Performance and cost effectiveness of caching in mobile access networks

Salah Eddine Elayoubi[*]
Orange Labs, Issy-Les-Moulineaux, France
salaheddine.elayoubi@orange.com

James Roberts
IRT-SystemX, Paris-Saclay, France
james.roberts@irt-systemx.fr

ABSTRACT

The paper considers the use of caching in mobile access networks and seeks to evaluate the optimal memory for bandwidth tradeoff at base station (BS), packet gateway (PGW) and a possible intermediate mobile cloud node (MCN). Formulas are derived for the hit rate under time varying popularity and for a novel cache insertion policy incorporating a pre-filter. The analytical model is applied first to demonstrate that reactive caching is not efficient for nodes with low demand due to the negative impact of content churn. This means BS or MCN caches must be managed proactively with popular content items pre-fetched under some centralized control. Quantifying the tradeoff at each level leads us to conclude that limited caching at BS and MCN levels brings significant savings while to store the vast majority of the content catalogue at the Internet edge at the PGW is clearly cost effective.

Categories and Subject Descriptors

C.2.1 [**Computer communication networks**]: Network Architecture and Design—*packet-switching networks*; C.4 [**Performance of systems**]: [modeling techniques, performance attributes]

General Terms

Algorithms, Performance

Keywords

Cache hit rate; memory-bandwidth tradeoff; mobile access network; time varying popularity

1. INTRODUCTION

Lively debate on the optimal use of caching in information centric networks is still ongoing. A major outstanding issue is whether caching should be limited to the edge of the Internet or whether the network should be designed to make optimal use of content stores incorporated in routers distributed throughout the core. Based on our own prior work, as well as arguments published in the literature, we tend to believe edge caching is more cost effective and has potential to significantly simplify the network architecture. If this belief is correct, it still remains to more precisely understand where and how caching should be performed within the access infrastructure beyond the edge. Our objective in this paper is to evaluate the performance and cost effectiveness of caching solutions in a mobile access network.

We consider the network depicted in Figure 1 where base stations (BS) are connected to a packet gateway (PGW) at the Internet edge via a so-called mobile cloud node (MCN) and all three levels might host a cache. In current 4G networks the MCN is absent and BSs are connected directly to the PGW via IP tunnels. The MCN is envisaged for 5G mobile access and is intended to fulfill diverse functions including radio link control and offloading of mobile device applications. We suppose the MCN would also perform caching for and in place of its connected BSs. The number of BSs connected to a PGW is around 1000 while the MCN would serve up to 100 BSs.

We consider the placement of caches from the point of view of a network operator seeking to minimize investment by realizing an optimal tradeoff of cache memory for network bandwidth. We assume ongoing tussles between operators and content providers have been resolved so that all stored content can indeed be cached. This implies the network offers content providers the necessary control over content delivery, for billing, accounting or ad placement, say, so that there is no impediment to delivering a content item from a remote cache if available.

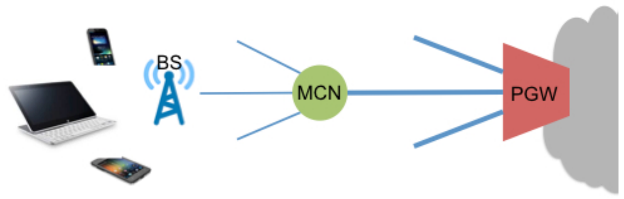

Figure 1: Mobile access network: connects base stations (BS) to Internet via packet gateway (PGW) and optional mobile cloud node (MCN).

[*]Both authors are members of the LINCS, Paris, France. See www.lincs.fr.

ICN'15, September 30–October 2, 2015, San Francisco, CA, USA.
© 2015 ACM. ISBN 978-1-4503-3855-4/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2810156.2810168.

Cache placement has previously been considered as an optimization problem where a fixed volume of memory, or "cache budget", must be distributed over a given network topology, typically in order to minimize the average download path length. Fayazbakhsh *et al.* [5] concluded in this way that edge caching was preferable while Rossi and Rossini [12] later showed that the conclusions of [5] would have changed had the authors considered more efficient cache insertion and request forwarding policies. Dabirmoghaddam *et al.* [3] showed that the notion of cache budget is in fact flawed since, obviously, edge caching becomes progressively more attractive as the budget increases while there is no obvious way to determine what the budget should be. In evaluating the memory for bandwidth tradeoff, we do not have to postulate a cache budget but rather to directly weigh the costs of caching against the resulting savings in network transport infrastructure.

At any given network node, the optimal cache size will be that which minimizes, $\text{cost}_b(D)+\text{cost}_m(C)$, where D is busy period demand upstream of the cache in bit/s, C is cache size and cost_b and cost_m are cost functions for bandwidth and caching, respectively. Demand D determines required infrastructure investment. It is proportional to the cache miss rate so that the tradeoff evaluation critically depends on this or, equivalently, on the hit rate $h(C)$. Much recent research has clarified cache hit rate performance under stationary demand, i.e., when the catalogue of content items is fixed and their individual popularities do not change. However, at points low in the access network, like the BS or MCN, where demand is relatively light, it is essential to account for time varying popularity as content churn then becomes the major source of cache misses.

We evaluate tradeoffs assuming cache performance is *ideal* in the sense that it stores the currently most popular items that fit into the cache. It is known that this ideal can be attained under stationary demand by applying a highly selective insertion policy where items are only added to the cache if they have been determined, by a history of previous requests, to have high popularity. The response to changing popularities of such policies is slow, however, and tends to further degrade cache performance in nodes with low demand. When reactive caching fails, it is necessary to envisage proactive caching. This means some external entity determines popularities in real time and ensures remote caches contain the most popular items by pre-fetching. Pre-fetching requires an appropriate architectural solution allowing the entity in question to infer popularities, typically by monitoring demand from a large user base.

Our first contribution in Section 2 is to derive accurate analytical hit rate approximations for an LRU cache applying a novel selective insertion policy. Hit rates can be evaluated under stationary or time varying popularity for any cache size. These formulas are then used in Section 3 to determine whether reactive or proactive caching should be employed at each of the considered network levels. We discuss the architectural implications of the need to perform pre-fetching when reactive caching is not viable. Lastly, in Section 4, we evaluate the cost tradeoff. We assume the BS would have a fixed-size cache of limited complexity while the PGW and MCN have a modular design, like a data center, allowing a wide range of cache sizes.

N	catalogue size
q_i	popularity of item i, $\sum_{1\le i\le N} q_i = 1$
α	Zipf law exponent
C	cache size
$c = C/N$	normalized cache size
K	pre-filter size
t_C	cache characteristic time
h	overall hit rate
h_i	hit rate for item i
$h_i^{(n)}$	hit rate of n^{th} requests for item i
τ_i	mean lifetime of item i
p_{in}	proba. n requests in item i lifetime
η_i	proba. current request for item i is not last
l_k	mean lifetime of items in class k
f_k	fraction of items in class k
σ	mean requests per item per day
k_b	monthly per Mb/s bandwidth cost
k_m	monthly per GB memory cost
Δ	overall cost
$\delta(c)$	normalized cost
Γ	ratio of zero cache cost to full cache cost

Table 1: Summary of notation..

2. CACHE HIT RATES

We first consider cache performance under stationary demand, applying the independent reference model, before evaluating the impact of changes in popularity over time. Notation used in the following is summarized in Table 1.

2.1 Stationary demand

Assume the content catalogue contains N items of equal size and that demand for item i is proportional to q_i independently of time. This stationarity assumption is reasonable if the q_i are popularities in a relatively short period like 1 day. We normalize the q_i such that $\sum_1^N q_i = 1$. The "items" might typically be chunks of videos or other data objects and are assumed to have the same size.

An ideal cache.

An ideal cache of size C would store just the C most popular items. The hit rate $h(C)$ is then simply the ratio $\sum_1^C q_i / \sum_1^N q_i$. For Zipf($\alpha$) popularity, $q_i \propto 1/i^\alpha$, with $\alpha < 1$ and N large, we have $h(C) \approx (C/N)^{1-\alpha}$.

Ideal cache performance can be realized if item popularities are accurately determined (e.g., by a server aware of demand from a large population of users and able to anticipate changes in popularity) and the cache content is proactively updated by pre-fetching. An alternative is to augment a reactive caching policy like LRU with a pre-filter, as discussed below.

LRU cache.

The hit rate of an LRU cache can be accurately evaluated by the Che approximation [6]. We have $h(C) = \sum_1^N q_i(1 - e^{-q_i t_C})$ where the *characteristic time* t_C satisfies $C = \sum_i (1 - e^{-q_i t_C})$. The characteristic time is the time to receive requests for C distinct items. Figure 2 shows that the performance of LRU can be far from ideal, especially for a cache that has capacity for only a small fraction of the catalogue.

Cache with pre-filter.

Several authors have observed that the performance of an LRU cache can be improved by equipping it with what we term a pre-filter. We illustrate this through the following particular filter design. The identities of the last K items to be requested are recorded in a filter list. If a request is a hit (the item is present in the LRU cache), it is moved to the front of the LRU as usual. If the item is not in the LRU but is in the filter list, it is fetched and placed at the front of the LRU displacing the item at the end. If the item is not in either the LRU or the filter list, it is fetched but not cached.

To evaluate the hit rate, we adapt an approximation proposed by Martina *et al.* [9] for another type of pre-filter. We assume the states of the pre-filter and the LRU are statistically independent and consider $h_i^{(n)}$, the probability the n^{th} request for item i is a hit. Conditioning on the fate of the n^{th} request, we can write,

$$h_i^{(n+1)} = \left(1 - e^{-q_i t_C}\right) \left(h_i^{(n)} + (1 - h_i^{(n)})(1 - (1 - q_i)^K)\right), \tag{1}$$

where t_C is a characteristic time, to be determined. The first term, $(1 - e^{-q_i t_C})$, is the probability the $(n+1)^{th}$ request arrives within t_C of the n^{th} while the second term is the probability i was already in the LRU following the n^{th} request. Under the present stationarity assumption, we have $h_i^{(n+1)} = h_i^{(n)} = h_i$ and, from (1),

$$h_i = \frac{(1 - e^{-q_i t_C})(1 - (1 - q_i)^K)}{1 - (1 - e^{-q_i t_C})(1 - q_i)^K}.$$

Equating $\sum_1^N h_i$ to C, as in the Che approximation for LRU, gives the equation for t_C and consequently all the individual hit rates. The overall hit rate is $h(C) = \sum_1^N q_i h_i$.

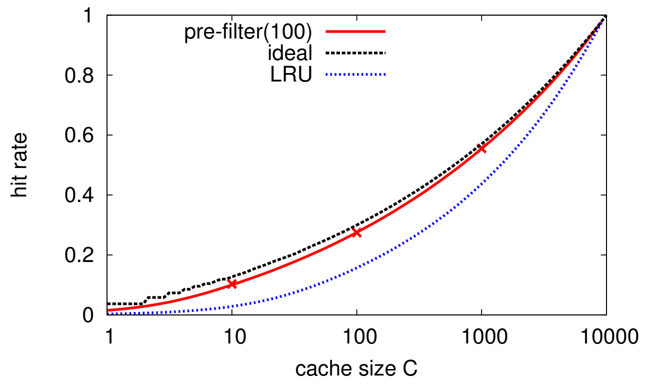

Figure 2: Hit rates with pre-filter of 100 places against cache size: $N = 10^4$ items with Zipf(.8) popularity; crosses are simulation results.

Figure 2 shows the hit rate obtained with a pre-filter of 100 places in comparison to ideal and LRU caching. Crosses are hit rates derived by simulation and illustrate the high accuracy of the approximation based on the independence assumption. The act of filtering preferentially selects the most popular items for insertion so that the cache has close to ideal performance. Performance could be further improved by reducing the size of the filter but this also reduces the reactivity of the cache to changing popularities, as discussed next.

2.2 Time varying popularities

We propose a simple model, following Wolman *et al.* [15], to account for the fact that content items have finite lifetimes. We suppose that the catalogue remains fixed at N and that at any instant there is an item i with popularity q_i for $1 \le i \le N$. However, the item labelled i can change between requests as the current item ends its lifetime and a new one begins. This modelling device allows us to retain the notion of popularity law while accounting for the fact that content churn can significantly reduce the hit rate.

LRU cache.

Let the mean lifetime of items labelled i be τ_i. The rate of misses for such items in an unlimited capacity cache is then $1/(1 + q_i\tau_i)$ (i.e., 1 in an average number of requests of $1 + q_i\tau_i$). For a cache of capacity $C < N$, the first request for a new item i is necessarily a miss and the LRU miss probability for any request after the first is $e^{-q_i t_C}$, as for the IRM, where t_C is the Che characteristic time. The overall hit rate is thus

$$h_i = \left(1 - \frac{1}{1 + q_i\tau_i}\right)\left(1 - e^{-q_i t_C}\right). \tag{2}$$

To compute t_C, note that the cache may contain more than one version of item i. To simplify we exclude this possibility by assuming the old version is overwritten by the new. We then still have the identity $C = \sum h_i$ to compute t_C. However, simulations used to validate the approximation do maintain copies of old versions until they are ejected by the LRU policy.

Note that, in the present case, the Che equation does not necessarily have a solution when the τ_i are small. This is because content churn is too rapid for their ever to be a finite interval in which there are requests for C or more distinct items. In this case we must set t_C to ∞ in (2).

Cache with pre-filter.

Formula (1) constitutes a recursion for calculating the hit rate of the n^{th} request for item i after a renewal, for $n \ge 3$ with $h_i^{(1)} = h_i^{(2)} = 0$.The mean hit rate is

$$h_i = \frac{h_i^{(1)} + h_i^{(2)}(1 - p_{i1}) + h_i^{(3)}(1 - p_{i1} - p_{i2}) + \dots}{1 + (1 - p_{i1}) + (1 - p_{i1} - p_{i2}) + \dots}$$

where p_{in} is the probability the number of requests between renewals is n, for $n \ge 1$.

To proceed, we assume the number of requests has a geometric distribution, $p_{in} = (1 - \eta_i)\eta_i^{n-1}$ with $\eta_i = q_i\tau_i/(1 + q_i\tau_i)$. The expression for h_i then simplifies to $h_i = (1 - \eta_i)\sum_n \eta_i^{n-1} h_i^{(n)}$. Multiplying both sides of (1) by η_i^n and summing leads eventually to an equation for h_i with solution,

$$h_i = \frac{\eta_i^2\left(1 - e^{-q_i t_C}\right)\left(1 - (1 - q_i)^K\right)}{1 - \eta_i\left(1 - e^{-q_i t_C}\right)(1 - q_i)^K}.$$

As for the LRU cache, we determine t_C from the equation $\sum_1^N h_i = C$, letting $t_C \to \infty$ when there is no solution.

Lifetimes.

Traverso *et al.* have derived some lifetime statistics from trace data [14, Table 2]. The authors give the fractions of items having a lifetime in five intervals: 0 to 2 days, 2 to 5, 5 to 8, 8 to 13, more than 13. They also observe that

class k	interval	l_k	f_k
1	0-2	1.1	.005
2	2-5	3.3	.008
3	5-8	6.4	.005
4	8-13	10.6	.008
5	> 13	365	.974

Table 2: Lifetime in days of most popular items, deduced from Traverso *et al.* : l_k is the class mean lifetime and f_k the fraction of items in the class.

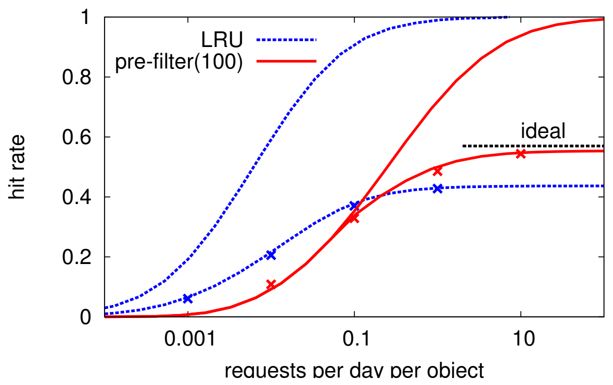

Figure 3: Hit rate against normalized demand σ: analytical results for a cache of size 1000 and an unlimited cache for LRU and LRU+pre-filter of 100; simulation results are shown as crosses for $C = 1000$; $N = 10^4$ items with Zipf(.8) popularity.

lifetimes cannot be measured for the 85% most unpopular items with fewer than 10 requests in the entire trace. Table 2 consolidates the results from [14], grouping in class 5 all items with lifetime > 13 days or less than 10 requests in the entire trace. Somewhat arbitrarily in the absence of data, we suppose the mean lifetime of these items is one year. We discuss the impact of divergence from this assumed lifetime distribution below.

The average item popularity in the classes decreases with increasing mean lifetime. In our evaluations we assume the items in class j are in fact all more popular than the items in class k when $j < k$. While these data are clearly very rough, they do allow us to appreciate the significant impact of time locality on cache performance.

Numerical results.

In the formulas, lifetimes τ_i are normalized with respect to the overall request arrival rate since we have $\sum q_i = 1$. In Figure 3 we plot the overall hit rate against overall demand σ expressed as the mean number of requests per item per day. The corresponding mean lifetimes are then $\tau_i = \sigma N l_k$, for $\sum_{j<k} f_j N < i \le \sum_{j\le k} f_j N$ and $k \le 5$. We choose to normalize with respect to the catalogue size since, for Zipf(α) popularity with $\alpha < 1$, hit rates $h(\sigma, N)$ tend to a limit as $N \to \infty$ (with filter size K increasing proportionally).

Figure 3 plots hit rates against σ for an LRU cache and for a cache equipped with a pre-filter of size 100. The catalogue size is 10^4 and popularity is Zipf(.8). Simulation results for $C = 1000$, derived *without* the assumption that an old version of item i is removed when a new version is inserted, are shown as crosses and confirm the accuracy of the formulas. Lifetimes in the simulations have geometric distributions with means given in Table 2.

The depicted results demonstrate that, though the pre-filter brings close to ideal performance for stationary demand (i.e., for large σ), performance at moderate demand is worse than LRU. Neither caching strategy is effective when demand is low. The difference between the two policies for an unlimited cache is due to the fact that the pre-filter imposes a miss for the first two requests for a new item whereas LRU imposes only one miss.

The lifetime data constitute the best guess we have but are clearly imprecise. We have therefore examined the impact of deviations from our assumptions. As previously noted, as N increases, the hit rates as a function of σ tend to a limit. The curves in this limit are slightly above those depicted in the figure. Increasing all lifetimes by a common factor gives the same curves translated horizontally by the same factor. Changing the lifetime of class 5 from 1 year to 6 months or 2 years, say, shifts the lower part of the curves by a factor of 2 to the right or the left, respectively without modifying the top that is mainly determined by classes 1 to 4. Slightly modifying the lifetimes of the latter (e.g., lifetimes $l = \{10.6, 6.4, 3.3, 1.1\}$ instead of $\{1.1, 3.3, 6.4, 10.4\}$) has only a small impact on computed hit rates.

3. DEMAND AND CACHING POLICIES

Depending on the volume and characteristics of demand it may or may not make sense to apply a reactive caching policy like LRU or LRU with pre-filter (cf Sec. 2). If demand is too low, we need to envisage a proactive pre-fetching policy where the most popular items are determined externally. Note that cache performance has been shown to be largely independent of chunk size and we therefore choose to express catalogue size as a volume in bytes.

3.1 Traffic mix

For illustration purposes, we consider two types of content with very different catalogue sizes, 1 TB and 1 PB, respectively. The first might represent VoD where the number of titles available in the US is currently around 10^4 and mobile compatible rates limit the mean byte volume per object to around 100 MB. The second 1 PB catalogue corresponds to all other content. This is a rough order of magnitude estimate accounting for the data volume represented by applications like user-generated video content, social networking, file sharing and the web.

In traffic statistics published by Sandvine [13], 95% of mobile access downstream traffic in North America is content retrieval with VoD applications counting for some 20% of this. The VoD proportion has an increasing trend. In fixed networks the VoD proportion already attains 60%.

3.2 Popularity laws

For the sake of simplicity and reproducibility we choose to model popularity for both catalogues using a Zipf law with exponent 0.8. This value is typical of what has been observed for the main part of the popularity law (e.g., [7, 11]). The Zipf law does not accurately model the tail of the law, however, and care must be taken in interpreting the assumed catalogue sizes.

The BitTorrent popularity data described in [11] reveal a significant tail where popularity decreases much faster than Zipf(.8). The observed torrent catalogue was of 1.6 PB with 1.5 PB in the tail counting for 31% of requests. To model this law by a Zipf(.8) having the same weight of requests in the tail, and therefore the same ideal cache hit rates for capacities up to the start of the tail, the catalogue size should be limited to only 500 TB.

The above discussion highlights a deficit in our understanding of popularity, despite numerous published measurements. The common approach of trace analysis appears unable to correctly determine the behaviour of the tail of the law: the trace must typically be very long to observe any requests for the least popular items; on the other hand, we know that popularities can change quite rapidly and can only be measured reliably over a relatively short period.

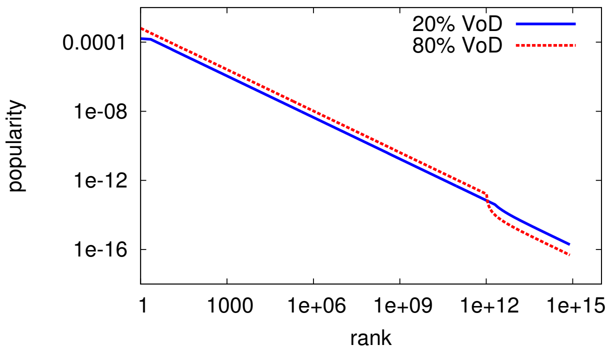

Figure 4: Popularity laws for a composition of 1 TB catalogue, representing VoD, and 1 PB catalogue, representing all other content; both catalogues have Zipf(.8) popularity and the relative traffic volumes are 20:80 and 80:20, respectively.

We consider the 1 TB and 1 PB catalogues individually to evaluate the impact on performance and cost effectiveness of what might be considered two extremes. We also combine the catalogues with traffic proportions 20:80 and 80:20, respectively. The resulting popularity laws are depicted in Figure 4. From traffic data reported in Section 3.1, the 20:80 combination is most plausible.

Time varying popularities.

Figure 3 shows the impact on hit rates of time varying popularities for individual Zipf(.8) catalogues. Figure 5 shows hit rates for an unlimited cache without pre-filter as a function of normalized demand for 100%, 80%, 20% and 0% VoD. The lifetime distribution of Table 2 is applied to each catalogue separately.

Demand σ is normalized with respect to the overall catalogue of 1.001 PB, even when we assume the VoD proportion is 100%. The difference between hit rate plots for 0% and 100% VoD corresponds therefore to a translation by a factor of 1000. These two plots constitute a reference to understand the impact of the mix. When VoD counts for only 20% of demand, the performance of reactive caching is close to that of a single 1 PB catalogue. When VoD represents 80% of demand, the hit rate first behaves like the 100% VoD plot but differs for the largest hit rates when the tail of the 1 PB catalogue popularity law predominates.

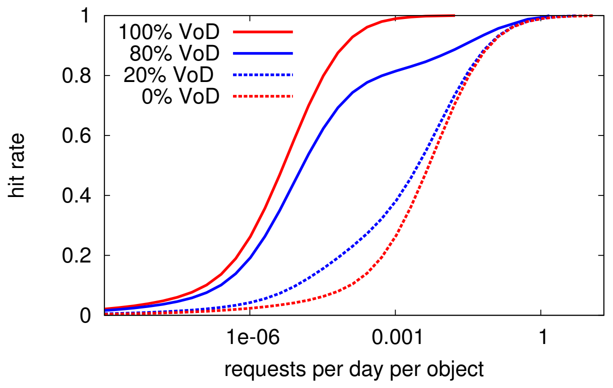

Figure 5: Hit rate against normalized demand σ when content is from two catalogues, a 1 TB catalogue representing VoD and a 1 PB catalogue representing all other content. The proportion of VoD demand varies from 0 to 100%.

3.3 Traffic volumes

Based on traffic records from a mobile operator, we consider two base station demand scenarios:

- "current demand" with 40 Mb/s busy hour content traffic and 144 GB daily download (8 times the busy hour volume),
- "future demand" with 240 Mb/s busy hour and 864 TB per day, accounting for a possible future sixfold capacity extension.

PGW and MCN concentrate the traffic of 1000 and 100 BS, respectively, and are supposed to have proportional demand with the same popularity laws.

3.4 Caching policies

We suppose the objective is to realize ideal cache performance, either by using a pre-filter cache policy or by proactively filling caches with the most popular items as determined by some external analysis. We discuss which of these policies is viable at each of the considered cache locations.

Base stations.

Even for the most optimistic scenario of 240 Mb/s demand for 1 TB of content (i.e., 100% VoD), the number of requests per day per item (σ = 864 GB / 1 TB) is less than 1 and, according to the lifetimes assumed for Figure 3, the considered reactive caching policies (either LRU or LRU with pre-filter) are not completely efficient. They are even less so for smaller demand (e.g., 40 Mb/s) or a bigger catalogue (e.g., 1 PB). It would be far preferable therefore to perform pre-fetching to populate the cache.

Pre-fetching would need to be orchestrated by a server that is aware of current demand from a much larger population of users than that of a single BS. This server would need to be made aware of all user requests, even those that result in a hit at the BS. It might be located at the PGW or at an upstream node concentrating demand from multiple access networks. Note that to inform such a centralized entity of user requests would also facilitate necessary content provider control over delivery of its particular content items (for billing, accounting, ad placement, etc.), as in current CDN solutions

This server would perform, so far unspecified, data analytics to determine the most popular items for each BS accounting for the nature of its particular population of users. Popularity might for instance be determined using the pre-filter, as long as demand σ is high enough, with changes to local BS content performed incrementally as necessary.

Packet gateway.

The PGW may or may not concentrate sufficient traffic to make reactive caching efficient. We find $\sigma = 144$ with the current demand scenario and a catalogue of 1 TB (meaning the pre-filter gives close to ideal hit rates) but $\sigma = .144$ only, if the catalogue size is 1 PB.

If the latter case applies one would need proactive pre-fetching of the most popular items, even at this highest level of concentration. This is clearly problematic for a mobile provider whose knowledge of demand is limited to that of its own limited population of users. The provider would typically need to be informed of demand from a much larger population of users than its own.

Rather than creating a cache in the PGW, one can thus envisage the creation of a large-scale content store shared by multiple mobile and fixed operators and physically located at the same point of presence at the edge of the Internet. This content store would take the form of a data center with the data analytics capacity needed both to perform content provider functions and to determine the items that need to be pre-fetched to BS caches.

It is significant that this calls for a separation of functions between the mobile access transport infrastructure provider on one hand and a large-scale content store provider on the other. The former would share the cost of the latter, typically by some usage-based pricing scheme.

Mobile cloud node.

The envisaged mobile cloud node plays the role of a BS with 100 times more demand. This would make reactive caching viable for the current and future demand scenario and a catalogue of 1 TB when $\sigma = 14.4$ and 86.4, respectively. Pre-fetching would still be necessary, however, for the larger catalogue of 1 PB since σ is then 1000 times smaller.

3.5 Discussion

The above results and observations may have significant impact on the ICN architecture and business model.

Our conclusion that pre-fetching is necessary to ensure efficient BS caching implies a central server (at the PGW or higher) would need to be made aware of all requests, including BS hits, to be able to track dynamic popularity. This is likely necessary in any case to provide content providers adequate control over delivery of their content. An MCN would improve caching efficiency but still require pre-fetching if our assumption that the catalogue attains 1 PB and not 1 TB is correct.

For the PGW, reactive caching is still only just effective for the larger catalogue. It may be preferable in this case to create a content store that federates the content demand from multiple access network providers. This would lead to a separation of functions between the content store operator and the access network operators. The former would likely host content as well as provider control logic (for accounting, ad placement, etc.).

rate (Mb/s)	charge ($)	rate (Mb/s)	charge ($)
10	500	100	850
20	575	200	1025
50	750	300	1220

Table 3: Monthly backhaul connection charges

4. MEMORY BANDWIDTH TRADEOFFS

Whether it is profitable to install a cache and what size it should have depends on the realized memory for bandwidth tradeoff. In this section we seek to evaluate this tradeoff at the BS, PGW and MCN locations assuming that caching is ideal thanks either to an efficient reactive cache policy or to pre-fetching.

4.1 Base stations

We propose rough cost estimates and evaluate the tradeoff realized under current and future BS demand scenarios.

Cost of storage.

We suppose base stations would either be equipped with a dedicated cache of some given capacity or have no cache. BS cache capacity is limited for cost and complexity reasons and, for the present evaluation, we suppose this capacity is 1 TB. Such a cache could be realized using a solid state disk with 1 TB hard drives currently priced at around \$400. A DRAM cache would cost roughly 20 times more.

To the cost of the hard drive must be added that of its server logic yielding a total caching cost of perhaps \$1000. Note that cost trends tracked over many years show the price of memory is decreasing by around 40% each year[1].

Cost of bandwidth.

We assume the backhaul is sized to realize a busy hour utilization of 80%, i.e., required connection bandwidths are 50 and 300 Mb/s for the current and future demand scenarios, respectively. The mobile operator typically pays a fixed network operator for backhaul capacity. We suppose a monthly charge per Mb/s that depends on demand, as given in Table 3. The table is based on current tariffs for optical backhaul in France. These tariffs have been fairly stable over the past 3 years. The charge determines the bandwidth cost of the mobile operator irrespective of the distance from the PGW.

Tradeoff.

An assumed cache cost of \$1000 amortized over a 3 to 4 year lifetime corresponds to a monthly charge of barely \$25 to be compared to the monthly cost of the connection. According to the tariffs of Table 3, the 1 TB cache would economize \$250 (\$750 − \$500) per month for current demand and \$470 (\$1220 − \$750) for future demand if the catalogue were only 1 TB (100% hit rate). On the other hand, bandwidth savings would hardly be worthwhile if the catalogue

[1]J. C. McCallum: http://www.jcmit.com/mem2014.htm

were 1 PB since the hit rate is then less than 10%, even for ideal caching.

For a catalogue composed of 1 TB of VoD and 1 PB of other content in proportions 20:80, an ideal cache of 1 TB would give a hit rate of around 30%. This is significantly more than the 20% obtained by caching exclusively the VoD content. This is because it is more effective to cache the most popular other content items than the least popular VoD items. The average saving would then be around $80 per month per BS in both demand scenarios. Assuming the more favourable proportions of 80% VoD and 20% other content yields an ideal cache hit rate of nearly 85% and savings only slightly less than those given above for an individual 1 TB catalogue.

4.2 Packet gateway

The packet gateway is assumed to concentrate the traffic of 1000 BSs with, therefore, a busy hour downstream demand of 40 Gb/s and 240 Gb/s for the current and future scenarios, respectively. The PGW might have its own dedicated cache or share a co-located cache with other fixed and mobile operators. In the latter case, the tradeoff must be considered as a whole with the assumption that a given operator shares the cost of caching in proportion to its demand.

For the considered large scale cache at this location, we assume linear cost functions and seek therefore to minimize

$$\Delta = k_b T(1 - h(C)) + k_m C, \tag{3}$$

for particular values of cost coefficients k_b and k_m. Now, for Zipf(α) popularity with $\alpha < 1$ and a large catalogue N, it is known that the hit rate $h(C, N)$ is in fact a function of the ratio $c = C/N$. This is true also for the considered combinations of Zipf catalogues. It is convenient therefore to re-write (3) in a normalized form as follows:

$$\delta(c) = \Gamma(1 - h(c)) + c, \tag{4}$$

where $\Gamma = k_b T/k_m N$. Γ is the ratio of the cost of no cache to the cost of a full capacity cache and summarizes the combined impact of the demand and cost parameters.

Cost of storage.

We base storage costs on the charges levied for data storage by Cloud providers. This is currently around $.03 per GB per month. The traffic related cost of retrieving data from the cache is small compared to the cost of network bandwidth and is therefore neglected in our tradeoff evaluation.

Cost of bandwidth.

From Table 3, the cost of backhaul bandwidth is a decreasing function of demand and the lowest incremental charge for connections of rate greater than 200 Mb/s is $2 per Mb/s. In the absence of more precise data, we take this as the value of k_b. This would be a charge paid by the mobile operator for traffic retrieved from the Internet. The actual tariffs are certainly more complex than this but they are not public and the linear cost model allows for simpler comparisons.

Tradeoff.

Figure 6 plots normalized cost δ against normalized cache size c for Zipf(.8) popularity assuming ideal caching. The blue lines correspond to different values of Γ between 0.01 and 100, as determined by their intercept on the y-axis. Vertical lines give the hit rate realized by the given cache size.

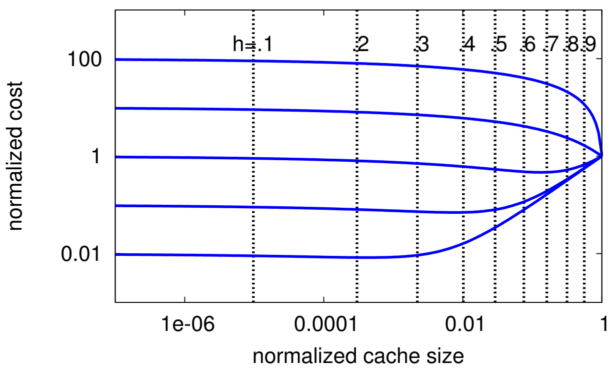

Figure 6: Normalized cost against normalized cache size for Zipf(.8) popularity and ideal caching. The horizontal lines correspond to values of Γ of .01, .1, 1, 10, 100 from bottom to top.

From the figure, for most values of Γ, the optimal choice is either no cache ($c = 0$) or a full capacity cache ($c = 1$). As can readily be verified, the cost δ is minimized for some $c \in (0, 1)$ if $\Gamma < 5$. The gain is very limited when Γ is small, however. Maximum gain relative to $c = 0$ or $c = 1$ occurs for $\Gamma = 1$ and $c = .13$ when $\delta = 0.47$.

Using the above cost estimates, $k_b = \$2$ per Mb/s, $k_m = \$.03$ per GB, we have $\Gamma = 2000$ for current demand and a 1 TB catalogue. It is clearly optimal in this case (and *a fortiori* with the future demand scenario) to equip the PGW with a full capacity cache. The choice is less obvious for a catalogue of 1 PB where Γ is only 2 for current demand. Rather than optimizing the cache size (at around 10% of the catalogue), it would make more sense in this case to share a common cache with multiple mobile and fixed operators having co-located facilities at the edge of the Internet, as discussed above in Section 3.

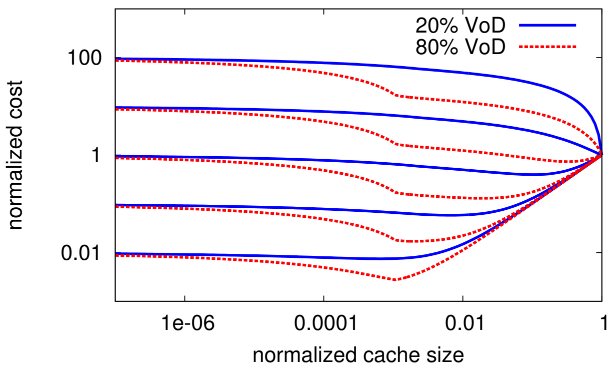

Figure 7: Normalized cost against normalized cache size for mix of popularity laws and ideal caching: VoD and other. The horizontal lines correspond to values of Γ of .01, .1, 1, 10, 100 from bottom to top.

Figure 7 plots the cost δ for the two catalogue mixes with 20% VoD and 80% VoD, respectively. Conclusions for the more plausible 20% VoD mix are broadly the same as for homogeneous Zipf(.8) popularity. When VoD counts for 80% of demand, on the other hand, there is scope for optimizing cache size. For example, for $\Gamma = 2$ (i.e., for the current demand scenario) it would be optimal to equip the cache with a capacity of 40 TB. This would yield a hit rate of 90% with cost reduced by a factor of 4 compared to the cache all solution.

4.3 Mobile cloud node

The MCN node is supposed to be equipped as a small data center performing various network functions and application offloads so that marginal storage charges are appropriate. We suppose these are the same as for a PGW cache. Since demand is greater than 200 Mb/s, we also assume the marginal bandwidth charge of \$2 per Mb/s applies to the considered content traffic.

The tradeoff can be derived from Figures 6 and 7 on calculating the appropriate values of Γ. For instance, for current demand bandwidth requirement $T = 5$ Gb/s and catalogue size $N = 1$ TB, we have $\Gamma = 333$ and to cache all content at the MCN would be cost effective. On the other hand, if $N = 1$ PB we have $\Gamma = .33$ and caching brings only marginal gains.

Note that caching at the MCN is an alternative to caching at the BS. Its greater effectiveness constitutes an additional argument for actually creating such a node in future mobile access networks.

4.4 Discussion

At the BS, a 1 TB cache costing \$1000 would be beneficial if pre-fetching were used to realize ideal caching (cf. Sec. 3). Cost trends suggest this tradeoff will become more favorable over time as the cost of memory is decreasing faster than the cost of bandwidth.

The charts in Figures 6 and 7 show that it is rarely worthwhile to optimize the size of the cache at the PGW where simply caching the entire catalogue is the most cost effective solution. This is not true for the 80% VoD scenario but this not realistic according to current usage statistics. It must be remembered that this conclusion applies under the simplifying assumption that catalogue popularity is Zipf. If the tail of the popularity law were more accurately represented, with a significant volume of very unpopular items, it would likely make sense to store only a fraction of the catalogue leading to a hit rate of 99%, say.

5. RELATED WORK

The models for cache performance in Section 2 are developments on the approximation originally proposed by Che *et al.* [1] and extended since, notably by Fricker *et al.* [6] and Martina *et al.* [9]. We further extend the approach here to account for time varying popularities for both an LRU cache and our particular design of an LRU cache with pre-filter. The corroborating simulation results presented here confirm the versatility and high accuracy of the Che approach.

The pre-filter cache policy we describe is an adaptation of the discrete persistent access cache proposed by Jelenkovic *et al.* [8]. It is simpler than insertion policies like k-LRU, where the filter is one or more successive virtual LRU caches, and more reactive to changing popularity than probabilistic q-LRU caching, where items are inserted with probability q. These names were given by Martina *et al.* [9] whose models directly inspired those developed here.

The need to take proper account of time varying popularities has been underscored by recent analyses of trace statistics, notably by Traverso *et al.* [14], Olmos *et al.* [10] and Imbrenda *et al.* [7]. The authors of both [14] and [10] propose a so-called shot noise model where items are born at some instant in time, receive requests at possibly varying rate for a certain lifetime and then die. The overall request process is a superposition of such shots. In our approach, items that die are immediately replaced by a new item with identical popularity, simplifying demand characterization and allowing a straightforward application of the Che approach. This model or content churn was first used by Wolman *et al.* [15] in a study of hit rates in an unlimited capacity web cache. We generalize the model of [15] by considering finite capacity caches with LRU or pre-filter caching policies.

The direct evaluation of the memory for bandwidth tradeoff has received little direct attention in the literature. Cidon *et al.* [2] consider the problem of minimizing the joint cost of bandwidth and storage cost in a tree network but provide no numerical evaluations. Erman *et al.* [4] apply a similar approach to ours for a 3G access network assuming unlimited cache capacity and applying trace data. Our formulation of this issue is derived from prior work on the effectiveness of edge caching [11]. The present work provides a quantified evaluation of the tradeoffs in a mobile access network using our best guess demand and cost data.

6. CONCLUSIONS

To evaluate the effectiveness of caching in the mobile access network we have developed original mathematical models to determine the hit rate as a function of content popularity statistics. We have notably proposed a model to account for time varying content popularities. The results show that using a pre-filter makes an LRU cache have close to ideal performance under stationary popularity but can be counterproductive due to the impact of content popularity churn when the request rate is low.

The analytical models have been applied to determine the effectiveness of reactive caching, using LRU or LRU with a pre-filter and relying, therefore, only on locally gained knowledge of content popularity. Results for the most plausible demand scenarios suggest reactive caching at the BS or MCN levels would not be efficient and that a proactive policy of prefetching would therefore be necessary. This implies some upstream entity must be made aware of requests from a population of users that is large enough that request rates are typically much greater than the rate of churn. Even the PGW may not have sufficient demand to enable efficient reactive caching in the more pessimistic scenarios implying that the upstream entity in question might be located in a common cache at the Internet edge shared by multiple access networks.

The memory for bandwidth tradeoff is favourable at all levels of the considered access network. In the scenario considered most plausible where 20% of demand is for VoD, a relatively small 1 TB cache at the BS would be cost effective in reducing upstream traffic by 30%. The addition of the MCN level would be beneficial by enabling a single cache to be shared by up to 100 BS and, under our assumptions, enabling more flexible cache sizing. A cache at the PGW would generally be cost effective when sized to capture up to 99% of all requests. It may, however, make more economic sense to share a dedicated content store located in proximity to the PGW at the Internet edge between multiple access network providers.

It is clear that the above conclusions rely on input data on demand, popularity and costs that are necessarily questionable. However, we have considered a range of scenarios and illustrated the impact of possible deviations from our best guess data and believe the above conclusions to be robust. The methodology and formulas can of course be applied with alternative, more precise data when these are available. Evaluating the realized memory for bandwidth tradeoff, even imprecisely as here, more directly informs the debate on ICN caching architecture than any consideration of the optimal placement of a hypothetical "cache budget".

Acknowledgment

This research work has been partially funded by the SystemX Technological Research Institute within the "Network Architecture" project hosted at LINCS.

7. REFERENCES

[1] H. Che, Y. Tung, and Z. Wang. Hierarchical web caching systems: modeling, design and experimental results. *IEEE JSAC*, 20(7):1305–1314, 2002.

[2] I. Cidon, S. Kutten, and R. Soffer. Optimal allocation of electronic content. *Computer Networks*, 40(2):205 – 218, 2002.

[3] A. Dabirmoghaddam, M. M. Barijough, and J. Garcia-Luna-Aceves. Understanding optimal caching and opportunistic caching at "the edge" of information-centric networks. In *Proceedings of the 1st International Conference on Information-centric Networking*, INC '14, pages 47–56, New York, NY, USA, 2014. ACM.

[4] J. Erman, A. Gerber, M. Hajiaghayi, D. Pei, S. Sen, and O. Spatscheck. To cache or not to cache: The 3g case. *Internet Computing, IEEE*, 15(2):27–34, March 2011.

[5] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker. Less pain, most of the gain: Incrementally deployable icn. *SIGCOMM Comput. Commun. Rev.*, 43(4):147–158, Aug. 2013.

[6] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for LRU cache performance. In *Proceedings of ITC 24*, 2012.

[7] C. Imbrenda, L. Muscariello, and D. Rossi. Analyzing cacheability in the access network with hacksaw. In *Proceedings of the 1st International Conference on Information-centric Networking*, INC '14, pages 201–202, New York, NY, USA, 2014. ACM.

[8] P. Jelenkovic, X. Kang, and A. Radovanovic. Near optimality of the discrete persistent access caching algorithm. In *International Conference on Analysis of Algorithms,*, 2005.

[9] V. Martina, M. Garetto, and E. Leonardi. A unified approach to the performance analysis of caching systems. In *INFOCOM, 2014 Proceedings IEEE*, pages 2040–2048, April 2014.

[10] F. Olmos, B. Kauffmann, A. Simonian, and Y. Carlinet. Catalog dynamics: Impact of content publishing and perishing on the performance of a lru cache. In *Teletraffic Congress (ITC), 2014 26th International*, pages 1–9, Sept 2014.

[11] J. Roberts and N. Sbihi. Exploring the memory-bandwidth tradeoff in an information-centric network. In *Teletraffic Congress (ITC), 2013 25th International*, pages 1–9, Sept 2013.

[12] G. Rossini and D. Rossi. Coupling caching and forwarding: Benefits, analysis, and implementation. In *Proceedings of the 1st International Conference on Information-centric Networking*, INC '14, pages 127–136, New York, NY, USA, 2014. ACM.

[13] Sandvine. Global internet phenomena report - 2h 2014. White paper, 2014.

[14] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini. Temporal locality in today's content caching: Why it matters and how to model it. *SIGCOMM Comput. Commun. Rev.*, 43(5):5–12, Nov. 2013.

[15] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy. On the scale and performance of cooperative web proxy caching. *SIGOPS Oper. Syst. Rev.*, 33(5):16–31, Dec. 1999.