Can Recommenders Compensate for Low QoS?

Mateus Nogueira, Carlos Bravo, Daniel Menasché
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil
{mateussnogs,sadoc}@dcc.ufrj.br

Thrasyvoulos Spyropoulos
Institut Eurecom
Sophia Antipolis, France
spyropou@eurecom.fr

Pavlos Sermpezis
Aristotle University of Thessaloniki
Greece
sermpezis.pavlos@gmail.com

***Abstract*—Content recommendation systems, also known as recommenders, are pervasive and impact a significant portion of users demands over the Internet. Although recommenders have been primarily devised to account for users interests with respect to the content catalog, mobile users are typically served by a network that is unreliable and subject to losses and low QoS. Can content recommenders compensate for low QoS? To answer this question, we conducted experiments over the Internet, and report our findings on *(i)* the characterization of QoS and *(ii)* the compensation for low QoS. Our measurements suggest that content that is far from the trends tends to be far from the user. We quantify the extent at which unpopular content tends to be served with lower QoS and establish a methodology to determine the relationship between contents' popularity and its physical proximity to the users. Then, we verify that making requests a bit trendier can hit much closer content. In particular, our results suggest conditions under which a recommender can compensate for low QoS, at zero costs for operators.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Content recommenders and caches are two fundamental pillars in the Internet ecosystem. While recommenders such as those used by Netflix influence a significant portion of users demands, caches such as deployed by Akamai serve a vast amount of content to end users. Caches reduce the load on custodians, decrease the latency for users, and benefit network infrastructure reducing the traffic over bottlenecks.

Given the benefits of caching, a significant effort has been invested in order to improve cache performance. In particular, similarity caching [1], [2] and cost-aware caching, including QoS-aware [3], [4] and utility-driven caching [5], are some of the various recent developments in that domain [6]–[8]. Such advances, in turn, suggest novel opportunities but also pose new challenges in the realm of content distribution.

The basic idea behind similarity caching and cost-aware caching consists in determining both the similarity between contents and the cost to serve and/or retrieve a content, and then make decisions about which content to store and/or serve based on such assessments. Clearly, the multiple dimensions involved in the problem are intertwined, and optimal decisions are non trivial. In particular, a user consuming a content not stored in a local cache may experience low quality of service (QoS), and may prefer to rely on a content recommender to find a title that suits its expectations both in terms of content and QoS, motivating our research question: *can a recommender compensate for low QoS?*

In this paper, we report results on Internet measurements indicating the feasibility of characterizing the QoS at which different items can be served. Given such characterization, and information about the content recommendation graph, we present findings to support conditions under which we have an affirmative answer to our main question.

Our methodology involves both network and recommender measurements. Network measurements are used to collect information about the delays incurred to consume different videos. We refer to the approach used to assess those delays as content-aware pings and traceroutes, as it involves first identifying the host which will serve the desired content, and then issuing a `ping` and a `traceroute` towards that host to assess the corresponding delay. Then, we combine information collected from such measurements together with measurements of the recommendation graph. By sampling the YouTube recommendation graph we learn which contents are close to each other. Then, we bridge the gap between *closeness* with respect to the nature of the content (*content distance*) and with respect to host proximity (*network distance*) to draw our main findings. We summarize our key contributions as follows.

Characterization: far from the trends, far from the user. We quantify the extent at which unpopular content tends to be served with lower QoS. In particular, we establish a methodology to determine the relationship between contents' popularity and its physical proximity to the users, combining sampling of the recommendation graph and traceroutes in the physical network. The proposed method allows us to determine how popular a content has to be, to be closer to the user, and it is instrumental for tuning recommenders.

Compensation: a bit trendier, much closer. Favoring slightly trendier content while issuing recommendations (i.e., allowing a *content distance* between the requested content and the served content) can significantly increase the proximity of contents to users (i.e., decreasing *network distance*), positively impacting QoS. In particular, our results suggest conditions under which a recommender can compensate for low QoS, at zero costs for the network admin.

Next, we introduce our measurement methodology. Then, Sections III and IV report a characterization of delays towards contents and the extent at which recommenders can compensate for those delays, respectively. We present related work in Section V, and Section VI concludes.

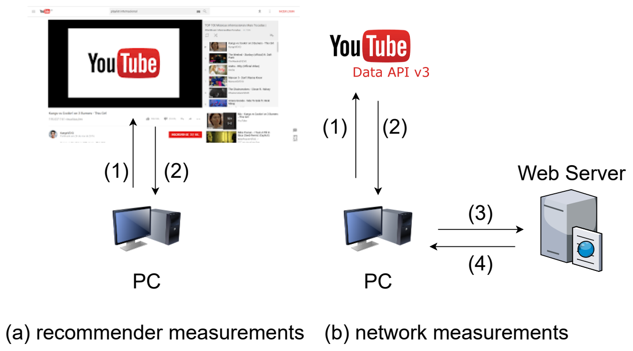


Fig. 1. Measurement setup: (a) recommender measurements to retrieve recommendation graph; (b) network measurements capture delays towards contents.

TABLE I
DATASETS DESCRIPTORS

Country	Videos (samples)	CHR	Unique URLs
BR	872	38%	167
CA	969	44%	211
FR	3360	71%	202
HK	1758	86%	203
IN	617	78%	162
US	1859	69%	281

II. Methodology

Next, we introduce our measurement methodology.

Goals. Popular and trending videos are known to be cached close to users. In YouTube, in particular, a list of trending contents is presented to users at their home page. Beyond such remarkably trending contents, which other contents are cached closer to users? How do different features, such as number of views, impact closeness? And how do users profiles, e.g., reflected through their vantage points, impact delays towards different contents?

Overview. We use YouTube API to access the recommendation system and generate recommendation graphs for each trending content by performing a Breadth-First Search (BFS) through the network of recommendations [9]. Then, we emulate a request towards each of the videos, and determine the corresponding media server urls.

Let $\mathcal{T}$ be the set of trending videos considered as our seeds for the BFS. We let $|\mathcal{T}|$ = 50, i.e., we consider the top-50 most popular videos in each of the considered regions, and start a BFS from each of those. The BFSs are executed up to a depth of five hops in the recommendation graph, and accounting for the first three videos in each of the observed recommendation lists.

Is a video cached close to requester? We measured network level features, using ping and traceroute towards video servers, and group the videos into two clusters based on those metrics. Let C denote an indicator variable, equal to 1 if our measurements suggest that the video is cached close to our measurement vantage point, and 0 otherwise. Then, we computed correlations between C and metrics related to the recommenders, that are introduced in the sequel.

Datasets. A summary of our dataset descriptors is presented in Table I. Our dataset collected from vantage points in Rio de Janeiro, for instance, has 872 video samples and 167 unique hosts serving those videos, out of which 38% are marked as low delay hosts. Content served by low delay hosts is assumed to be cached close to users, i.e., $C = 1$.

A. *Content distances by recommenders*

We represent YouTube videos and their relations through a recommendation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Thus, each video is represented by a node, and each recommendation is represented by an edge, where $(i, j) \in \mathcal{E}$ if j appears in the recommendation list of i.

We consider an abstraction of a surfer that navigates across the nodes of the recommendation graph. The surfer begins at the seed of the graph. The shortest path between each video and the seed is referred to as the video *depth*. The position of the video in its corresponding recommendation list is referred to as the video *width*. Note that whereas the depth is oblivious to the ordering at which videos appear in the recommendation lists, width is sensitive to such ordering.

Recommender measurements. Next, consider a surfer sample path. Each time the surfer visits a node, it produces a sample. The sample comprises the identifier of the visited video and its view count, together with its width and depth. Such measures are collected in a passive manner, without interfering with the network state (Figure 1(a)). Then, we proceed with an active measurement of delays.

B. *Host distances in the network*

We have developed a software module to identify YouTube media servers associated to each video in the recommendation graph. Our module is similar to [10] wireshark webdeveloptools. It is important to note that the YouTube hosts returned by our program are dependent on the network used when collecting data. In this work, we focus on data collected in Rio de Janeiro in an ISP's network. After identifying hosts associated to each sample, we perform traceroutes and record the corresponding delays and path lengths (Figure 1(b)).

Feature summary. A sample is a tuple containing (a) video identifier; (b) view counts; (c) width; (d) depth; (e) host; (f) delay observations; (g) path length. Features are summarized in Table II.

III. Experimental findings

Next, we report our experimental findings. Our experimental goal is to characterize the relationship between the recommendation graph and QoS metrics.

A. *Bridging recommender and network*

Next, we report correlations between recommender and network metrics, bridging the collected measurements.

Let V and A denote the video view counts and age, and let N be an indicator variable, equal to 1 if the video is in the native language of the region wherein our vantage point was located, and 0 otherwise. We observed correlations

TABLE II
DESCRIPTION OF COLLECTED FEATURES PER SAMPLE

feature	description
Recommender and recommendation graph features	
I, video identifier	YouTube video unique identifier
V, view counts	number of views, since video creation
W, width	position of video recommendation list
D, depth	number of hops to seed of recommendation graph
QoS and network related features	
H, host	host serving video I
L, path length	length of path (number of hops) to H
P, ping values	vector of 10 delay observations collected with ping
$\mathbb{E}(P)$, mean delay	average of ping values
C, cached close?	equals 1 if $\mathbb{E}(P)$ is "small", 0 otherwise

between C and the above three variables of 0.18, 0.12 and 0.13, respectively. Such correlations indicate the extent at which more popular content is cached closer to our vantage points. In particular, the correlation between video language and caching suggests that the recommendation of videos in native languages may favor not only users interests but also higher QoS.

Let W and D be the video width and depth, respectively. We observed negative correlations of -0.19 and -0.23 between C and the above two metrics, respectively. Such correlations quantify the tendency that videos closer to the root of the recommender graph are closer in the cache network.

B. A few hosts serve most of the content

We found that different video links have significant overlap in their URLs, referring to machines stored in the same datacenter. To illustrate that point, we consider the hosts observed from our vantage points in Rio Janeiro. We observed four common prefixes when issuing requests to top-trending Brazilian contents from a vantage point in Rio de Janeiro: *b8u*, *8p8* and *bg0*. A fourth prefix, *q4f*, appeared in less than 5% of our requests, and was associated to high delays, in the order of 150ms (six times higher compared to the others).

- *b8u* and *8p8* correspond to low delay values, as observed from pings issued from Rio de Janeiro. Delays are around 21ms and the corresponding standard deviations equal 0.28ms and 0.42ms;
- *bg0* and *q4f* correspond to larger delays, around 28ms and 150ms respectively. The corresponding standard deviations equal 11ms and 0.45ms.

In what follows, we refer to the aforementioned prefixes simply as *hosts*, noting that they may subsume multiple machines in the same domain. As hosts are distinguished into two groups based on delays, we refer to those hosts as high delay hosts and low delay hosts depending on the group they fit. The corresponding videos served by those hosts are marked as videos served by caches closer and farther away from users, with $C = 1$ and $C = 0$, respectively.

C. A birds eye view across countries

Next, we present a birds eye view of our measurements. We have run measurement campaigns across six countries. Each country corresponds to one vantage point, and to its own set of trendy contents.

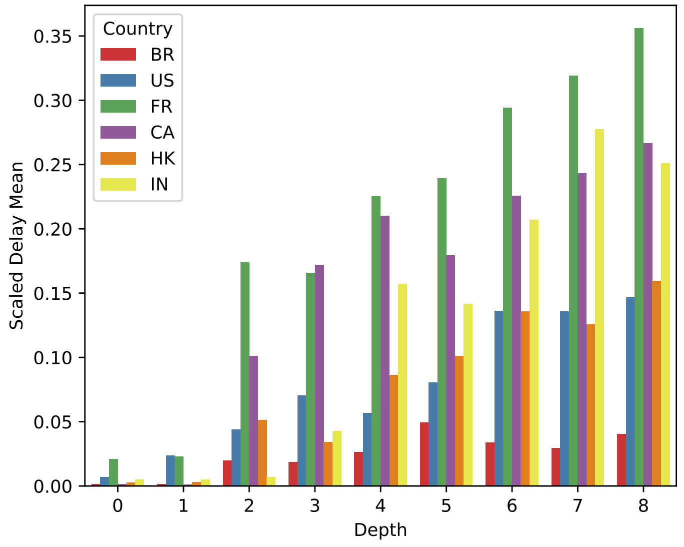


Fig. 2. Scaled delays versus recommender depth

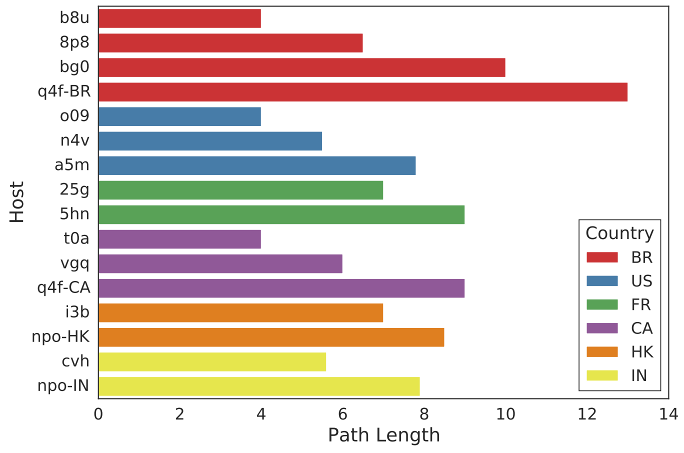


Fig. 3. Path lengths towards different hosts

Figure 2 shows the relationship between distance in the recommender graph against standardized delays, obtained using the `MinMaxScaler` [11] so that standardized delay values fall in the range between 0 and 1. In all the considered countries, moving deeper in the recommender graph corresponds to higher delays. One of our aims in the remainder of the paper is to further investigate and quantify this relationship, discussing its implications for the design of recommenders.

Figure 3, in turn, shows the path lengths for hosts serving contents in the six considered countries. It shows that hosts are always up to thirteen hops from our vantage points, and that the difference between the closest to furthest host is of at least two hops.

Due to space limitations, in what follows we focus our detailed analysis on the Brazilian measurements, noting that all our results extend to the six considered countries.

TABLE III
MEDIAN PING VALUES (TIMES IN MILLISECONDS)

Origin city (SpeedChecker vantage point)	Target host with top-trending contents			
	Brazilian contents		French contents	
	bg0 Sao Paulo	b8u Rio de Janeiro	fr1 5hn 4g5 Chartres I	fr2 25g Chartres II
Sao Paulo	17	21	223	223
Rio de Janeiro	21	16	219	219
Athens	295	302	94	94
New York	333	331	334	403
Chartres	223	219	17	8

D. *Accounting for multiple vantage points*

Next, our goal is to further quantify the extent at which unpopular content is impacted by slower-to-respond hosts. To that aim, we consider top trending Brazilian and French videos. Then, we ping hosts storing those contents from different vantage points, leveraging the measurement infrastructure provided by SpeedChecker. We collect at least 10 measurements from each host, and report median values.

Table III illustrates our results. To produce Table III, we consider the Brazlian and French top-trending videos and ping their closest hosts, from vantage points in Brazil, Greece, USA and France. We group samples according to whether they are associated with slower-to-respond or fast-to-respond hosts, using the methodology described in the previous section.

Clearly, the lowest delays for the top trending contents are obtained in the corresponding regions. As the difference in delays can be of an order of magnitude, those results illustrate that recommenders can have a potentially significant impact on the experienced QoS.

IV. CAN RECOMMENDERS COMPENSATE FOR LOW QOS?

Next, we leverage the QoS characterization introduced in the previous section to establish conditions under which content recommennders can compensate for low QoS. First, we indicate that a decision tree can capture delays based on data collected from the recommendation graph. Then, we illustrate a simple mechanism for recommenders to leverage knowledge about delays to compensate for low QoS.

A. *From content recommendation graphs to network metrics*

In this section we aim at answering the following question: is it feasible to assess delays (QoS) solely based on recommeder features? An affirmative answer to such question can significantly simplify the design of QoS-aware recommenders, as in this case the distance of contents in the recommender graph can be taken as a proxy to QoS metrics.

Figure 5 shows a decision tree (DT) that serves to illustrate the feasibility of classifying contents as a function of the corresponding QoS at which they are served, solely based on their view counts, published date and on their position in the recommendation graph. The training and test sets were setup in a way such that the two target classes ($C = 1$ and $C = 0$) are balanced, i.e., 50% of the samples correspond to each class.

Each node of the tree corresponds to a decision. The root node corresponds to the decision which entails largest discriminatory power, and consists of classifying videos based on their depth in the recommender graph. If depth is lower than 4, a fraction of 70% of the videos are served with high QoS (low ping values, $C = 0$). Alternatively, if the view count is larger than 2,42 million views, the content is classified as served with high QoS (low ping values and $C = 0$). Finally, if the age of the video is larger than one month, it is assumed not to be cached (flash crowds towards recent content are not captured in this simple DT).

Despite its simplicity, the presented DT already corresponds to an accuracy, precision and recall of 0.72, 0.63 and 0.88, respectively.

B. *From network metrics to novel content recommendations*

Next, we consider the extent at which QoS-aware recommenders can compensate for low QoS. Figure 4(a) shows the relationships between recommender features (width, depth and view counts) and QoS (ping values). Each point corresponds to a content. The orange (resp., blue) points correspond to contents served by high ping (resp., low ping) hosts. Note that "close" to an orange point we typically have multiple blue points, corresponding to low ping values. Points which are "close" to each other in that figure are near each other in the recommendation graph, suggesting that the recommender can compensate for low QoS.

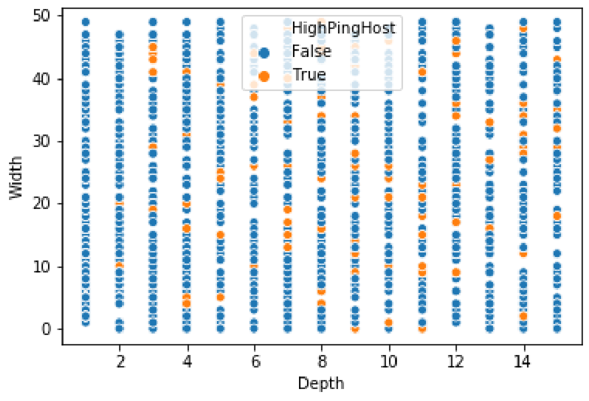
To further illustrate the feasibility of determining which contents can be served with high QoS, Figure 4(b) shows how view counts and depth impact QoS. Figure 4(b) shows a clear phase transition when depth grows beyond 2, suggesting that contents in levels 1 and 2 can be safely assumed to have low ping values. A recommender can leverage that phase transition to tune the order of suggested contents, compensating for low QoS.

C. *Recommenders against low QoS*

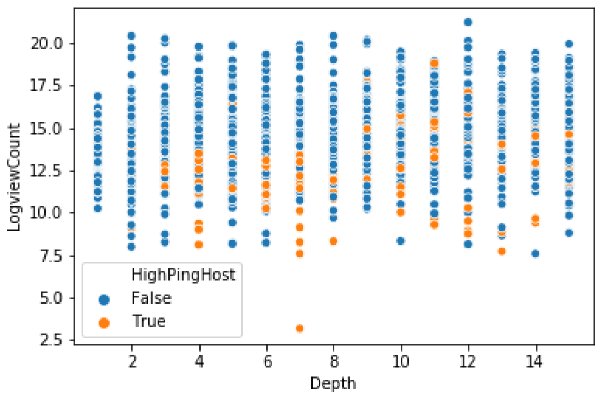
To answer our main research question, we conduct experiments simulating scenarios in which a user randomly starts selecting one of the most trending videos, followed by videos from subsequent recommendation lists. These lists are presented in 2 ways: *(i)* the original order, i.e., like they would be in YouTube and *(ii)* according to an algorithm that prioritizes cached videos, the Cache-Aware & BFS-related Recommendations (CABaRet) [9]. CABaRet recommendations replace some non-cached videos by cached counterparts, and order the videos in a way that cached videos are preferably presented at the top of the recommendation lists.

To determine whether a video is cached or not, we leverage our measurements as described in the previous section. In particular, we use the inferred indicator C to determine whether a video is cached, and to eventually increment hit counts. While the authors of the CABaRet algorithm assumed, shrewdly, that the top 50 trending videos were cached, our measurements allow the application of CABaRet algorithm with more information regarding the network conditions of the media servers.

Thus, we compare the cache-hit ratio (CHR) produced by YouTube's lists of recommendations against the lists generated



(a)



(b)

Fig. 4. Relationships between recommender features (width, depth and view counts) and QoS (ping values).

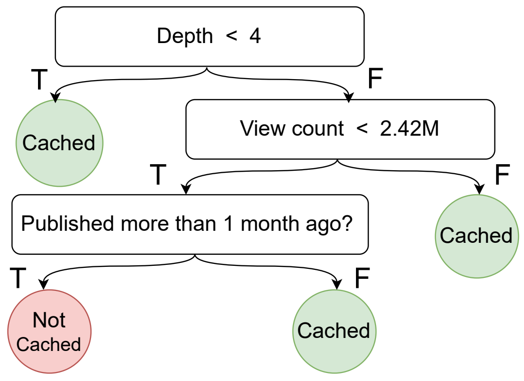


Fig. 5. Decision tree

by CABaRet. To that aim, we vary the mechanism through which users select a video from a recommendation list, considering two alternatives: uniform and Zipf. The uniform distribution assumes users select videos uniformly at random, whereas the Zipf distribution captures a preference towards videos ranked in top positions [12]. We also vary the two main CABaRet parameters: maximum depth ($\widetilde{D}$) and maximum width ($\widetilde{W}$). Larger values correspond to broader searches for cached contents in the recommendation graph, providing more flexibility for recommenders to compensate for QoS. In particular, when $\widetilde{D} = 1$ CABaRet only reorders the recommendations, whereas for $\widetilde{D} = 2$ it also replaces some non-cached videos by cached alternatives.

Figure 6 shows that CABaRet easily achieves a higher CHR than Youtube baseline. When the request workload is uniform, CABaRet requires larger $\widetilde{D}$ and $\widetilde{W}$ to show its benefits. This is because both replacements and reorderings of recommendations affect CHR under the Zipf workload, whereas the uniform workload is insensitive to reorderings.

Summary. Combining recommender and network measurements, we learned that recommendation reorderings are sufficient to increase CHR from 0.64 to 0.89 under a Zipf workload. Diminishing returns are gained by allowing for replacements of recommendations, in addition to reorderings. In particular, allowing for replacements of videos that are at most two hops away in the recommendation graph suffices to reach a CHR of 0.98.

V. Related work

There is a large body of work on recommenders and QoS, and on their interplay [9], [13]. However, to the best of our knowledge there has been no measurements to assess the extent at which recommenders can compensate for low QoS.

Content recommenders are sensitive to the user context [14], [15]. The user context is typically understood as the day of the week, the place where the user is consuming its contents, or even the device. In this paper, we considered a novel dimension of context, namely, QoS. We have shown that it is feasible to characterize and quantify QoS per content, and that a recommender can benefit from such characterization. In particular, our work is complementary to the state of the art in recommendation systems [16], as our insights can be coupled to existing recommenders [17]–[21].

It is well known that proxy caches are a cheap and effective solution to counter bottlenecks in networked systems [22]–[24]. In particular, they are the first choice before considering a costly upgrade of the network infrastructure [25]. In this paper, we have experimentally shown that content recommenders can catalyze the benefits of caching, further increasing its potential at virtually zero costs. In [10], [26] the authors investigate statistical properties of the paths towards contents in Youtube and Netflix. In this paper, we build on such measurements, and consider how a content recommender can leverage those to benefit the users.

There has been a recent surge in interest in the relationship between cache networks and recommendation systems [9], [27], [28]. Optimal allocations in cache networks typically require knowing the cost-to-go, i.e., the cost incurred by a miss up until finding the content at the closest cache [29]. The methodology considered in this paper to characterize

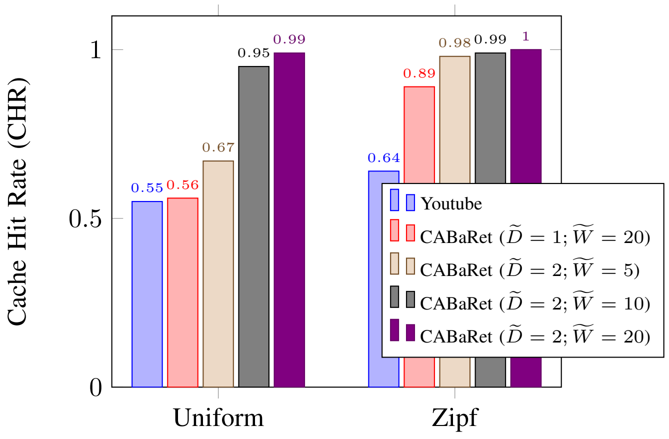


Fig. 6. CHR varying workloads and recommenders

QoS at content level can be used for those purposes, as well as to parametrize content recommenders. The interplay between QoS and recommenders involves both human-related and network-related elements. CABaRet [9] comprises a conceptual framework to capture those factors in an integrated fashion. In this paper, we report proof-of-concept network measurements that are complementary to CABaRet.

VI. Conclusion

Content recommenders play a fundamental role in the way content is consumed over the Internet. Nonetheless, they are typically devised without accounting for one of the most critical aspects of the Internet infrastructure, namely, its best effort nature. In this paper, we embrace the delays incurred by content requests in the Internet environment and characterize QoS-related metrics in a per-content basis. While doing so, we observed that the recommenders have significant flexibility with respect to the available contents to be offered and their corresponding QoS levels. We found that contents that are found at depth one or two are cached closed to users with very high probability, suggesting a simple heuristic to favor QoS while satisfying users interests with an increased overall QoE. In future work, we plan to integrate the insights collected in this work together with experiments with real subjects to assess their sensitivity to recommendations adjusted based on QoS-related features.

References

[1] M. Garetto, E. Leonardi, and G. Neglia, "Similarity caching: Theory and algorithms," in *Proc. IEEE INFOCOM*, 2020.

[2] J. Zhou, O. Simeone, X. Zhang, and W. Wang, "Adaptive offline and online similarity-based caching," *IEEE Networking Letters*, vol. 2, no. 4, pp. 175–179, 2020.

[3] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Transactions on services computing*, vol. 4, no. 2, pp. 140–152, 2011.

[4] S. Kastanakis *et al.*, "Network-aware recommendations in the wild: Methodology, realistic evaluations, experiments," *IEEE Transactions on Mobile Computing*, 2020.

[5] M. Dehghan, L. Massoulie, D. Towsley *et al.*, "A utility optimization approach to network cache design," in *INFOCOM*, 2016, pp. 1–9.

[6] P. Sermpezis, T. Spyropoulos, L. Vigneri, and T. Giannakas, "Femto-caching with soft cache hits," in *GLOBECOM*, 2017, pp. 1–7.

[7] T. Giannakas, P. Sermpezis, and T. Spyropoulos, "Show me the cache: Optimizing cache-friendly recommendations for sequential content access," in *IEEE WoWMoM*, 2018.

[8] P. Sermpezis, T. Giannakas, T. Spyropoulos, and L. Vigneri, "Soft cache hits: Improving performance through recommendation and delivery of related content," *JSAC*, 2018.

[9] S. Kastanakis, P. Sermpezis, V. Kotronis, and X. Dimitropoulos, "Cabaret: Leveraging recommendation systems for mobile edge caching," in *ACM SIGCOMM workshops*, 2018.

[10] T. V. Doan, L. Pajevic, V. Bajpai, and J. Ott, "Tracing the path to YouTube: A quantification of path lengths and latencies toward content caches," *IEEE Communications Magazine*, vol. 57, no. 1, pp. 80–86, 2019.

[11] "Scikit learn," 2021, https://scikit-learn.org, sklearn.preprocessing.MinMaxScaler.

[12] D. K. Krishnappa, M. Zink, C. Griwodz, and P. Halvorsen, "Cache-centric video recommendation: an approach to improve the efficiency of youtube caches," *ACM TOMM*, vol. 11, no. 4, p. 48, 2015.

[13] P. Sermpezis *et al.*, "Towards QoS-aware recommendations," *arXiv preprint arXiv:1907.06392*, 2020, CARS/ACM RecSys.

[14] N. M. Villegas, C. Sánchez, J. Díaz-Cely, and G. Tamura, "Characterizing context-aware recommender systems: A systematic literature review," *Knowledge-Based Systems*, vol. 140, pp. 173–200, 2018.

[15] V. Bajpai, S. Ahsan, J. Schönwälder *et al.*, "Measuring YouTube content delivery over IPv6," *ACM SIGCOMM CCR*, vol. 47, no. 5, pp. 2–11, 2017.

[16] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for Youtube recommendations," in *RecSys*. ACM, 2016, pp. 191–198.

[17] T. Agagu and T. Tran, "Context-aware recommendation methods," *Intl. Journal of Intelligent Systems and Applications*, vol. 10, no. 9, p. 1, 2018.

[18] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2011, pp. 217–253.

[19] L. Baltrunas, B. Ludwig, and F. Ricci, "Matrix factorization techniques for context aware recommendation," in *Proc. ACM RecSys*, 2011.

[20] T. V. Nguyen, A. Karatzoglou, and L. Baltrunas, "Gaussian process factorization machines for context-aware recommendations," in *Proc. ACM SIGIR conference on Research & development in information retrieval*, 2014, pp. 63–72.

[21] R. Pagano, P. Cremonesi, M. Larson, B. Hidasi, D. Tikk, A. Karatzoglou, and M. Quadrana, "The contextual turn: From context-aware to context-driven recommender systems," in *Proc. ACM RecSys*, 2016, pp. 249–252.

[22] S. Traverso, K. Huguenin, I. Trestian, V. Erramilli, N. Laoutaris, and K. Papagiannaki, "Social-aware replication in geo-diverse online systems," *IEEE TPDS*, vol. 26, no. 2, pp. 584–593, 2014.

[23] ——, "Tailgate: handling long-tail content with a little help from friends," in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 151–160.

[24] B. M. Maggs and R. K. Sitaraman, "Algorithmic nuggets in content delivery," *ACM SIGCOMM CCR*, vol. 45, no. 3, pp. 52–66, 2015.

[25] J. F. Kurose, *Computer networking: A top-down approach featuring the internet, 3/E*. Pearson Education India, 2005.

[26] T. V. Doan *et al.*, "A longitudinal view of Netflix: Content delivery over ipv6 and content cache deployments," in *IEEE INFOCOM*, 2020.

[27] L. Chatzieleftheriou, G. Darzanos, M. Karaliopoulos, and I. Koutsopoulos, "Joint user association, content caching and recommendations in wireless edge networks," *PER*, vol. 46, no. 3, pp. 12–17, 2019.

[28] D. Zheng, Y. Chen, M. Yin, and B. Jiao, "Cooperative cache-aware recommendation system for multiple internet content providers," *IEEE Wireless Communications Letters*, 2020.

[29] S. Ioannidis and E. Yeh, "Adaptive caching networks with optimality guarantees," *ToN*, vol. 26, no. 2, pp. 737–750, 2018.