Cooperative Caching for Adaptive Bit Rate Streaming in Content Delivery Networks

Phuong Luu Vo
Department of Computer Science and Engineering, International University - VNUHCM, Vietnam
vtlphuong@hcmiu.edu.vn

Tuan-Anh Le
Faculty of Information Technology, University of Thudaumot, Vietnam
letuanh@tdmu.edu.vn

Choong Seon Hong
Department of Computer Engineering, Kyung Hee University, Korea
cshong@khu.ac.kr

Seung Il Moon
Department of Computer Engineering, Kyung Hee University, Korea
moons85@khu.ac.kr

Sungwon Lee
Department of Computer Engineering, Kyung Hee University, Korea
drsungwon@khu.ac.kr

Nga Ly Tu
Department of Computer Science and Engineering, International University - VNUHCM, Vietnam
ltnga@hcmiu.edu.vn

ABSTRACT

This work proposes a cooperative caching model which supports adaptive bit-rate streaming in content delivery networks. A linear program (LP) problem is applied to maximize the total user satisfaction. The optimal content placement and content fetching strategies are the solutions to the LP. The LP is a large-scale optimization problem because of the large amount of contents in the network. We also introduce a technique to decompose the large-scale LP into many subproblems which can be solved in parallel. Each subproblem is associated with one content.

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Network Protocols—*content caching, content delivery network*; G.1.6 [**Numerical Analysis**]: Optimization—*linear programming*

General Terms

Theory

Keywords

content caching, adaptive bit-rate streaming, content delivery network, linear programming

1. INTRODUCTION

In content delivery networks (CDNs), popular contents are cached in the cache nodes placed in the network. Thus, the users can obtain the contents at the intermediate nodes instead of downloading from the original sources. The performance of content delivery to the end users is improved. Ideally, all the contents should be cached in each storage node. However, there is a large number of contents in the network whereas the storage capacity is limited. Hence, intuitively, the storage nodes usually cache only the most popular contents. When a node receives a request for a particular content, if it already has the content at its storage, the content is directly sent back to the end user. Otherwise, the content is fetched from the other nodes in the network. It has been known that the cache nodes should cooperate in caching to reduce the amount of traffic transferred inside the network. Given the demands of the contents, the optimal content placement and fetching strategies are proposed by solving a large-scale integer linear program (LP) that minimizes the total cost of the network [4, 5, 10].

According to Cisco VNI report [8], videos will soon be a dominant traffic type in the Internet. By 2017, video traffic will be accounting for 69% of the internet traffic where nearly a million minutes of video will be transferred. On the other hand, adaptive bit-rate (ABR) streaming is a main video streaming technique in the Internet nowadays. It can utilize the network resource and provide a smooth playback and high quality-of-experience (QoE) [2, 9]. With ABR, each content is encoded into different representations corresponding to different playback rates. For example, reference [3] describes the recommended representation set of Apple's HTTP live streaming. Depending on network condition, an appropriate representation is dynamically chosen. Intuitively, a higher QoE video consumes more bandwidth and also requires a bigger representation size. Considering storage capacity of the nodes as the constraint, the target of this paper is to find an optimal content placement and content delivery strategies such as maximizing the network performance. The problem is modelled by a linear program (LP). Solving the LP yields the optimal content placement and fetching strategy. This LP is usually a large-scale problem due to the large number of contents in the network. We introduce a technique to decompose the LP to multiple subproblems which can be solved in parallel. Each subproblem corresponds to a content.

IMCOM '15, January 08-10, 2015, BALI, Indonesia.

http://dx.doi.org/10.1145/2701126.2701236 ...$15.00.

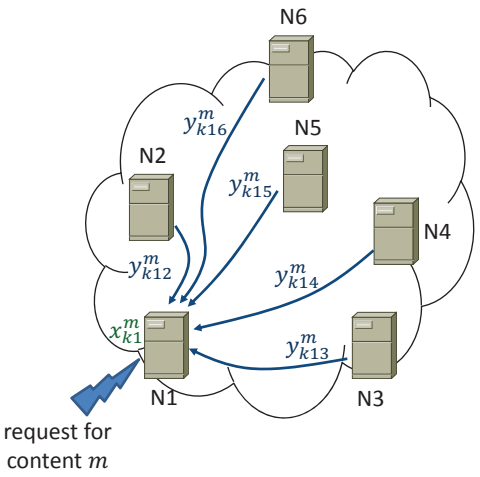

Figure 1: An example of cooperative caching in a content delivery network. To serve the request on content m at node 1 which has the x^m_{k1} fraction of content already, node 1 receives y^m_{k1i} fraction of content from node i.

To the best of our knowledge, there is still a lack of works in literature proposing a theoretical caching model specifically for ABR streaming. Few of them are the works in [11] and [1]. Reference [11] models the caching problem as an optimization problem. Every content has a storage budget and the aim of the algorithm is to maximize QoE of each content. The constraint in [11] is not flexible since each video content has a limited storage capacity, and this storage budget is equally for every contents. The content request rates are not taken into account in the model in [11]. Moreover, reference [11] only model the caching problem on a single node. Our work addresses cooperative caching problem. The cache nodes cooperate to leverage caching capability of each other. The amount of contents stored on each node depends not only on the storage capacity, but also on the demands of the contents and the link costs. Reference [1] models the ABR-capable caching by an optimization problem maximizing the number of concurrent requests are served while meeting the QoE requirement. The authors propose a heuristic caching algorithm called ABR-LRU-P which is a modification from the traditional eviction algorithm least-recently-used (LRU). The storage capacity constraint is not considered in this work.

The remain of the paper is organized as follows. Sections II describes the mathematical model. Section III decomposes the large-scale LP. The numerical results and conclusions are presented in Sections IV and V, respectively.

2. THE OPTIMIZATION MODEL

Consider a CDN cluster with N cache nodes providing the caching service to the users (please see the notations given in Table 1). Nodes in the network collaborate in caching to reduce the total link cost. When a user requests content m at node i. Depending on the network condition, the user receives k-th representation of content m from node i if node i has this representation already in its storage, *i.e.*, $x^m_{ki} = 1$, or from node j if $x^m_{ki} = 0$, $y^m_{kij} = 1$ (see Figure 1). In this paper, we assume that the demands for the contents on the nodes are given and remained constants during the time period of interest. All the contents are assumed to be cached in the CDN cluster.

The amount of link cost for serving the demand of content m at node i is the total link cost to transfer content m from other nodes to node i for all levels of representations, *i.e.*, $\sum_{k\in\mathcal{K}} s^m_k d^m_i \big(\sum_{\substack{j\in\mathcal{N}\\ j\neq i}} c_{ij} y^m_{kij}\big)$. Hence, the total network cost for serving its demands will be

$$\sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} s^m_k d^m_i \big(\sum_{\substack{j\in\mathcal{N}\\ j\neq i}} c_{ij} y^m_{kij}\big).$$

On the other hand, when the user that requests content m at node i receives k-th representation of content m, the user satisfaction is U_k.[1] Thus, the utility when obtaining k-th representation of content m from node i is $U_k s^m_k d^m_i \big(x^m_{ki} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y^m_{kij}\big)$. Therefore, the total utility is

$$\sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} U_k s^m_k d^m_i \big(x^m_{ki} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y^m_{kij}\big).$$

Our goal is to maximize the network performance, *i.e.*, the total utility after subtracting the bandwidth cost. The optimization model of the content placement problem is as follows.

$$\text{Max.} \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} s^m_k d^m_i \left(U_k x^m_{ki} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} (U_k - c_{ij}) y^m_{kij}\right) \quad (1)$$

$$\text{st.} \sum_{m\in\mathcal{M}}\sum_{k\in\mathcal{K}} s^m_k x^m_{ki} \leq B_i, \forall i \in \mathcal{N}, \quad (2)$$

$$y^m_{kij} \leq x^m_{kj}, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, m \in \mathcal{M}, \quad (3)$$

$$\sum_{k\in\mathcal{K}} \left(x^m_{ki} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y^m_{kij}\right) = 1, \forall i \in \mathcal{N}, m \in \mathcal{M}, \quad (4)$$

$$x^m_{ki}, y^m_{kij} \in \{0,1\}, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, m \in \mathcal{M}. \quad (5)$$

Constraint (2) is the storage capacity constraint of each node, *i.e.*, the total storage of all contents for all representations on node i must be less than its storage capacity B_i. Constraint (3) means that the amount of content m from node j serving the request at node i must be less than the amount of content m stored at node j. Constraint (4) guarantees that the amount of content m stored at node i and the total received amount of m for all representations must be equal to the whole content in order to serve the request (from the assumption that all the contents are cached inside the network).

Problem (1)-(5) is an integer linear program which is solved by a high complexity algorithm. In case the content can be stored fractionally, constraint (5) can be relaxed to

$$0 \leq x^m_{ki} \leq 1, 0 \leq y^m_{kij} \leq 1, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, m \in \mathcal{M}. \quad (6)$$

The fractional caching problem (1)-(4), and (6) becomes a linear program and can be solved with polynomial time. In this problem, x^m_{ki} is the fraction of k-th representation of content m stored on node i and y^m_{kij} is the fraction of k-th representation of content m fetched from node j to serve the request on node i.

[1]User satisfaction and quality-of-experience are used interchangeably in this paper.

Table 1: Main notations

Parameters:	
$\mathcal{M}$	set of contents
$\mathcal{N}$	the set of storage nodes in the considered network
B_i	storage capacity of node i
$\mathcal{K}$	maximum number of representations of a content
U_k	user satisfaction when receiving one unit of content at k-th representation
s_k^m	size of content m at k-th representation
d_i^m	demand for content m at node i
c_{ij}	cost for one unit of data from node j to node i
Variables:	
x_{ki}^m	decision variable indicating node i has content m at k-th streaming rate / fraction of content m at k-th streaming rate stored on node i
y_{kij}^m	decision variable indicating content m at k-th streaming rate on node j serves the demand on node i / fraction of content m at k-th streaming rate from node j serving the demand on node i

There are some additional remarks about the caching optimization formulation (1)-(5):

1. In case of an additional assumption that if a content is stored on any node up to k-level representation, all the lower-level representations are also stored exists, the formulation (1)-(5) remains unchanged except the constraint (2), which will become

$$\sum_{m\in\mathcal{M}}\sum_{k\in\mathcal{K}} S_k^m x_{ki}^m \le B_i, \forall i \in \mathcal{N}, \tag{7}$$

where $S_k^m = s_1^m + \dots s_k^m$, *i.e.*, the storage size for content m. The objective is unchanged because the utility is calculated based on the bit-rate of the highest representation.

2. The problem (1)-(5) can be utilized to model the cooperative caching among the clusters at a higher level. For example, each cluster which includes several storage nodes is considered as a node in the above formulation. The variable x_{ki}^m is the fraction of k-th representation of content m stored on cluster i, y_{kij}^m is the fraction of k-th representation of content m fetched to cluster i from cluster j.

We introduce a technique to decompose the large-scale fractional caching problem into multiple subproblems which can be solved in parallel in next section.

3. SOLVING THE LARGE-SCALE LINEAR PROGRAM

Although a linear program can be solved in a polynomial time, the caching LP is actually a large-scale problem since there is a large number of contents in the network. However, the caching LP has a special structure, the block-angular form (see Figure 2) which can be decomposed into many subproblems. To solve the problem, we first relax the non-separable constraint (2). Then, we decompose the relaxed problems into many subproblems according to the contents. Each subproblem is a linear program associated with each content.

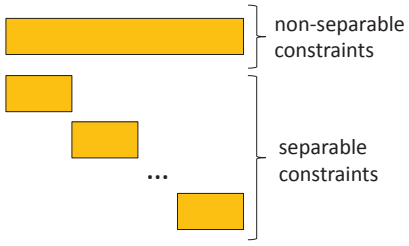

Figure 2: The block-angular structure of the linear program.

Particularly, the relaxed problem of LP has the following form

$$\begin{aligned}
\text{Max. } & \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} s_k^m d_i^m \Big(U_k x_{ki}^m + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} (U_k - c_{ij}) y_{kij}^m \Big) \\
& - \sum_{i\in\mathcal{N}} \alpha_i \Big(\sum_{m\in\mathcal{M}}\sum_{k\in\mathcal{K}} s_k^m x_{k,i}^m - B_i \Big) \qquad (8) \\
\text{st. } & y_{kij}^m \le x_{kj}^m, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, m \in \mathcal{M}, \\
& \sum_{k\in\mathcal{K}} \Big(x_{ki}^m + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y_{kij}^m \Big) = 1, \forall i \in \mathcal{N}, m \in \mathcal{M}, \\
& 0 \le x_{ki}^m, y_{kij}^m \le 1, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, m \in \mathcal{M}, \\
& \alpha_i \ge 0, i \in \mathcal{N},
\end{aligned}$$

where $\boldsymbol{\alpha}$ is the Lagrange multiplier associated with constraint (2). Therefore, the subproblem associated with content m is as follows:

$$\begin{aligned}
\text{Max. } & \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} s_k^m d_i^m \Big((U_k - \frac{\alpha_i}{d_i^m}) x_{ki}^m + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} (U_k - c_{ij}) y_{kij}^m \Big) \qquad (9) \\
\text{st. } & y_{kij}^m \le x_{kj}^m, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j, \\
& \sum_{k\in\mathcal{K}} \Big(x_{ki}^m + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y_{kij}^m \Big) = 1, \forall i \in \mathcal{N}, \\
& 0 \le x_{ki}^m, y_{kij}^m \le 1, \forall k \in \mathcal{K}, i, j \in \mathcal{N}, i \neq j.
\end{aligned}$$

Giving $\boldsymbol{\alpha}$ in each iteration, M subproblems (the number of contents in the network, $|\mathcal{M}|$) are solved parallelly. We apply Dantzig-Wolfe decomposition technique to find the optimal multipliers in each iteration by solving the master problem as described in [6, chap.12]. The objective of the master problem monotonically decreases in each iteration and converges to the optimal solution of the original LP.

In case there is symmetry in the network, we can reduce the size of the problem. Note that a LP can have multiple solutions. If two variables are symmetric in both objective and constraints, there is no guarantee that their optimal values have to be the same. However, if we only focus on finding one solution of the problem, then the symmetric solution is one of the optimal solutions of the LP. We utilize this feature to reduce the size of the large-scale problem.

4. NUMERICAL RESULTS

We assume the demands of the contents follow Zipf - Mandelbrot distribution with shape parameter $a = 1$ and shift

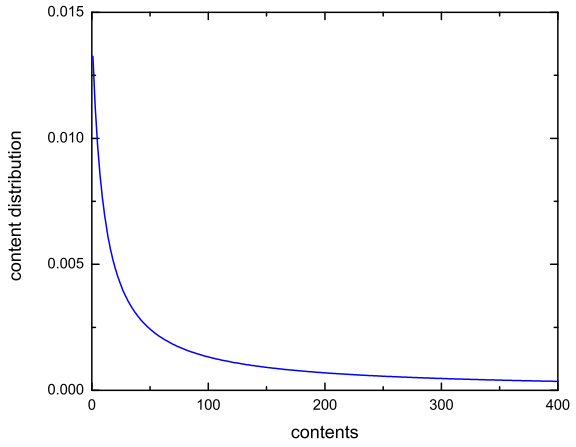

Figure 3: The distribution of 10,000 contents with $a = 1$ and $q = 10$ (only the populations of first 400 contents are plotted).

Table 2: Base-line and additional main and additional high profile settings recommended by Apple HTTP live Streaming (16:9 aspect ratio), [3].

Representations (k)	1	2	3
Resolutions (16:9 aspect ratio)	224p	720p	1080p
Bit-rates (kbps)	240	2,540	25,000

parameter $q = 10$ in the experiments [7]. The request are uniformly distributed among the nodes. Hence, the demand of m-th popularity content at each node in a considered interval is given by

$$d_i^m = r_i \frac{(q+m)^{-a}}{\sum_{m=1}^{M}(q+m)^{-a}}, \tag{10}$$

where r_i is the number of requests to node i per second. For example, Figure 3 shows the distribution of first 400 contents in 10,000-content catalog with $a = 1$, $q = 10$, and $r = 1$.

We consider the recommended representation set of Apple's HTTP live streaming [3]. For simplicity, three levels of representations, which are chosen from base-line, additional main, and additional high profile settings, represent low, medium, and high performance representations, respectively (see Table 2). We assume that every contents are all 10-minute playback time. The size of representation is approximated to $s = r \times T/8/1024$ MB, where r is the corresponding bit-rate (in Kbps) of the representation and $T = 600$ is the video playback time in seconds. Specifically, $\boldsymbol{s} = [17.58, 186.04, 1831.06]$ MB corresponding to the base-line, additional main, and additional high representations. Logarithmic utility is used [11, 12]. Without loss of generality, we assume that $U = \log(r)$. Therefore, the utilities of the user corresponding to three type of representations are $\boldsymbol{U} = [2.38, 3.41, 4.40]$.

4.1 A toy model

In this simulation, we assume that the network has 2 nodes with the storages $B = 4$ GB. There are 6 contents in the network. The shift parameter $q = 10$ and the shape parameter $a = 1$. Number of requests per second to nodes#1 and #2 are 1 and 1.5 requests per considered interval, respectively. The link cost between two cache nodes is 4.

Table 3: The optimal content placement strategy for the toy model ($\boldsymbol{x}$).

Node 1 (x_{k1}^m):

k \ m	1th	2th	3th	4th	5th	6th
1 (base-line)	0	0	0	0.66	1	1
2 (add. main)	0	0	0	0	0	0
3 (add. high)	1	0	0.82	0.34	0	0

Node 2 (x_{k2}^m):

k \ m	1th	2th	3th	4th	5th	6th
1 (base-line)	0	0	0	0	0	0
2 (add. main)	0	0	0	0	0	0
3 (add. high)	1	1	0.18	0	0	0

Table 4: The optimal content fetching strategy for the toy model ($\boldsymbol{y}$).

From node 1 to node 2 (y_{k21}^m):

k \ m	1th	2th	3th	4th	5th	6th
1 (base-line)	0	0	0	0.66	1	1
2 (add. main)	0	0	0	0	0	0
3 (add. high)	0	0	0.82	0.34	0	0

From node 2 to node 1 (y_{k12}^m):

k \ m	1th	2th	3th	4th	5th	6th
1 (base-line)	0	0	0	0	0	0
2 (add. main)	0	0	0	0	0	0
3 (add. high)	0	1	0.18	0	0	0

Node#1 stores the first and part of third and fourth contents at additional high representation; fifth, sixth, and part of fourth contents at base-line representations. Node#2 with higher request rate stores first, second, and part of third contents at additional high representations (see Table 3). Table 4 shows that except the first content, node#2 receives all the content from node#1 to serve node#2's requests. Node#1 receives the second and part of third contents at baseline from node#2 to serve node#1's requests.

4.2 Trade-off between performance and bandwidth cost

We now consider a the cluster network includes 10,000 contents stored on 10 cache nodes. Each node has 100 GB of storage for the video contents. The number of requests to the cluster of node is 10 request per second and is uniformly distributed among the nodes ($r_i = 1$ request per second). The link costs between every pair of nodes are all equal, *i.e.*, c. We increase c gradually. Figure 4 shows the trade-off between the overall performance, which is calculated by subtracting the network cost from the total utility of the network, and the link cost c. The higher cost decreases the overall performance. When c increases gradually, the network cost also increases. However, if the link cost is too high, the network cost decreases since it costs too much to transfer the representations between the nodes. The local

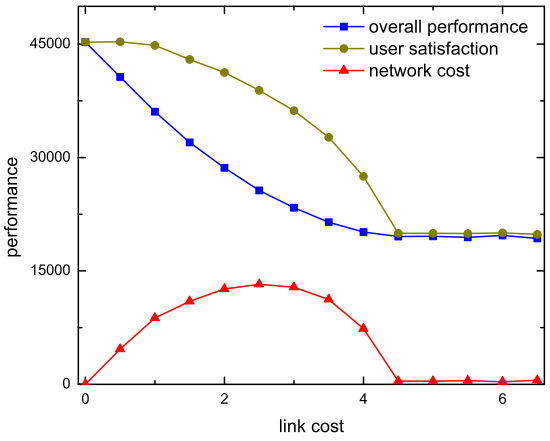

Figure 4: The performance of the network (network utility - bandwidth cost) when the link cost is gradually increased.

node tends to serve the requests by its currently stored representations rather than downloads the better performance representations from the other nodes.

5. CONCLUSIONS

We have proposed a linear programming model for the optimal content placement which supports adaptive bit-rate streaming. The performance and the bandwidth cost of the network is optimized. We also propose a technique to decompose the LP to many subproblems which can be solved in parallel. Each subproblem corresponds to a content. The simulations demonstrate the content placement on the cache nodes and the trade-off between the link cost and the performance of the network.

6. ACKNOWLEDGMENTS

This research was funded by Vietnam National University HoChiMinh City (VNU-HCM) in 2015 and the MSIP(Ministry of Science, ICT & Future Planning), Korea in the ICT R&D Program 2014. Dr. CS Hong is the corresponding author.

7. REFERENCES

[1] H. Ahlehagh and S. Dey. Adaptive bit rate capable video caching and scheduling. In *Proceeding of IEEE Wireless Communications and Networking Conference*, WCNC'2013, pages 1357–1362, April 2013.

[2] S. Akhshabi, A. C. Begen, and C. Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 157–168, New York, NY, USA, 2011. ACM.

[3] Apple. Using http live streaming. [Online]. Available: http://goo.gl/fJIwC.

[4] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan. Optimal content placement for a large-scale vod system. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 4:1–4:12, New York, NY, USA, 2010. ACM.

[5] S. Borst, V. Gupta, and A. Walid. Distributed caching algorithms for content distribution networks. In *Proceedings of the IEEE Conference on Computer Communications*, INFOCOM'10, pages 1 –9, march 2010.

[6] S. P. Bradley, A. C. Hax, and T. L. Magnanti. *Applied mathematical programming*. Addison-Wesley, 1977.

[7] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. *Proceedings of the IEEE Conference on Computer Communications*, INFOCOM'99, pages 126 –134, vol.1, Mar 1999.

[8] Cisco. Cisco visual networking index: Forecast and methodology, 2012-2017. 2013.

[9] T. Stockhammer. Dynamic adaptive streaming over http: Standards and design principles. In *the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 133–144, New York, NY, USA, 2011. ACM.

[10] H. Xie, G. Shi, and P. Wang. Tecc: Towards collaborative in-network caching guided by traffic engineering. In *Proceedings of the IEEE Conference on Computer Communications*, INFOCOM'12, pages 2546 –2550, march 2012.

[11] W. Zhang, Y. Wen, Z. Chen, and A. Khisti. Qoe-driven cache management for http adaptive bit rate streaming over wireless networks. *IEEE Transactions on Multimedia*, 15(6):1431–1445, Oct 2013.

[12] P. Reichl, B. Tuffin, and R. Schatz. Logarithmic laws in service quality perception: where microeconomics meets psychophysics and quality of experience. *Telecommunication Systems*, vol. 52, no. 2, pp. 587–600, 2013.