Modelling and Evaluation of CCN-caching Trees[⋆]

Ioannis Psaras, Richard G. Clegg, Raul Landa, Wei K. Chai, and George Pavlou

Department of Electronic & Electrical Engineering
University College London
WC1E 7JE, Torrington Place, London, UK,
{i.psaras, r.clegg, r.landa, w.chai, g.pavlou}@ee.ucl.ac.uk

Abstract. *Networking Named Content* (NNC) was recently proposed as a new networking paradigm to realise *Content Centric Networks* (CCNs). The new paradigm changes much about the current Internet, from security and content naming and resolution, to caching at routers, and new flow models. In this paper, we study the caching part of the proposed networking paradigm in isolation from the rest of the suggested features. In CCNs, every router caches packets of content and reuses those that are still in the cache, when subsequently requested. It is this caching feature of CCNs that we model and evaluate in this paper.
Our modelling proceeds both analytically and by simulation. Initially, we develop a mathematical model for a single router, based on continuous time Markov-chains, which assesses the proportion of time a given piece of content is cached. This model is extended to multiple routers with some simple approximations. The mathematical model is complemented by simulations which look at the caching dynamics, at *the packet-level*, in isolation from the rest of the flow.

Keywords: Content-Centric Networks, Packet Caching, Markov Chains

1 Introduction

The Internet today operates in a *machine-resolution* fashion, that is to say, content is accessed via requests to given hosts. However, the user is not usually interested in a host, but in specific content. Users are essentially interested in *content-resolution*, or *service-resolution*. Recently, the influential paper [8] introduced the *Networking Named Content* (NNC) overlay paradigm as a means to realise *Content-Centric Networks* (CCN). The paper suggests numerous architectural changes to networks based upon the CCN paradigm. For example, according to the NNC approach, names are hierarchical, can be aggregated, are structured, and also human-readable, at least to a certain extent. The approach also brings certain security benefits with named content being signed and secure. The present paper models just one of those proposed changes: the caching

⋆ The research leading to these results has received funding from the EU FP7 COMET project - Content Mediator Architecture for Content-Aware Networks, under Grant Agreement 248784.

strategy used in CCN. This is modelled in isolation from the other proposals within [8], i.e., security, content-resolution scalability and routing are subject to different modelling and evaluation approaches and therefore, of different studies.

The NNC scheme proposed in [8] changes the whole Internet flow model from a sender-driven, TCP-like buffering/congestion controlled environment to a receiver-driven, caching (instead of buffering) scheme. In particular, by exploiting the fact that each data packet can now be identified by name, CCN can take advantage of forwarding the packet to many interested recipients. That is, instead of buffering a packet until it is forwarded to the interested user and then discarding it, as happens today, CCN first forwards the packet to the interested user and then *"remembers"* the packet, until this packet *"expires"* [8].

The *"remembering"* and *"expiration"* of packets is accomplished using caching techniques at the *packet* level. The model requires that every CCN-compatible router is equipped with a cache, which holds all packets for some amount of time; in [8] and in the present study, Least Recently Used (LRU) policies [10] are used, but alternative designs are also possible. If two or more subsequent requests *for the same content* arrive before the content expires *at any one of the routers/caches along the way*, the packet is forwarded again to the new interested user, instead of having to travel back to the content server to retrieve it. Clearly, this will benefit *popular content* (e.g., reduced delivery delay) and will reduce network resource requirements (e.g., bandwidth and server load).

In this study, we focus on the new flow model introduced in [8] and attempt to quantify the potential gains that the paradigm shift will bring with it. To achieve this goal and in order not to violate the semantics in [8], we study the caching part of the CCN paradigm isolated from the rest of the proposed architecture. We carry out a *packet-level* analysis, instead of a flow-level one, since it is our opinion that given the totally unexplored research field, investigations on packet-level dynamics should precede flow-wide conclusions. This study attempts to determine how long a given packet, which is referred to as the Packet of Interest, *PoI*, remains in *any of the caches* along the path from the user back to the server (i.e., not in a single cache), given a system topology and rates of requests. The contributions of the present study to this new field of research are as follows: an analytical model is developed for a single and then multiple caches; the validity of various modelling assumptions is discussed and a Monte-Carlo simulation is used to check some assumptions; a Java model is built based upon the analytical model; this is compared against model-simulator results from *ns-2*.

Our findings indicate that, as expected, there is a clear network-wide performance gain for popular content, but this gain: i) goes down to almost zero for unpopular content and this will be the more common case, ii) depends heavily on the size of the router-cache and iii) is different between routers distant from the original server of the data and those close to the server.

1.1 Macro- vs Micro-Caching

Strategic content replication techniques to reduce response time and the amount of network resources (e.g., bandwidth and server load) needed to send the content back to the user have been investigated in the form of i) Web Proxy Caching (e.g., [15] [3], [7]) and ii) Content Delivery Networks (CDNs) (e.g., [13]).

In both cases, the issue of utmost importance is the choice of the best possible network location to (either statically or dynamically) replicate content. In the case of Web-Proxy caches, the ISP is responsible for deciding where to replicate, while in the case of CDNs, this role is taken by the corresponding company that owns the network of surrogate servers. In both cases, however, *the location of the proxy/surrogate server is always known and fixed in advance.*

Furthermore, IP-multicast has been proposed and investigated (e.g., [14], [6]) to serve multiple users that are simultaneously interested in the same content. However, IP-multicast serves users that belong to the same group *only* and are prepared to receive all of the content that the *group* wants to receive (i.e., not necessarily the parts of it that individual users are interested in).

CCNs, as proposed in [8], constitute the conceptual marriage of the above technologies, *but in the "micro-level"*. We classify Web-Caching and CDNs as *"macro-caching"* approaches, since they target caching of entire objects, be it web-pages, or entire files; we consider CCNs as a *"micro-caching"* approach, since caching here is done at the packet-level[1]. Moreover, in the above technologies the setup is fixed and predefined. In contrast, *in CCNs content is cached and multicast "on-the-fly", wherever it is requested or is becoming popular.*

1.2 Single- vs. Multi-cache System Modelling

Web-caching and the properties of the LRU (and other similar) replacement policies have been extensively studied in the past, e.g., [1, 10, 9]. However, most of these studies focus either on single caches, e.g., [10], or chains of well-known and pre-defined caching points [3]. These studies can model many issues in detail, such as the correlation between subsequent requests, the request frequencies of the various packets in the system and the cache size [11, 10, 9].

Generally speaking, single-cache modelling has to consider mainly: i) the number of requests for a specific content *at a specific cache*, ii) the number of requests for other contents *at this specific cache*, iii) the size of the cache and iv) correlations between packet requests.

According to the authors in [8], *"CCN caches are the same as IP buffers, but have a different replacement policy"*. Trying to model multi-cache systems, where no hierarchy ([3]) exists is not trivial. The problem of modelling the caching system now comprises a multi-dimensional problem, where the above parameters still have to be considered, but they need to be considered *simultaneously for all the routers/caches along the path* from the content client to the content server.

For example, there is some pre-existing work on multiple cache situations. In [3], the authors develop models that apply to LRU caches which are two levels

[1] In fact, CCN [8] uses the term "message" to refer to both interests (request for data) and "content" (the data itself). Here we are specifically looking at the caching of *addressable* data messages. Whether multiple packets fit in one "message", or multiple "messages" fit in one packet does not affect much our modelling herein.

deep (that is, one cache feeds into a second cache). In [5] and [16] the authors develop a system which can approximate cache miss ratios for hierarchical cache systems using a solution which converges to a solution iteratively. However, the complexity for a single cache is $O(KB)$, where B is the buffer size and K is the number of independent items that have distinct access possibilities. There remains a need for a simple approximate model to estimate loss rates on a system when both B and K may be extremely large. This is where we focus on in this study. We propose a single-cache model, which is simple enough in order to remain valid when extended to multiple caches along the end-to-end path.

The model works from the point of view of estimating how long a particular content packet, the Packet of Interest, *PoI*, remains in *any of the caches* along the path. It outputs the proportion of time that PoI is nowhere cached, essentially reflecting the *cache miss ratio for PoI.*

2 Modelling CCN using Markov chains

2.1 A simple model for a single router

Assume that for a single router requests for the PoI arrive as a Poisson process with rate λ. Whenever such a request arrives, it moves this packet to the top slot in the cache. Assume that requests which will move the PoI further down the cache (either requests for packets not in cache or requests for packets in cache, but further down than the PoI) also arrive as a Poisson process with rate μ.

This process can be simply modelled as a continuous time homogeneous Markov chain, where the state of the chain represents the exact slot that the packet currently occupies in the cache. Number the Markov chain states as follows. State 1 is when the PoI is at the "top" of the cache (just requested), state N is when our packet is at the bottom of the cache (any request for a packet not already in cache will push our packet out of the cache). State $N+1$ represents the state where our packet is *not* in the cache. The chain and cache are shown in Fig. 1 (left).

All states (except state 1) have a transition to state 1 with rate λ (another request for the packet arrives and the packet moves to the top of the cache). All states i in $1 \leq i \leq N$ have a transition to state $i+1$ with rate μ (a packet arrives which moves our packet lower in the cache).

Now let $\pi = (\pi_1, \pi_2, \ldots, \pi_{N+1})$ be the equilibrium probabilities of the chain – these have the simply physical interpretation of being the proportion of the time the PoI spends at that cache position (or for π_{N+1} the proportion of the time the packet is not in cache). The chain can be trivially shown to be ergodic and hence, these probabilities exist for $\mu, \lambda > 0$ and finite N.

The balance equations can be easily solved to give $\pi_i = \frac{\lambda}{\mu}\left[\frac{\mu}{\mu+\lambda}\right]^i$ for $i < N+1$ and

$$\pi_{N+1} = \left[\frac{\mu}{\mu+\lambda}\right]^N . \tag{1}$$

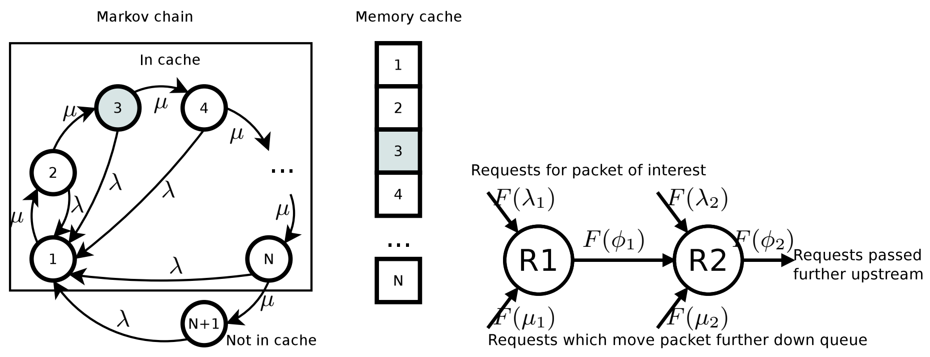


Fig. 1. Models for a single cache (left) and two caches (right)

This equation gives the proportion of the time that the *PoI* is not in cache (and also the proportion of requests for the *PoI* that get a cache miss). Naturally, the proportion of the time our packet gets a "cache hit" is one minus this. Therefore, the mean rate of requests that propagate upstream is: $\lambda\pi_{N+1} = \lambda\left[\frac{\mu}{\mu+\lambda}\right]^N$.

The next question is how can this be extended for the onward process when there are multiple caches in the system. The situation is shown in Fig. 1 (right). Router $R1$ has N_1 memory slots and $R2$ has N_2 memory slots. The notation $F(x)$ means an arrival process (not necessarily Poisson) with a rate x. The processes $F(\lambda_i)$ and $F(\mu_i)$ are Poisson by assumption. The remaining processes $F(\phi_i)$ are those requests for the packet of interest corresponding to cache misses in router Ri. These onward processes can be shown to have a phase distribution (see [12] for a general discussion of the phase distribution) derivable from a simple transform to the chain transition probabilities but it is not analytically tractable.

2.2 Modelling the multiple router system as a Markov chain

Consider again the diagram of two CCN routers in Fig. 1 (right). Note, however, that from (1) we have that the rate $\phi_1 = \lambda_1\gamma_1^{N_1}$, where $\gamma_1 = \mu_1/(\lambda_1 + \mu_1)$ – the rate of requests passed upstream from router $R1$ for the *PoI* is the rate of incoming requests times the proportion of time spent in the cache miss state. This trivially follows since the Poisson arrival process at rate λ_1 is memoryless, so a proportion $\gamma_1^{N_1}$ of them are passed upstream. The total arrival rate at router 2 is $\lambda_1\gamma_1^{N_1} + \lambda_2$, call this λ_2'.

Now the question is what proportion of the time is the packet in cache at $R2$. The proportion of time at $R1$ is still given by (1). The problem can again be formulated as a Markov chain, this time with states numbered in $N_1 \times N_2$, where the state number (i, j) refers to the position of the packet of interest in cache at $R1$ and $R2$, respectively. If $i = N_1 + 1$, then the packet is not in cache at router $R1$ and if $j = N_2 + 1$, then the packet is not in cache at router $R2$. Let $\pi_{(i,j)}$ be the equilibrium probability that the *PoI* is in state i at $R1$ and state j at $R2$. Let $\pi_{(i,\bullet)} = \sum_j \pi_{(i,j)}$ be the equilibrium probabilities of the states of

$R1$, independent of $R2$. It can be trivially shown that these are exactly those probabilities as for the single router model – that is, the presence of the second router does not affect the first as expected. Calculating $\pi_{(\bullet,j)} = \sum_i \pi_{(i,j)}$ has proved more difficult and the only result obtained so far is that

$$\pi_{(\bullet,1)} = \frac{\lambda_2' - \lambda_2 C/(\mu_2 + \lambda_2)}{\mu_2 + \lambda_2' - C},$$

where λ_2' is the adjusted arrival rate discussed earlier and

$$C = \lambda_1 \left[(\mu_1/(\lambda_1 + \mu_1))^{N_1} - (\mu_1/(\lambda_1 + \lambda_2 + \mu_1 + \mu_2))^{N_1} \right].$$

From this we can see that C is positive and that $\pi_{(\bullet,1)}$ will tend to $\pi_{(\bullet,1)} = \lambda_2'/(\mu_2 + \lambda_2')$ as N_1 increases. This is the equation obtained when the non Poisson nature of the arrival stream is ignored. It can be easily seen that C tends to zero when N_1 increases as both terms inside the brackets are less than one and raised to a positive power.

For the other states, the Markov chain can be explicitly simulated (Monte-Carlo simulation) and the proportion of time spent in each of the states explicitly measured and summed to produce an estimate of the true equilibrium probabilities. One million iterations are performed with results from the first half a million iterations being discarded as "warm up". To ensure convergence, small values for N_1 and N_2 are used. The first simulation was with $N_1 = 6$, $N_2 = 4$ and $\lambda_1 = \lambda_2 = \mu_1 = \mu_2 = 1.0$. The results for $\pi_{(\bullet,j)}$ were in agreement with the Poisson approximation to within ± 0.002. A more rigorous test is provided when the value ϕ_1 is high compared to λ_2. Again with $N_1 = 6$ and $N_2 = 4$ the theory is tested against experiment, but this time with $\lambda_1 = 100.0$, $\mu_1 = 10,000$, $\lambda_2 = 0.1$ and $\mu_2 = 100.0$. For these parameters, the agreement with the Poisson assumption was weaker but all probabilities agreed within ± 0.008. This was the highest level of disagreement found with any of many sets of test parameters tried. We also note that these tests are with N_1 very low compared with the figures which will be used in real runs (small numbers of states were used because Markov chains with fewer states converge better in Monte Carlo simulation).

This section, therefore, provides good evidence that a pair of CCN caches feeding from one to another can be reasonably approximated by the simple single cache model presented before. The low error rate in assuming that the misses from one cache can be used as the input to a second (and the violation of the Poisson assumption ignored) leads us to our final multiple router model. In this model, the miss rates for the *PoI* are calculated by assuming that the input rate to a router is a combination of the rates of requests made directly at that cache and those cache misses passed upstream from previous routers.

2.3 Investigating the μ process further

In this section we investigate further the nature of the process μ. It is a subtle but very important point that the *PoI* can be moved down the cache by two related processes. The first possibility is that a packet arrives which is not in cache.

Assume requests for all other packets (cached or uncached) arrive at rate ν and that these arrivals are Poisson. However, not all arrivals of the rate ν will move the *PoI* further down the cache. Split ν into ν_c and ν_n for those requests, which are in cache and not in cache respectively – so the rate will sum to $\nu = \nu_c + \nu_n$. Arrivals in ν_n will always move the *PoI* further down (or out of) the cache, if it is in the cache. Arrivals in ν_c will only move the packet further down the cache, if they are requests for a packet which is at a lower cache position. For example, if the *PoI* is at cache position n, a request for any packets in positions $1, \ldots, n-1$ will not move the position of the *PoI* in the cache, but will simply reshuffle those packets above it.

The next step is to assume that packets are equally likely to be found in any position in the cache. This is an unlikely assumption in real life but allows us to calculate an upper bound for the *PoI* cache miss rate. The reason this is an upper bound will be discussed later.

With these assumptions the position of the *PoI* can again be calculated as a Markov chain, but this time, the rate of moving from a state j to a state $j+1$ is given by $\nu_n + \nu_c(N+1-j)/N$, where the first part represents the arrival of a request for a packet not in the cache and the second part represents the arrival of request for a packet which is in the cache, in a position lower than the *PoI* (currently in position j) – impossible if the packet is in position N and always the case for a request for a packet in the cache, if the packet is in position 1. The derivation of equilibrium probabilities π_j can proceed in a similar manner to the previous section, however, while λ remains the same, the μ is replaced by $\nu_n + \nu_c(N+1-j)/N$.

With these assumptions then $\pi_{N+1} = \frac{\Gamma[1+N(\nu_c+\nu_n+\lambda)/\nu_c]}{\Gamma[N+N(\nu_c+\nu_n+\lambda)/\nu_c]}$, where Γ is Euler's gamma function. Now, using the well-known formula $\frac{\Gamma[z+a]}{\Gamma[z+b]} = z^{a-b}[1+O(z^{-1})]$ then

$$\pi_{N+1} = \left[\frac{\nu_c+\nu_n}{\nu_c+\nu_n+\lambda}\right]^N \left[1+O\left(\frac{\nu_c}{N(\nu_c+\nu_n)}\right)\right] \left[1+O\left(\frac{\nu_c}{N(\nu_c+\nu_n+\lambda)}\right)\right]. \quad (2)$$

The order terms become small as N becomes large, or as ν_c becomes small with respect to ν_n. It should also be noted that the time spent out of cache must be greater than if only the ν_n arrivals were considered. Therefore, the proportion of time the *PoI* is not in cache is $\pi_{N+1} \simeq \left[\frac{\nu_c+\nu_n}{\nu_c+\nu_n+\lambda}\right]^N$ when $\nu_n \gg \nu_c$ (only a small proportion of requests are in cache). In those circumstances, the fact that the μ process in Section 2.1 is a good approximation for the model that takes more careful account of whether arriving packets are in cache, or not and the approximation $\mu = \nu_n + \nu_c$ can be used. In other words, the rate μ is equal to the rate of requests for all packets other than the packet of interest (which is requested at rate λ).

The question remains how (2) alters if the assumption is dropped that arriving non PoI packets are equally likely to be at any position within the cache.

This assumption is true only when the non-PoI packets are requested with equal frequency. For non-equal distributions then the term $\nu_n + \nu_c(N+1-j)/N$ would always be larger for lower numbered states and π_{N+1} is hence smaller than (2). No lower bound for π_{N+1} exists (indeed π_{N+1} can be arbitrarily close to zero for a given ν_c, ν_n, N and λ). Therefore, (2) is an upper bound which occurs in the case when non-PoI packets are requested with equal frequency and the lower bound is zero. It is worth remembering, however, that the definition of the μ process would, in that case, give the μ process an arrival rate near zero (it is explicitly defined as the arrival rate of packets which move the *PoI* further down the cache).

2.4 Summary of mathematical results

The reality of the Poisson assumptions may be questioned. It is well known for example, that Internet traffic is long-range dependent (for a recent review see [4]). However, that is the traffic process as a whole, not the process of requests, which can look very different. Also, it may be argued that nobody knows what traffic statistics in a CCN-based Internet would look like. Many request processes (for example www, ftp and smtp) today can, in fact, be modelled perfectly well with Poisson processes (e.g., Table 3 in [2]).

Section 2.1 showed a simple closed form solution when requests for the *PoI* are Poisson with rate λ and requests which move the *PoI* further down the cache are Poisson with rate μ. In Section 2.2, it was shown experimentally that certain deviations from Poisson for the requests for the *PoI* make only small differences to the proportion of time spent in cache. In Section 2.3 the assumption of Poisson was relaxed slightly for the μ process by considering requests for packets in cache and requests for packets not in cache separately, however, it was necessary to make strong assumptions about the distribution of packets within the cache. Further work in this direction would require an explicit model of the request frequencies (heavy tailled models would be a good candidate). This would bring the simple model presented here closer to those detailed single cache models discussed in previous sections.

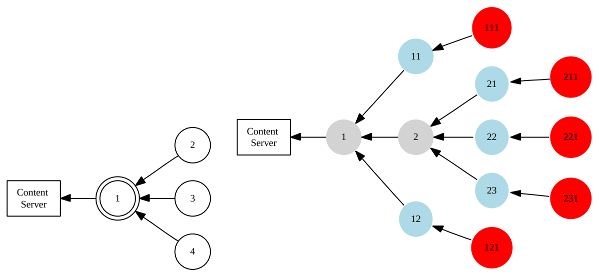
Finally, therefore, the single cache model of Section 2.1 can be used to model a tree of caches by assuming that the misses from caches propagate towards the original holder of the content and that requests for the *PoI* arriving at a cache are a Poisson process with a total rate which is the sum of the rate of requests directly to that cache and the rate of missed requests to all caches which directly pass on to this cache. This situation is directly amenable to simple simulation even for situations with many caches and large router memories.

3 Simulation Results

3.1 Scenario Setup and Parameter Settings

Two simulation models are evaluated. The first directly encodes the analytical model from Section 2, while the second model is developed on *ns-2* and is presented in Section 3.4.

Our first model takes as input the caching tree topology and, for each cache, the values of λ (request rate for *PoI*), μ (rate at which other requests move the *PoI* down the cache) and N (cache size in packets). For simplicity, results are presented in terms of the ratio λ/μ, referred to as the *Content Popularity Ratio* (R_{CP}). The larger this measure is, the longer the *PoI* will stay in cache. For each router the simulation outputs the *Proportion of Time Not in the Cache* for the *PoI* from Eq. (1) and the proportion of requests that have to be forwarded further on towards the content server, referred to as λ_{Output} and given in Section 2.1. The two topologies tested are given by Fig. 2; here, we are interested in scenarios where end-users request content from professional content-providers, or data-centers, i.e., we are not interested in peer-to-peer scenarios.



(a) Topology I (b) Topology II

Fig. 2. Simulation Topologies: Simple and Extended Tree

It is reasonable to assume that routers which get more requests have more cache and therefore reasonable to assume that μ is proportional to N. For simplicity, the constant of proportionality is set to 1 for most results presented here, unless explicitly stated differently. This reduces the number of parameters to test in the system.

3.2 Scenario 1: Content Popularity and Cache Size

We initially assess the properties of requested content with regard to its popularity, as CCNs were originally proposed to deal with popular content and *flash crowds* [8]. This scenario uses the topology of Fig. 2(a) and experiments with different values for R_{CP}. Values for R_{CP} range from 0.000125 to 0.01 – all of these values represent very popular content but the experiment shows how the popularity affects the time in cache. The buffer size N is set to 200 packets.

Fig. 3 presents the proportion of time the *PoI* is not in any of the caches along the path. Two conclusions can be drawn from this scenario. First, unpopular content spends very little time in the cache. This time is comparable to the life-cycle of an IP packet in the forwarding buffer in the prevailing networking model. This would be the case for most packets. Second, there is a clear difference between caching times for popular content in the core and leaf routers. Caching at the leaf nodes can reverse the expected effect on popular content nearer the server with the data source. That is, more caching at the leaf leads to less caching

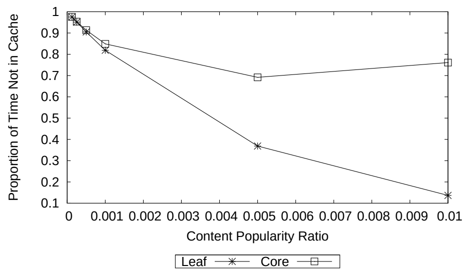
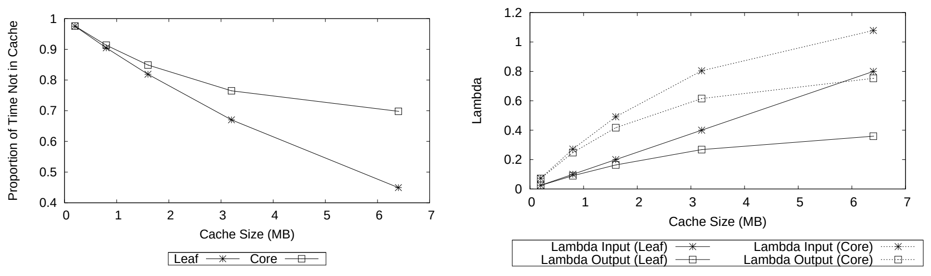


Fig. 3. Scenario 1, Topology I - Increasing Content Popularity

nearer to the data source for very popular content. (Note that the requesters and the data sources are all likely to be "edge" nodes in traditional Internet terms).

We go one step further to investigate the cache-size properties of the CCN paradigm. The buffers in small IP routers can serve the purpose of implementing CCNs, but the gain against today's end-to-end model will be marginal (i.e., even popular content is going to be "forgotten" very quickly). Hence, bigger amounts of memory may have to be considered in order to obtain gains from the paradigm shift. Using the same settings as the previous simulation, and keeping $\mu = N$, the cache size N is varied from 100 to 64,000 packets (rounding packet sizes to 1KB, this is 100KB to 6.4MB).



(a) Increasing Cache Sizes (b) Input and Output Lambdas

Fig. 4. Scenario 1, Topology I - Cache Size

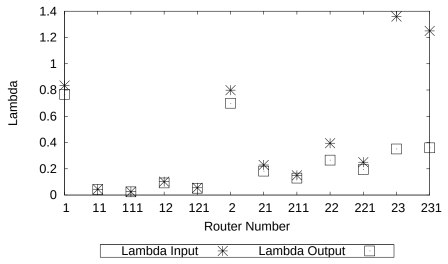
In Fig. 4(a), we see that for larger caches, even the less popular content has the chance to stay cached for longer times, as would be expected. Above 6MB of cache, even the less popular content is in cache at the leaf node for more than 50% of the time. Again, caching at leaf nodes has reduced the proportion of time spent in cache nearer to the content source.

Fig. 4(b) shows the request rates for the PoI at each router (λ_{Input}) as well as those requests passed upstream towards the content server (λ_{Output}). The results agree with Fig. 4(a) and confirm that leaf nodes are passing less traffic upstream when caching levels are higher. This causes this specific content to become less popular nearer the packet source (see Fig. 4(b)).

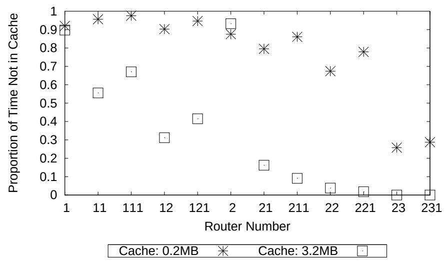
3.3 Scenario 2: More Complex Tree Topologies

Simulation is now carried out on the *extended tree* topology of Fig. 2(b). In this scenario we apply different λs on each of the leaf routers. The purpose is to evaluate heterogeneity in requests. The request ratios for *PoI*, λ_{Input}, together with the corresponding output λs for each one of the participating routers, are shown in Fig. 5(a). Fig. 5(b) shows the proportion of time that the *PoI* is not in the cache for each of the caches of Fig. 2(b). Results are given for two different cache sizes (0.2MBs and 3.2MBs). The R_{CP} for the content of interest ranges between 0.0001 and 0.005. We observe the following:

1. *Even extremely popular content is "forgotten" quickly by routers near the data source.* Given a tree topology, similar to the one in Fig. 2(b) (a loose reflection of the topology of the Internet), these servers nearer to the data source receive many requests, much of which is for unpopular content. Given that many requests have to be forwarded up the chain to the content server – see Fig. 5(a), routers $R1$ and $R2$) – even popular content gets forgotten quickly – see Fig. 5(b) routers $R1$ and $R2$.
2. *Leaf routers "remember" popular content for longer time.* In Fig. 5(a), we see that $\lambda_{Input} \simeq 0.4$ for router $R22$, while it is more than 0.8 for core router $R2$. Although both routers forward upwards a fairly big percentage of these requests, in Fig. 5(b) we see that the proportion of time not in cache for router $R22$ still drops lower than for core router $R2$.
3. *Larger cache sizes exaggerate point number 2 above, while leave point 1 untouched.* In Fig. 5(b), we see that the proportion of time the *PoI* spends in leaf caches increases with the cache size. The same is not true for routers nearer the source as can be seen in the same Fig. for routers $R1$ and $R2$.



(a) Input and Output Lambdas



(b) Caching Time for Different Cache Sizes

Fig. 5. Scenario 2, Topology II

3.4 Scenario 3: Model Simulator vs *ns-2* Simulations

The basic functionality of caching using the CCN paradigm [8] is implemented in *ns-2*. Close approximations to the scenarios from the previous section were tested to estimate the agreement between theoretical modelling and *ns-2* simulation. The full CCN paradigm as stated in [8] cannot easily be implemented in *ns-2*. Some of the reasons why are listed below.

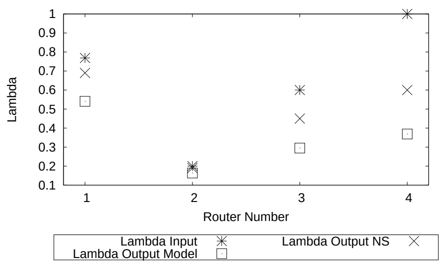
– *CCN Router.* The structure of the CCN router influences several parts of the networking stack. Several designs are possible. It is not clear yet, for example, how traffic from different flows is multiplexed before it enters the outgoing link.
– *Cache Size.* In case of small caches, similar to the IP buffers we have today, the network-wide gain will be limited. On the other hand, larger caches will increase the routers' computational burden and will complicate the collaboration between outgoing interfaces and the cache.
– *Transport Protocol.* The CCN router structure will influence massively the design of the CCN transport entity. The initial proposal focuses on a *receiver-driven* transport protocol [8]. Therefore, flow rate adjustments and retransmission timers will have to be implemented on the receiver side. This constitutes a role- and functionality-swap from the current situation. This will heavily influence simulations and results at the flow level.

The experiments here attempt to minimise the above effects. The topology used is that of Fig. 2(a); the setup includes a simple constant bit rate (CBR) application over UDP that sends requests for 1KB-packet responses. By using UDP and CBR the issues of implementing a CCN-friendly transport protocol (which is yet to be defined) are avoided. Requests arrive as Poisson distributions to the leaf routers $R2$, $R3$ and $R4$. The rate of request for the *PoI* over each leaf node can be set and normalised against the total number of requests. This is similar to the R_{CP} defined in the previous section. However, the exact implementation of μ is hard to achieve in a simulation environment – the effect of the *PoI* being pushed further down the cache is achieved by requesting packets other than the *PoI*, but here, the rate of requests equal to a given value of μ for a given router will be a function of N. The cache size is a fraction of the total number of packets in our pool; this fraction was initially set to 0.5. The simulation time was long enough to guarantee statistically stable results[2]. Simulations were carried out for different values of the cache-size fraction with regard to the number of packets in our pool. Although these results are not presented here, due to space limitations, we report that they follow similar trends.

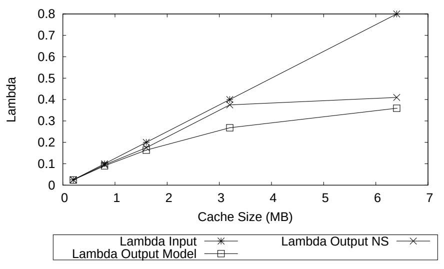
Initially, $R_{CP} = 0.001$ (for $R2$), $R_{CP} = 0.003$ (for $R3$) and $R_{CP} = 0.01$ (for $R4$) and $N = 200$. Arrivals for other packets are set to a rate of 200 per unit time, to approximate $\mu = 200$. The *ns-2* simulation outputs λ_{Input} and λ_{Output} as in the previous section.

Fig. 6(a) shows λ_{Input} and λ_{Output} for each one of the routers involved. It can be seen that simulation-routers tend to *"forget"* content more readily than model-routers (i.e., NS λ_{Output} is larger than the Model one). Indeed, even for popular content (e.g., router $R4$), the simulation result shows "forgetfulness" of around 30% more than the model result. Differences between the simulator and the model were expected, because of the difficulties of tuning μ. However, it is encouraging the fact that both show results, which are broadly speaking, very similar. The curve shape for both simulations for λ_{Output} follows the same trend.

[2] This time varies depending on the R_{CP} and the cache size.



(a) Input and Output Lambda



(b) Increasing Cache Size

Fig. 6. Scenario 3: Model Simulator vs $ns - 2$

In Fig. 6(b), we increase the cache-size and observe the effect on λ_{Output} for the NS and the model simulator, respectively. For small cache sizes, the results are largely in agreement, although there are small differences that cannot be seen here due to the Fig. scale. These differences do not exceed the threshold of 10%. As cache size increases to 3.2MB, the *ns-2* λ_{Output} follows the model-simulator input, instead of the model-simulator output (i.e., λ_{Output}). The results are closer to the model-simulator when the cache size increases to 6.4MBs. The disagreement between results in the 3.2MB case could again be attributed to difficulties tuning the μ parameter. In the 6.4MB case, there is better agreement between the results, due to the fact that there are very few cache misses for the *PoI* (i.e., the proportion of time not in the cache for *PoI* is very small, as we have also shown in Fig. 4(a)).

While the results from the two simulators are not directly comparable it is clear they are giving the same indications about the systems being studied.

4 Conclusions

This paper approaches the unexplored area of *Content-Centric Networking*, proposed in [8] from the viewpoint of a *packet*, or *message* (i.e., not a flow) and its corresponding caching dynamics. The approach involved a Markov-chain analysis, followed by model and *ns-2* simulations. To the best of our knowledge, this is the first study to investigate this new networking paradigm.

The analytical model presented for the caching system presents a simple and tractable model for the time that a packet spends in cache. While the model began with strong assumptions of Poisson behaviour, subsequent analysis shows that many of these assumptions can be weakened and the analytical model remains valid. By necessity, the *ns-2* implementation made some simplifying assumptions as the detailed protocol of a CCN network is as yet unknown. Results from the model and the *ns-2* simulation were in broad agreement, but with some differences due to the different nature of the approaches.

In summary, the simulation findings indicate that: i) *popular content tends to be cached at the leafs of the network*. Content servers may receive a large number of requests for different content and hence forget more easily than leaf routers. ii) *Sizing CCN caches is not a trivial task*, since it depends on the distribution of arrival rates and the flows of requests from upstream. Smaller cache sizes may bring very little gain as only extremely popular content will spend significant time in cache.

Much future work remains to be done. The analytical model can be further developed and in particular the nature of the μ parameter needs to be refined to enable direct and fair comparison with simulation models. More features have to be integrated in both the mathematical model and the simulators when mechanisms for the transport layer are clarified in more detail and the router design is further elaborated. This will allow a rich vein of future investigation for this new networking research field.

References

1. Breslau, L., et al.: Web caching and zipf-like distributions: Evidence and implications. In: In INFOCOM. pp. 126–134 (1999)
2. Cairano-Gilfedder, C.D., Clegg, R.G.: A decade of internet research: Advances in models and practices. BT Technology Journal 23(4), 115–128 (2005)
3. Che, H., Wang, Z., Tung, Y.: Analysis and Design of Hierarchical Web Caching Systems. In: INFOCOM. pp. 1416–1424. IEEE (2001)
4. Clegg, R.G., Cairano-Gilfedder, C.D., Zhou, S.: A critical look at power law modelling of the Internet. Computer Communications 33(3), 259–268 (2009)
5. Dan, A., Towsley, D.: An approximate analysis of the lru and fifo buffer replacement schemes. pp. 143–152. SIGMETRICS '90 (1990)
6. Floyd, S., et al.: A reliable multicast framework for light-weight sessions and application level framing. IEEE/ACM Trans. Netw. 5, 784–803 (December 1997), `http://dx.doi.org/10.1109/90.650139`
7. Fujita, N., Ishikawa, Y., Iwata, A., Izmailov, R.: Coarse-grain replica management strategies for dynamic replication of web contents. Comput. Netw. 45(1) (2004)
8. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking Named Content. In: CoNEXT '09. pp. 1–12. ACM, New York, NY, USA (2009)
9. Jelenkovic, P.R.: Asymptotic approximation of the move-to-front search cost distribution and least-recently-used caching fault probabilities. The Annals of Applied Probability 9(2) (1999)
10. Jelenković, P.R., Radovanović, A.: Least-recently-used caching with dependent requests. Theor. Comput. Sci. 326, 293–327 (October 2004)
11. Jelenković, P.R., Radovanović, A., Squillante, M.S.: Critical sizing of lru caches with dependent requests. Journal of Applied Probability 43(4), 1013–1027 (2006)
12. Neuts, M.F.: Matrix-Geometric Solutions in Stochastic Models: an Algorthmic Approach Chapter 2: Probability Distributions of Phase Type. Dover Publications Inc. (1981)
13. Pallis, G., Vakali, A.: Insight and perspectives for content delivery networks. Commun. ACM 49(1), 101–106 (2006)
14. Ratnasamy, S., Ermolinskiy, A., Shenker, S.: Revisiting IP multicast. In: SIGCOMM '06. pp. 15–26. ACM, New York, NY, USA (2006)
15. Rosensweig, E.J., Kurose, J.: Breadcrumbs: Efficient, Best-Effort Content Location in Cache Networks. In: INFOCOM. pp. 2631–2635 (2009)
16. Rosensweig, E.J., Kurose, J., Towsley, D.: Approximate models for general cache networks. In: INFOCOM. IEEE (2010)