A versatile Markov chain model for the performance analysis of CCN caching systems

Hamza Ben-Ammar, Yassine Hadjadj-Aoul, Gerardo Rubino, Soraya Aït-Chellouche

A versatile Markov chain model for the performance analysis of CCN caching systems

Hamza Ben-Ammar, Yassine Hadjadj-Aoul, Gerardo Rubino and Soraya Ait-Chellouche
Univ Rennes, Inria, CNRS, IRISA
Emails: {hamza.ben-ammar, yassine.hadjadj-aoul, gerardo.rubino, soraya.ait-chellouche}@irisa.fr

***Abstract*—Beyond Content Delivery Networks (CDNs), Network Operators (NOs) are developing caching capabilities within their own network infrastructure, in order to face the rise in data consumption and to avoid the potential congestion at peering links. These factors explain the enthusiasm of industry and academics around the Content-Centric Networking (CCN) concept and its in-network caching feature. Many contributions focused these last years on improving the caching performance of CCN. In this paper, we propose a very versatile model capable of modeling the most efficient caching strategies. We first start by representing a single generic cache node. We then extend our model for the case of a network of caches. The obtained results are used to derive, in particular, the cache hit probability of a content in such caching systems. Using a discrete event simulator, we show the accuracy of the proposed model under different network configurations.**

***Index Terms*—CCN, Caching, Markov Chain, Modeling.**

I. Introduction

The increase in data consumption over the past few years has allowed CDNs to be at the center of the content distribution value chain [1]. Internet service providers (ISPs), for their part, are struggling to benefit from this increase. ISPs are, thus, investigating the possibilities of extending their infrastructure with caching capabilities to finally be part of the value chain.

The advent of Content-Centric Networks (CCNs) represents a real opportunity for ISPs [2]. In fact, these networks allow focusing on the content itself and not on its location, which in turn allows to overcome the limitations of the current Internet. Therefore, the end-users' requests (known as interests in CCN), that are routed to the Content Providers' servers, can be satisfied by the cached data at the intermediate nodes, thus reducing the round-trip time, network traffic and servers' load.

When content caching is possible, a significant improvement can be achieved, as shown in several studies [3]. The analytical quantification of caching performance is, however, not sufficiently explored in the literature. In fact, several issues need to be addressed in order to understand the behavior of such a caching network. Indeed, in addition to the cache hit probability, other metrics deserve a deeper analysis, such as the average number of hops to reach a particular content, or the impact of cache size and traffic pattern on performance.

In our previous work [4], we proposed an analytical model based on Markov chains to estimate the cache hit probability under the popular Least Recently Used (LRU) replacement scheme for a system with multiple caching nodes, where the default caching strategy of CCN, called Leave-Copy-Everywhere (LCE), is used. LCE consists in caching the requested contents in all the nodes along the downloading path. In this paper, we extend our work by proposing a versatile and general model, which is able to mimic the most efficient caching strategies developed for CCN like Leave Copy Down (LCD) and LRU-K [5] [6]. Let's recall that in a multi-cache system, the decision of whether a data packet should be stored or not in the cache is managed using a caching scheme. Here, we intend to propose a methodology that allows modeling this process in general, that can be used to analyze different caching algorithms.

We consider in this work a network of caches in which the requests arrive independently and follow a popularity law (i.e. Zipf), which is generally the case in real systems [7]. All the requests are forwarded through the shortest path to the nearest source. We start by proposing a discrete time Markov chain to model a single cache node. This model is, then, generalized to a system of caches. Compared to most of the existing contributions in the literature, the proposed solution can easily deal with different network topologies and allow having a good approximation of such type of caching systems.

The reminder of this paper is organized as follows. In Section II, we present the related work, which focuses on contributions providing an analytic model of caching systems. Section III provides the detailed description of the proposed model. Then, the evaluation of our proposal is introduced in Section IV. Finally, Section V summarizes the achievements of the paper and introduces our future work.

II. Related Work

Many studies have been conducted to deal with the performance analysis of a single cache and a network of caches [8]. The study in [9], was probably the first attempt to evaluate and model caching systems. The author proposed a model for predicting the buffer hit probability under the LRU and FIFO replacement policies. Unfortunately, the computational complexity of this model grows exponentially with the cache size C and the number of data items R. Dan and Towsley proposed, in [10], an algorithm, with a complexity of $O(CR)$, for predicting an approximate cache hit probability under the LRU replacement policy. The proposed solution is, however, limited to the case of a single cache.

Che et al. developed, in [11], an analytic modeling technique, which was further investigated in [12]. The solution allows identifying a characteristic time approximation for each item in the cache, which was used to estimate the cache hit

rate. In [10], the authors extended this algorithm to the case of a network of caches. Psaras et al. proposed, in [13], a Markov chain-based caching model to estimate the proportion of time a given piece of content is cached, in the case of a single router. They also extended their model to cover the case of multiple caches. Nevertheless, the aforementioned proposals are applicable only when the LCE caching strategy is used.

Recently, there have been many proposals that treated caching schemes different from LCE in systems with multiple caches. In [14], the authors extended the work of [11] and proposed a unified framework to analyze the performance of caches (both isolated and interconnected). Their model covers various insertion and eviction policies (including LRU). They evaluated the accuracy of their proposal through simulation. However, in the case of interconnected caches, the tests were conducted only in the case of a 6-nodes chain with fixed network settings. The authors in [15] developed algorithms to approximate the hit probability of cache replacement algorithms that are variants of LRU and similar to LRU-K [6], by extending the work of [11]. Their study is, however, limited to the case of single caches.

III. A general Markov chain model for multi-cache systems

We use here the LRU algorithm to manage the node's content store. Other memory management algorithms have been studied in the case of cache modeling like First In First Out (FIFO) and Random Replacement (RR). However, LRU is known to perform much better than FIFO [16] and the expected long-run performances of both RR and FIFO replacement policies were shown to be roughly the same [17].

A. System description

In CCNs, the content's name is the only identifier of data. To request and retrieve data, two types of packets are commonly used [2]: Interest Packet and Data Packet. Clients can ask for specific data objects by sending Interest packets, which are forwarded towards the data sources using the Forwarding Information Base (FIB). A record of the forwarded Interests is kept in the Pending Interest Table (PIT) in order to keep track of the Interests waiting for a data packet. When a node receives multiple requests for the same content, the Interest packets will be aggregated in one entry in the PIT and only the first one is routed. Once the requested content is found, it is automatically routed back to the clients on the reverse path. All the nodes along this path can store a copy of data to answer future demands.

Let $G = (V, E)$ be a graph representing a network of caches, where $V = \{v_1, ..., v_M\}$ is the set of nodes and $E \subset V \times V$ is the set of links between the nodes. Let $C = \{c_1, ..., c_R\}$ be the set of contents available for the users (the catalog). We assume that all the accessible contents in the system have the same size. The cache capacity is then expressed in terms of the number of objects that can be stored. All available contents are stored permanently at one or more servers attached to some nodes within the network. In the rest of the paper and for the sake of readability, we will use the term node/cache interchangeably as well as the terms rank/popularity and content/item/object.

Clients, which are attached to the network nodes, send requests into the network looking for content. The pattern of these requests is characterized by the Independent Reference Model (IRM) [10]. Users generate, thus, an independent and identically distributed sequence of requests from the catalog C of R objects. Specifically, the probability p_r to request an item c_r from the set of available contents is constant and follows a popularity law, where the contents are ranked decreasingly according to their popularity from 1 to R. Since, in our work, we address video services, the contents feature a skewed popularity distribution. As already argued in many previous studies, the latter fits the Zipf law [18]: the probability to request the content of rank r is: $p_r = r^{-\alpha} / \sum_{i=1}^{R} i^{-\alpha}$, where α, the skew of the distribution, depends on the type of the accessible objects [12]. For our approach, we only need to assume that $0 < p_r < 1$, for $R \geq 2$.

B. LRU cache analysis

Let's consider a node in a CCN operating under the LRU replacement policy and having clients attached to it. Whenever a user requests a content of rank r in the catalog, it will generate a cache miss if the content is not present in the cache, or otherwise a hit. In the latter case, the object will be sent back to the user. In the case of a cache miss, the client's interest is forwarded to the next nodes in the direction of the nearest content server storing a permanent copy (i.e. origin server). Once located, the object is sent on the reverse path and depending on the caching strategy used in the network, the content will be cached or not in the nodes it passes through. In the general case, this decision can be seen as the probability that we denote $\beta(r)$, with which a received content c_r will be stored in a cache. The value of $\beta(r)$ will then depend on the caching strategy adopted by the node. In the LCE policy, every object will always be cached by each node that passes by it, that is, $\beta(r) = 1$ for any content c_r. In the LCD scheme [5], for example, where a requested object is cached only on the node that resides immediately below the location of the hit, $\beta(r)$ will be equal to the cache hit probability of the content on the parent node.

Suppose now that we have a cache managed with LRU and sized to contain N items, $N \ll R$. Whenever a local cache hit or a caching decision occurs for an object, the object is placed at the top position in the cache. When a content c_r is received, and for any given content $c_{r'}$ (with $r' \neq r$) occupying position i, three actions are then possible:

- $c_{r'}$ will be moved down by one position if c_r is outside the cache and it has been decided to cache it (with probability $\beta(r)$) or if it occupies block j with $j > i$;
- $c_{r'}$ will remain at the same position if c_r occupies block j, with $j < i$, or if it is outside the cache and it has been decided to not cache it (with probability $1 - \beta(r)$);

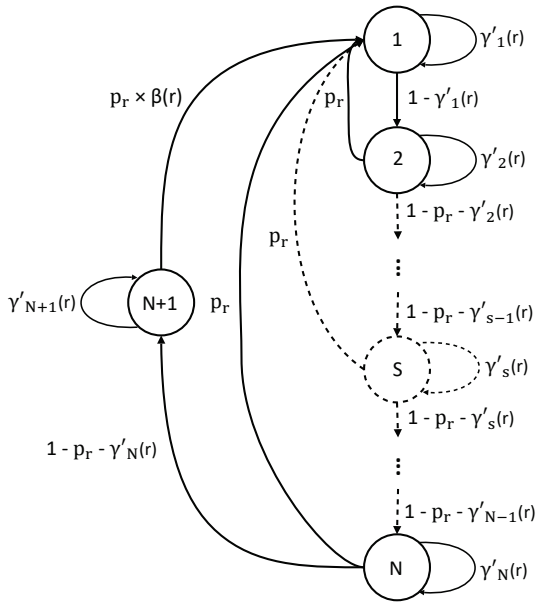

Figure 1: General Markov chain model for a content c_r in an LRU cache

- $c_{r'}$ will be evicted from the cache if it occupies the N^{th} position (last block) and if it has been decided to cache c_r (with probability $\beta(r)$), which is outside the cache.

Let us call a *configuration* of the cache any vector $\vec{x} = (x_1, \ldots, x_N)$, where x_i is the content present at position i. We only consider the case where all positions are occupied because the cases where the cache is partially filled concerns only the initial transient phase. Let us denote by S the set of all possible configurations; we have $|S| = R!/(R-N)!$.

The problem with exact analysis is the huge size of the state space S and the complexity of the chain. In the next section, we describe an efficient way of obtaining an approximation for such a metric that is shown to be very accurate compared to results obtained using simulations.

C. A general single LRU cache model

In our previous work [4], we modeled the LCE caching strategy where every data packet is cached at every node that passes by it, which matches the case where the caching decision probability (represented by $\beta(r)$) is always equal to 1. In this paper, we extend our previous model MACS (Markov chain-based Approximation of CCN Caching Systems) to cover the general case considered here.

Let us consider a Markov chain $X(r) = \big(X_h(r)\big)_{h\geq 0}$ with $N+1$ states, as depicted in Figure 1. This chain will represent the evolution with time of the position occupied by content c_r in the cache, where state $N+1$ means that content c_r is absent, state 1 means that the object is at the top of the cache and state N that it is at the bottom. The probability that content c_r stays at the same position s upon the reception of another item is denoted $\gamma'_s(r)$.

Assume we know $\gamma'_i(r)$, $i = 1, 2, \ldots, N, N+1$, satisfying $0 < \gamma'_i(r) < 1$. This means that $X(r)$ is irreducible and aperiodic. Let us denote by $\pi(r) = (\pi_1(r), \pi_2(r), \ldots, \pi_{N+1}(r))$ its equilibrium distribution. Assume now that $\pi(r)$ is exactly the marginal distribution corresponding to content c_r and that the chains $X(1), \ldots, X(R)$ are independent of each other. The probability that a content of rank r remains in state s of the chain, $s \in \{1, 2, \ldots, N+1\}$, after a request arrives, is

$$\begin{cases} \gamma'_s(r) = \gamma_s(r) + \displaystyle\sum_{i=1, i\neq r}^{R} (1-\beta(i)) p_i P_{miss}(i), \ 1 \leq s \leq N, \\ \gamma'_{N+1}(r) = 1 - p_r \beta(r). \end{cases} \tag{1}$$

The value of $P_{\text{miss}}(r)$ used in (1) represents the cache miss probability of content c_r (its calculation will be detailed later) and the value of $\gamma_s(r)$ is a special case of $\gamma'_s(r)$ where $\beta(r)$ is 1, and it is equal to

$$\begin{cases} \gamma_1(r) = p_r, \\ \gamma_s(r) = \displaystyle\sum_{i=1, i\neq r}^{R} p_i \sum_{j=1}^{s-1} \pi_j(i), \qquad 2 \leq s \leq N, \\ \gamma_{N+1}(r) = 1 - p_r. \end{cases} \tag{2}$$

Again, observe that (1) is an approximation. Let's denote by $T_{i,j}$ the transition probability from state i to j in our model. Then, $T_{i,j}$ is equal to:

$$\begin{cases} T_{i,i} = \gamma'_i(r), \qquad 1 \leq i \leq N+1, \\ T_{i,1} = p_r, \qquad 2 \leq i \leq N, \\ T_{N+1,1} = p_r \beta(r), \\ T_{1,2} = 1 - \gamma'_1(r), \\ T_{i,i+1} = 1 - p_r - \gamma'_i(r), \qquad 2 \leq i \leq N, \\ T_{i,j} = 0, j \notin \{1, i, i+1\}. \end{cases} \tag{3}$$

The distribution $\pi(r)$ satisfies the Chapman-Kolmogorov equations:

$$\begin{cases} \pi_i(r) = \displaystyle\sum_{j=1}^{N+1} \pi_j(r) T_{j,i}, \qquad 1 \leq i \leq N+1, \\ \displaystyle\sum_{i=1}^{N+1} \pi_i(r) = 1. \end{cases} \tag{4}$$

If we develop the system of equations defined in (3) and (4), we obtain:

$$\begin{cases} \pi_1(r) = \gamma'_1(r)\pi_1(r) + p_r \displaystyle\sum_{i=2}^{N} \pi_i(r) + p_r \beta(r) \pi_{N+1}(r), \\ \pi_2(r) = \dfrac{(1-\gamma'_1(r))\pi_1(r)}{1-\gamma'_2(r)}, \\ \pi_i(r) = \dfrac{(1-p_r-\gamma'_{i-1}(r))\pi_{i-1}(r)}{1-\gamma'_i(r)}, \quad 3 \leq i \leq N+1, \\ \pi_1(r) + \pi_2(r) + \cdots + \pi_{N+1}(r) = 1. \end{cases} \tag{5}$$

By computing $\pi_{N+1}(r)$, the cache miss probability, we can obtain the cache hit rate $1 - \pi_{N+1}(r)$ (with computational complexity of $O(NR)$). We can see from (5) that each $\pi_i(r)$ depends on $\pi_{i-1}(r)$ ($2 \leq i \leq N+1$), and once we get

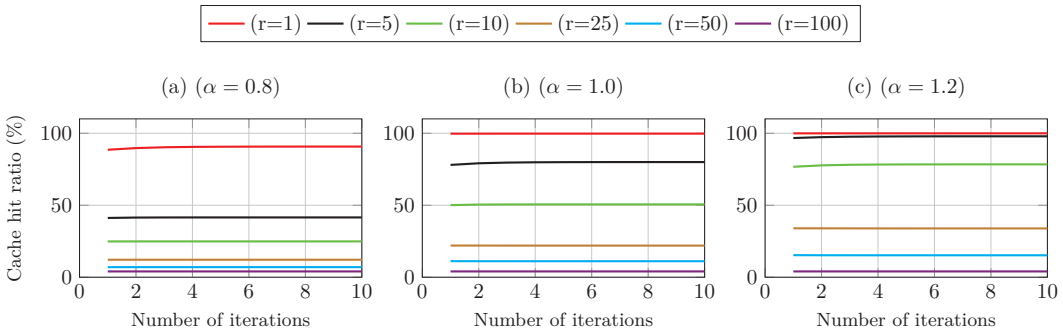

Figure 2: Cache hit ratio vs number of iterations in our algorithm of contents with different popularities under a single node ($\beta(r)$ = 0.5, catalog = 20000, cache size = 0.25% of the catalog)

$\pi_1(r)$, we can calculate $\pi_{N+1}(r)$ and deduce the cache hit probability. When $\beta(r) = 1$, we have $\pi_1(r) = p_r$. The equations then derived in (5) can be easily solved using (2):

$$\begin{cases} \pi_1(r) = p_r, \\ \pi_2(r) = \dfrac{p_r(1-p_r)}{1-\gamma_2(r)}, \\ \pi_i(r) = \dfrac{p_r(1-p_r)\prod_{j=2}^{i-1}(1-p_r-\gamma_j(r))}{\prod_{j=2}^{i}(1-\gamma_j(r))}, 3 \leq i \leq N+1. \end{cases} \tag{6}$$

In the general case ($0 \leq \beta(r) \leq 1$), $\pi_1(r)$ cannot be computed directly, it depends on all the other values of $\pi_i(r)$ (for $2 \leq i \leq N+1$). One way to solve this problem is to modify the expression of $\pi_1(r)$ in order to reduce its dependency on the other variables. To do so, we first add and subtract in the expression of $\pi_1(r)$ the value $p_r \times \pi_{N+1}$. Then, using the expression of $\gamma_1'(r)$, and since $\sum_{i=1}^{N+1} \pi_i(r) = 1$, we get a new expression of $\pi_1(r)$:

$$\pi_1(r) = \frac{p_r(1 + (\beta(r)-1)\pi_{N+1}(r))}{1-(\gamma_1'(r)-\gamma_1(r))}. \tag{7}$$

Now, $\pi_1(r)$ depends only on $\pi_{N+1}(r)$. The idea then is to use a fixed-point iteration in order to generate successive approximations to the solution, which consists on finding $\pi_{N+1}(r)$, starting from an initial guess. To do so, we start by considering an approximate value of $\pi_{N+1}(r)$ (that we denote by $\pi'_{N+1}(r)$) by using the one obtained when $\beta(r) = 1$, to compute $\pi_1(r)$. Once we have $\pi_1(r)$, we can calculate $\pi_i(r)$ ($2 \leq i \leq N+1$) using (5) to finally obtain a new value of $\pi_{N+1}(r)$. The value of $\pi'_{N+1}(r)$ is also used to calculate $\gamma'_s(r)$ ($s \in \{1, ..., N\}$) by replacing $P_{miss}(r)$ with $\pi'_{N+1}(r)$. These steps are repeated until we converge to the final solution by replacing in each step $\pi'_{N+1}(r)$ by the new computed value $\pi_{N+1}(r)$ (Figure 2 shows that convergence is fast).

D. Multiple caches system

Following common practice [10] [11], we assume in this work that after a cache miss and when a content is decided to be cached by a node, it will be downloaded instantaneously. Let's consider a system of multiple CCN nodes where the contents are forwarded according to the Shortest Path Routing (SPR) algorithm [19]. With SPR, when a client's interest cannot be satisfied by a node, it is forwarded along the shortest path to the closest permanent copy of the requested content. In this case, each node has to take into account, in addition to the local requests, the interests that come from other nodes due to a cache miss (we denote this stream of interests by "miss stream" or *MS*). The outgoing miss stream rate from a node u of a content c_r is equal to

$$MS_r(u) = req(r,u) \times \pi_{N+1}(r,u), \tag{8}$$

where $req(r,u)$ is the total proportion of requests for c_r received by u and $\pi_{N+1}(r,u)$ is the miss probability of content c_r at u. In CCNs the interests for the same object received by a node will be aggregated and only the first one is sent to the next nodes. This feature should be considered when computing the total miss stream received by a node having more than one child node. The incoming miss stream ratio for an object with a rank r at a node v, that we denote by $\eta_r(v)$, is equal to:

$$\eta_r(v) = \sum_{u:NH(u)=v} \big(MS_r(u) \prod_{w \neq u:NH(w)=v} \big(1-MS_r(w)\big)\big). \tag{9}$$

The set $\{u : NH(u) = v\}$ represents the nodes having v as the next hop in the shortest path toward the source. The value of $(1 - MS_r(w))$ represents the case where an interest sent from the node w is discarded because it was already received by another node. The probability that a node v will receive a request for c_r will no longer be p_r, but another value that we denote as p'_r, which will take into account in addition to the local requests, the interests due to a cache miss from previous nodes. For each node v, this value is equal to:

$$p'_r = \frac{p_r + \eta_r(v)}{\sum_{k=1}^{R}(p_k + \eta_k(v))} = \frac{p_r + \eta_r(v)}{1 + \sum_{k=1}^{R} \eta_k(v)}. \tag{10}$$

Consider again the previous Markov chain (see Figure 1). For every node v, we can compute the stationary state probabilities as we did in the case of a single node, by replacing p_r with p'_r in equations (1), (2) and (5). As we mentioned previously, the cache hit probability of a content with popularity r is equal to $1 - \pi_{N+1}(r)$. To compute the cache hit performance of a multi-cache system, we start by

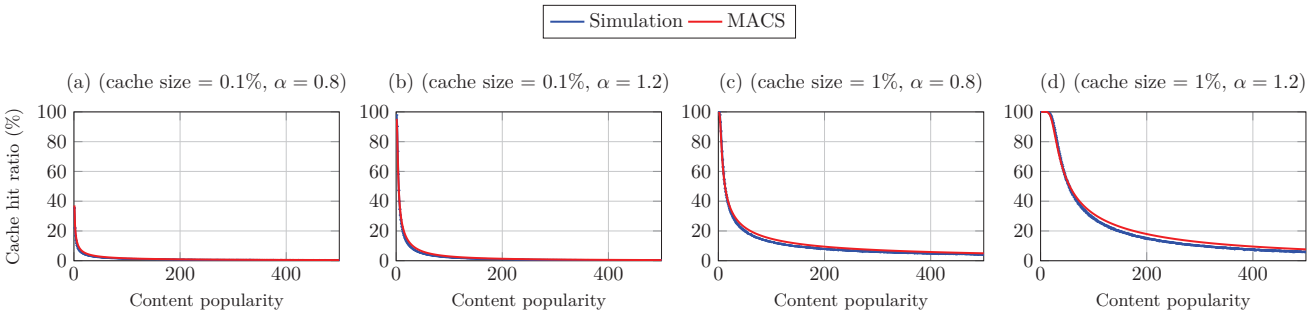

Figure 3: Total hit probability vs content popularity under binary tree topology (31 nodes)

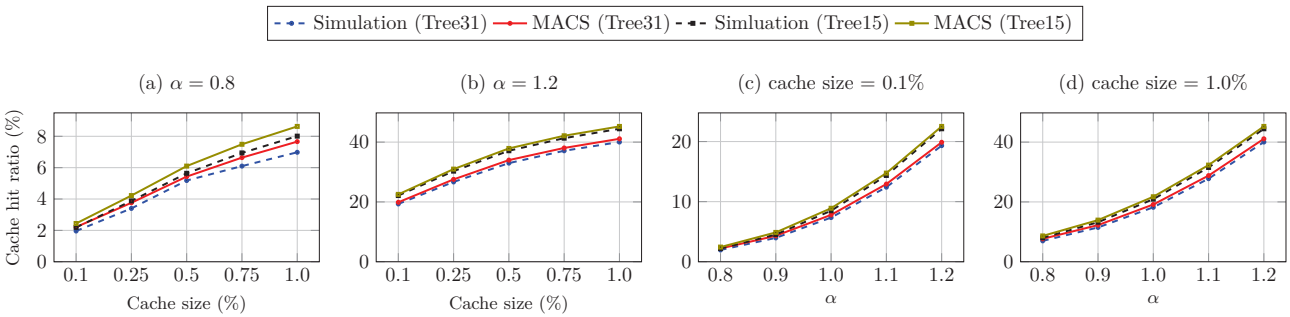

Figure 4: Total hit rate of the network under binary tree topology (31 and 15 nodes)

treating the leaf nodes of the network since in our model, each node needs to know all the incoming stream of requests, including those received due to cache miss from previous nodes. Starting from the leaves, we go through the core nodes of the network until arriving at the source node, where the permanent copies of the catalog's objects are attached.

IV. Performance evaluation

A. Simulation environment

In order to evaluate the accuracy of our proposal, we compared the analytic model presented in previous section with the results of simulations using ccnSim [20], which is a discrete-event and a chunk-level simulator for CCNs. The accuracy of MACS, compared to the simulation results, can be affected by many parameters. Our focus on the conducted experiments was on the following key settings: cache size and Zipf law's skew distribution value.

The simulations were conducted on two complete binary trees with 31 and 15 nodes[1]. In the simulation settings, we considered a catalog of contents containing $20,000$ 1-chunk sized objects whose popularity distribution follows the Zipf's law. Permanent copies of the available contents were hosted on one repository attached to the root node of the network. We set a uniform cache store capacity on the CCN nodes, which was defined as a proportion of the catalog size. Different simulations were conducted with a cache store size varying from 0.1% to 1.0% of the catalog capacity. The clients, attached to the network's leaves, generated requests according to a Poisson process with a rate per client corresponding to one request per second (each client representing an aggregate of users). We tested also different values of the Zipf law's skew parameter α going from 0.8 to 1.2. As shown in many studies [18], these two values correspond to the Zipf popularity exponent in the case of User Generated Content (UGC) and Video on Demand (VoD), respectively. As we mentioned previously, the Least Recently Used (LRU) is used as a cache replacement policy and the value of $\beta(r)$ in our model is fixed to 0.5[2]. Next, we will expose and compare the cache hit results obtained with our analytical model and with the ccnSim simulation tool (due to lack of space, only the cache hit rate metric is shown and results related to specific caching schemes are not exposed). The numerical simulation results, shown in the graphs, depict the mean values taken over 30 runs, where 10^6 requests are sent in the network when the system reaches stability (i.e., all the caches become full) with error bars representing 99% confidence intervals.

B. Model results and analysis

Figure 3 shows the values of the average cache hit ratio versus the content popularity under different scenarios using a simulator and our analytical model MACS. For the sake of clarity, we considered in the graphs only the objects whose popularity goes from 1 to 500. We can see from the charts that our analytical model gives in average an accurate hit rate for the whole range of item population with different network settings. When the cache size is set to a low value (0.1% of the catalog), the model performs better even with distinct values of α and for different types of content popularity (see Figures 3(a) and 3(b)). In the case when the cache capacity is set to 1% of the catalog size (see Figures 3(c) and 3(d)), the

[1]Realistic network topologies can be also used as we did in [4].

[2]The results obtained using other values of $\beta(r)$ are very similar in terms of accuracy.

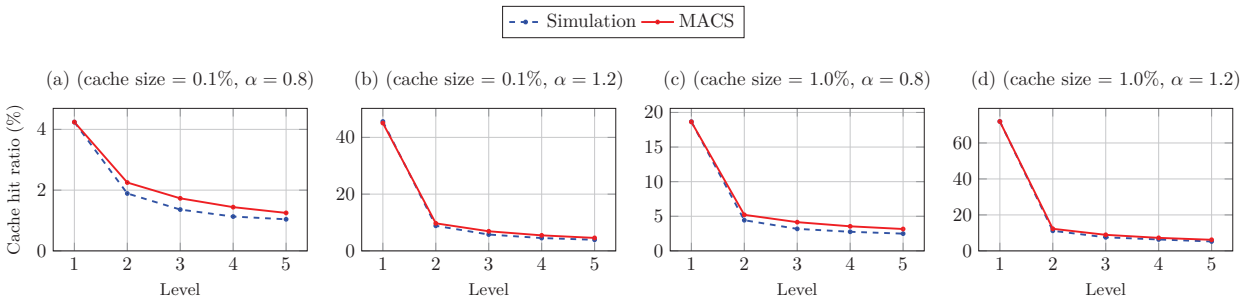

Figure 5: Total hit rate at different layers of the network under binary tree topology (31 nodes)

cache hit ratio per content in our model is estimated with a slightly lower accuracy compared to the other case (cache size of 0.1%). However, the cache hit approximation of the highly-popular objects remains precise in the different scenarios.

In Figure 4, we report the total hit probability as a function of the cache size and the Zipf law parameter α with two different network topologies. The goal here is to test a large range of values in the network configurations and see the model's performance in terms of accuracy. We observe that doubling the network size did not affect MACS performance since accuracy is similar in both tested networks. The results also indicate that the error rate of our model goes up when the total hit ratio is low. It can especially be seen when the value of α is equal to 0.8, where the cache exhibits a relatively poor performance. In the other cases, we obtain low error rates.

Figure 5 reports the cache hit rate at different levels of the network topology. Level one being the leaf caches and level five represents the root node. The results show that we get the best match between simulations and our model at the leaf nodes. The model's accuracy is slightly reduced at higher levels within the network. The main cause of this inaccuracy is related to the estimation of the miss stream. At the leaf nodes, the requests received contain only those generated by the clients, which can be easily estimated. However, the incoming miss streams of the nodes located at the core of the network are more difficult to estimate since they are computed using the cache hit rate of previous nodes.

V. Conclusion

We propose in this paper MACS, a Markov chain-based Approximation of CCN Caching Systems to estimate the cache hit probability under the popular LRU replacement policy. The versatility of MACS enables us to model different caching schemes and not just one specific strategy. The obtained results show that our model, compared to the simulations conducted under the ccnSim tool, can estimate with high accuracy the cache hit rate of a multi-cache system. In the future, we intend to use the model to analyze efficient caching strategies such as LCD and LRU-K, in order to better understand the main reasons of their performance. This will include examining various metrics in addition to the cache hit ratio like the content providers' load and the average distance or latency to retrieve data.

We would like to thank the Lannion-trégor community and the Regional Council of Brittany for their financial support.

References

[1] Cisco, "Cisco visual networking index: forecast and methodology, 2015-2020," White paper, Jun. 2016.

[2] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," in *Proc. of the 5th International Conference on Emerging Networking Experiments and Technologies*.

[3] G. Rossini and D. Rossi, "A dive into the caching performance of content centric networking," in *Proc. of IEEE CAMAD*, Sep. 2012, pp. 105–109.

[4] H. Ben-Ammar, S. Ait-Chellouche, and Y. Hadjadj-Aoul, "A Markov chain-based Approximation of CCN caching Systems," in *IEEE Symposium on Computers and Communications*, Jul. 2017, pp. 327–332.

[5] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Performance Evaluation*, vol. 63, pp. 609–634, Jul. 2006.

[6] E. J. O'Neil, P. E. O'Neil, and G. Weikum, "The lru-k page replacement algorithm for database disk buffering," *SIGMOD Rec.*, vol. 22, pp. 297–306, Jun. 1993.

[7] F. Guillemin, T. Houdoin, and S. Moteau, "Volatility of youtube content in orange networks and consequences," in *2013 IEEE International Conference on Communications*, Jun. 2013, pp. 2381–2385.

[8] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Communications Surveys Tutorials*, vol. 17, pp. 1473–1499, Aug. 2015.

[9] W. F. King, "Analysis of paging algorithms," in *IFIP Congress*, Aug. 1971, pp. 485–490.

[10] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes," *SIGMETRICS Performance Evaluation Review*, vol. 18, pp. 143–152, Apr. 1990.

[11] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 1305–1314, Sep. 2002.

[12] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance," in *Proc. of the 24th International Teletraffic Congress*, Sep. 2012, pp. 1–8.

[13] I. Psaras, R. Clegg, R. Landa, W. Chai, and G. Pavlou, "Modelling and evaluation of ccn-caching trees," in *Proc. of the 10th International IFIP TC 6 Conference on Networking*, May 2011, pp. 78–91.

[14] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 1, pp. 1–20, May 2016.

[15] N. Gast and B. V. Houdt, "Ttl approximations of the cache replacement algorithms lru(m) and h-lru," *Performance Evaluation*, vol. 117, pp. 33–57, Dec. 2017.

[16] M. Chrobak and J. Noga, "LRU is better than FIFO," *Algorithmica*, vol. 23, pp. 180–185, Feb 1999.

[17] E. Gelenbe, "A unified approach to the evaluation of a class of replacement algorithms," *IEEE Transactions on Computers*, vol. C-22, pp. 611–618, Jun. 1973.

[18] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network," in *Proc. IEEE INFOCOM Workshops*, Mar. 2012, pp. 310–315.

[19] L. Wang, A. Hoque, C. Yi, A. Alyyan, and B. Zhang, "Ospfn: An ospf based routing protocol for named data networking," NDN Technical Report, Jul. 2012.

[20] R. Chiocchetti, D. Rossi, and G. Rossini, "ccnSim: An highly scalable ccn simulator," in *IEEE International Conference on Communications*, Jun. 2013, pp. 2309–2314.