. In the Perfect-LFU scheme [15], an optimal hit ratio could be achieved under a static popularity, and where only the most popular contents are stored. All servers have the same contents in the evaluation, because content popularity is fixed in the evaluation. In our coloration strategy, the tags of contents (i.e., colors) are decided by the origin server using a preliminary calculation. In fact, the GA preliminarily calculates the sub-optimal content allocations in the network and the selected contents are stored into each server persistently. Figs. 9, 10, and 11 show the traffic reduction for the ring, 2D-mesh, and the NTT topologies, respectively. The adopted generation of the GA is shown in the parenthesis, and are obtained after the convergence according to Fig. 4.

From these three figures, we can see that the traffic reduction is more noticeable when the network gets larger since the cache servers can reduce more hop counts from the

Table III
THE NUMBER OF CONTENTS IN EACH POPULARITY CLASS

	High	Mid-High	Middle	Mid-Low	Low
Ring	10	10	80	170	730
2D-mesh	25	25	40	145	765
NTT	5	20	60	200	715

Table IV
CONFIGURATIONS OF THE COMPUTING HOST FOR GA

CPU	Intel Core i7-3930K @3.80GHz
CPU Core	6 physical cores / 12 logical cores
RAM	64GB

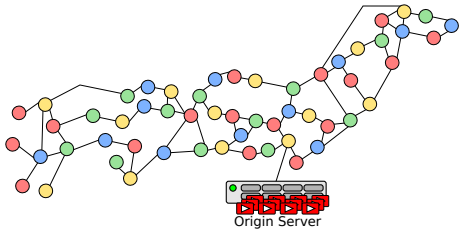

Figure 8. NTT mesh topology in Japan[12] and its coloration.

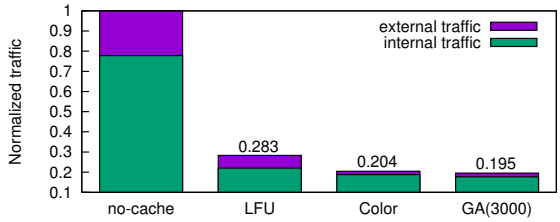

Figure 9. Normalized traffic on a ring-based network with 8 nodes.

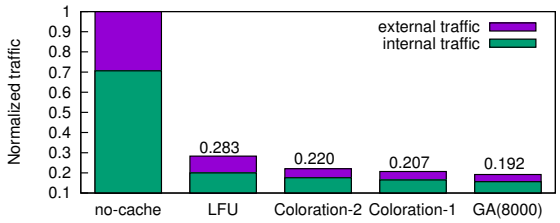

Figure 10. Normalized traffic on a 2D-mesh network with 25 nodes.

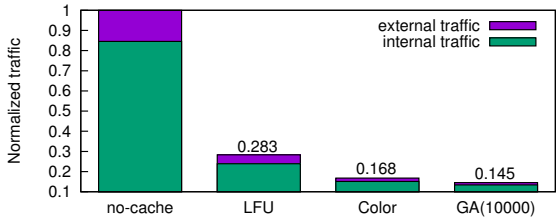

Figure 11. Normalized traffic on the NTT network with 55 nodes.

clients to the origin in large networks. In the 2D-mesh topology, the coloration-1 achieved slightly better reduction than coloration-2 since the probability of intermediate servers with different colors increases. This result indicates that an efficient server colorization scheme could further reduce the traffic.

In all topologies, our scheme with color tags achieves better traffic reduction, which is close to the sub-optimal result calculated with GA. In fact, when compared to the GA strategy, the differences in hit ratios are lower than 0.9%, 1.5%, and 2.3% in the ring, 2D-mesh, and NTT topology, respectively. From these result, it is clear that the density of content allocation plays an important role in the traffic reduction.

D. Hit Ratio Evaluation after Popularity Change

In order to evaluate our proposed hybrid caching strategy, and to see the effect of a rapid variation in the content popularity, we compared the hybrid cache with a cache that has only colored area. The hybrid one splits the cache area capacity to 90% for the contents with the colored area and 10% for the modified LRU area. The color-only cache has only a colored area that works without the use of the LRU area according to the Algorithm 1, but without the lines 3–5 and 12–16 responsible of the LRU area handling. To simulate the behavior of an update activity, we insert new five contents with the highest popularity, just before the request ID 500k. The obtained result is shown in Fig. 12.

The hybrid caching strategy maintains the hit ratio even when new contents are inserted, while the color-only cache drops its hit ratio. In contrast to the color-only cache where the hit ratio drops by 13.9%, the proposed hybrid strategy could limit the degradation to only 2.3%. Also, the hybrid one achieved better traffic reduction before the insertions since the modified LRU area is able to cache the mid-high contents that do not match the server color.

E. GA Computation Overhead

Table V shows the GA's computational time for different generations and for the three topologies. The computational time increases as the number of servers and links increases. It indicates that it takes a long calculation time to achieve sub-optimal allocations for large networks. For example, it takes more than 7 hours to calculate sub-optimal allocations for the NTT topology. However, the long calculation is rarely required once the popularity classes are defined. This is due to the fact that the VoD accesses always follow almost the same gamma distribution [11], and to the low variability of the backbone network's topology. Thus, the overhead of GA could be amortized. On the other hand, the colorization of contents takes only a few seconds since it requires just sorting with O($n$log$n$) and colorization with O(n) where n indicates the number of contents.

VI. Discussion

In our proposal, we used only four colors to colorize cache servers and contents. However, the use of more colors such as 8 or 16 would further highlight the merit of our proposal and enhance the traffic reduction. In fact, a large number of colors could help to fit the sub-optimal curve calculated by GA.

Moreover, if the popularities of the newly inserted contents could be known or predicted, the hit rates will be

Table V
Computational time for different generations.

Topology	Nodes	Generation			
		1000	3000	8000	10000
Ring	8	5m33s	16m01s	42m05s	52m33s
2D-mesh	25	34m44s	103m11s	274m40s	343m17s
NTT	55	42m08s	127m34s	350m38s	440m25s

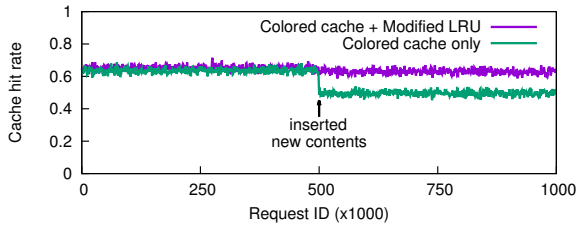

Figure 12. Cache hit ratio with a change in the popularity.

improved by adding a special tag for consistent caching in LRU area. Such operation does not highly degrade the hit ratio even if the prediction failed because the origin server can update content colors much frequently than calculating optimal allocations. Also, small LRU area is still considered enough to follow the rapid popularity changes. In addition, since people live in different regions, and may have different interests, it would be very interesting if the content popularity has geographical localities. Thus, the hit ratio can be improved by gathering access logs and colorizing contents for each region.

VII. Conclusion and Future Work

This paper proposed a light-weight scheme of cooperative caching that distributes contents in sub-optimal locations in the network. Our scheme adds color tags to both cache servers and contents. Each server caches contents that have the same color tag. Evaluation results demonstrated that our scheme could achieve a considerable improvement, close to the sub-optimal reduction of the traffic, with only a few seconds of computation overhead. Moreover, hybrid caching strategy with the colored cache and the modified LRU allows us to follow the rapid variation in content popularity. As future work, we plan to extend to proposed hybrid LFU-LRU hybrid scheme by using different combinations of ratios and cache capacities, with the aim to have more comprehensive evaluation of the traffic reduction, especially when using more realistic access pattern. It is also interesting to find more efficient ways to colorize cache servers, which will further enhance the traffic reduction by considering color distribution endorsed with efficient routing algorithms.

Acknowledgement

This work is partly supported by TIS Inc. under collaborative research project for reducing internet traffic by efficient utilization of distributed cache servers.

References

[1] "The Zettabyte Era—Trends and Analysis," White Paper, Cisco Systems, Inc., Jun. 2016.

[2] B. M. Maggs and R. K. Sitaraman, "Algorithmic Nuggets in Content Delivery," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 3, pp. 52–66, Jul. 2015.

[3] Z. Li and G. Simon, "In a Telco-CDN, Pushing Content Makes Sense," *IEEE Transactions on Network and Service Management*, vol. 10, no. 3, pp. 300–311, Sep. 2013.

[4] D. D. Vleeschauwer and D. C. Robinson, "Optimum Caching Strategies for a Telco CDN," *Bell Labs Technical Journal*, vol. 16, no. 2, pp. 115–132, Sep. 2011.

[5] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min, "Inside the Bird's Nest: Measurements of Large-scale Live VoD from the 2008 Olympics," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, 2009, pp. 442–455.

[6] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, "Understanding User Behavior in Large-scale Video-on-demand Systems," in *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, 2006, pp. 333–344.

[7] N. Choi, K. Guan, D. C. Kilper, and G. Atkinson, "In-Network Caching Effect on Optimal Energy Consumption in Content-Centric Networking," in *Proceedings of 2012 IEEE International Conference on Communications*, Jun. 2012, pp. 2889–2894.

[8] Z. Wang, H. Jiang, Y. Sun, J. Li, J. Liu, and E. Dutkiewicz, "A k-coordinated Decentralized Replica Placement Algorithm for the Ring-Based CDN-P2P Architecture," in *Proceedings of 2010 IEEE Symposium on Computers and Communications*, Jun. 2010, pp. 811–816.

[9] P. Amani, S. Bastani, and B. Landfeldt, "Towards Optimal Content Replication and Request Routing in Content Delivery Networks," in *Proceedings of 2015 IEEE International Conference on Communications*, Jun. 2015, pp. 5733–5739.

[10] Y. Zhou, L. Chen, C. Yang, and D. M. Chiu, "Video Popularity Dynamics and Its Implication for Replication," *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1273–1285, Aug. 2015.

[11] X. Cheng, J. Liu, and C. Dale, "Understanding the Characteristics of Internet Short Video Sharing: A YouTube-Based Measurement Study," *IEEE Transactions on Multimedia*, vol. 15, no. 5, pp. 1184–1194, Aug. 2013.

[12] A. Arteta, B. BarÃąn, and D. Pinto, "Routing and Wavelength Assignment over WDM Optical Networks: A Comparison Between MOACOs and Classical Approaches," in *Proceedings of the 4th International IFIP/ACM Latin American Conference on Networking*, 2007, pp. 53–63.

[13] D. J. A. Welsh and M. B. Powell, "An upper bound for the chromatic number of a graph and its application to timetabling problems," *The Computer Journal*, vol. 10, no. 1, pp. 85–86, Jan. 1967.

[14] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[15] G. Einziger and R. Friedman, "TinyLFU: A Highly Efficient Cache Admission Policy," in *Proceedings of 2014 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, Feb. 2014, pp. 146–153.