On sizing CCN content stores by exploiting topological information

Dario Rossi, Giuseppe Rossini
Telecom ParisTech, Paris, France – first.last@enst.fr

Abstract—**In this work, we study the caching performance of Content Centric Networking (CCN), with special emphasis on the size of individual CCN router caches. Specifically, we consider several graph-related centrality metrics (e.g., betweenness, closeness, stress, graph, eccentricity and degree centralities) to allocate content store space *heterogeneously* across the CCN network, and contrast the performance to that of an *homogeneous* allocation.**

To gather relevant results, we study CCN caching performance under large cache sizes (individual content stores of 10 GB), realistic topologies (up to 60 nodes), a YouTube-like Internet catalog (10^8 files for 1PB video data). A thorough simulation campaign allow us to conclude that (i) , the gain brought by content store size heterogeneity is very limited, and that (ii) the simplest metric, namely degree centrality, already proves to be a "sufficiently good" allocation criterion.

On the one hand, this implies rather simple rules of thumb for the content store sizing (e.g., "if you add a line card to a CCN router, add some content store space as well"). On the other hand, we point out that technological constraints, such as line-speed operation requirement, may however limit the applicability of degree-based content store allocation.

I. Introduction

Content centric networking (CCN) is a promising architecture to diffuse popular content over the Internet. As opposite to cache devices such as P2P-accelerator or Web proxies, CCN is content-aware but service-agnostic, so that it promises to offer a caching service that could not only relieve nowadays congestion, but would also apply to any future services or applications as well.

CCN is based on the idea that, instead of caching full objects, very small data chunks (typically packet-size) should be cached by CCN routers. Each of these data chunks can be unequivocally identified by users, i.e., they are *named* data chunks. In CCN, user requests for objects are expressed as a stream of interest packets for named data, that propagate in the CCN network. As each router in the network can be a CCN cache (or *content store* in CCN terminology), whenever an interest hits a cache, the corresponding named data packet is sent back in response (and may get cached along the way back). For reason of space, we cannot provide here a full overview of CCN inner working – hence, we refer the reader to [12] for the full details or our previous work [8], [10] for an overview.

Though the interest in CCN is motivated by the relevance of the problem it aims to solve, nevertheless an assessment of the achievable performance is needed to justify the hype. As CCN is deeply based on caching, this clearly constitutes one of the main system aspects that deserve a thorough performance evaluation.

Caching is definitively not a new topic, with many work delving into Web caching replacement and decision policies (see [15] and references therein). Both single-cache and multi-cache system have been studied, though with limited exceptions most work on multi-cache architectures consider a well structured *hierarchy of caches*, generally arranged as cascade or tree topologies [6]. Even more recently, with the interest of caching studies refueled by information-centric architectures, cascade [2]–[4] or tree [3], [4], [16] topologies are generally considered – though studies on caches arranged as *arbitrary networks* have started to appear [7], [8], [10], [13], [17].

In our previous work [8], [10], we studied arbitrary networks of CCN caches with special attention to different caching replacement (e.g., LRU, MRU, etc.) and decision (e.g., LCD [14], etc.) policies, and to the use of single vs multiple-paths in the CCN strategy layer. At the same time, [8], [10] consider homogeneous cache sizes, i.e., CCN content stores have all equal size. In this work, we take an orthogonal direction to [8], [10] and explore whether *heterogeneous cache size* can improve CCN caching performance. With this regard, closest work to our is [9], that studied hierarchies of Web caches, reaching the conclusion that "stream aggregation (perhaps more than network topology) is a key factor in optimizing cache placement". In [9] the topology is however abstracted by considering several traces, captured at different depths in the end-to-end path connecting a Web browser to a Web server (i.e., end-user client, client proxy after browser cache, network proxy and end-server).

In this work, we instead take an orthogonal approach to [9], and explicitly model the network topology as an arbitrary graphs $G = (V, E)$, of which we compute several centrality metrics (e.g., betweenness, closeness, stress, graph, eccentricity and degree centralities). We then use the centrality values of each node as a base for different strategies to heterogeneously distribute a given amount of cache, measuring the performance gain (or loss) *relatively* to an homogeneous CCN network having the same overall cache amount. Conversely, we refer the reader to [8], [10] for a more detailed focus on the *absolute* performance of homogeneous CCN networks.

Summarizing our main findings, first we gather that, disappointingly, the gain brought by heterogeneity of CCN caches is very limited. Hence, these results generalize the conclusions of [9] to an arbitrary graph of CCN caches storing chunk-level content.

Second, we find that the simplest metric, namely degree centrality, already proves to be a "sufficiently good" allocation criterion: this is a rather positive results, as it implies rather simple rules of thumb for the content store sizing (e.g., the CCN operational team of an ISP can be safely told that "if you add a line card to a CCN router, remember to add some content store space as well"). At the same time, we point out that technological constraints, such as line-speed operation requirement, may unfortunately limit the applicability of degree-based content store allocation – since nodes with large degree must still be able to operate at the *aggregate* line speed, which may tradeoff with the cost of building large and fast caches [2].

II. On graphs and topologies

To perform a thorough assessment of heterogeneous CCN content stores, we consider several real topologies and cache sizing strategies.

First, we consider 5 publicly available network topologies, that are either available on the Web or through [18]. Assuming to operate in a non-congested regime, we simulate links having infinite capacity (so that the delay matches the propagation delay). Tab. I reports the main characteristics of each graph $G = (V, E)$, namely, the network size $|V|$, the average degree $|E|/|V|$, the coefficient of variation of the node degree CoV, the average link propagation delay δ and graph diameter D. Next, for each topology, we compute the following graph-related metrics.

- **Degree Centrality: DC(n)** is defined as the number of links incident upon node n (as we consider bidirectional graphs, we do not differentiate between indegree/outdegree).
- **Stress Centrality: SC(n)** reflects the total number of shortest paths (or geodesics) between all other nodes which run through n. It is defined as $BC(n) = \sum_{\forall s,t \in V \setminus n} \sigma(s,t,n)$, with $\sigma(s,t,n)$ the number of shortest paths from s to t through n.
- **Betweenness Centrality: BC(n)** reflects how often the node n lies on the shortest paths between all the other nodes of the network. It is defined as $BC(n) = \sum_{\forall s,t \in V \setminus n} \delta(s,t,n)$, where $\delta(s,t,n) = \sigma(s,t,n)/\sigma(s,t)$ is the fraction of all shortest paths between s and t which run through n. In a sense, it is a normalized version of SC.
- **Closeness Centrality: CC(n)** relates to the distance of n to all the other nodes in the network: the lower the total distance toward all other nodes, the more the node is central in the topology. It is defined as the invert sum of the shortest path distances of node n from all other nodes, $CC(n) = 1/\left(\sum_{\forall s \in V \setminus n} d(n,s)\right)$ where $d(n,s)$ is the length of the shortest path from n to s.
- **Graph Centrality: GC(n)** relates to the distance of n to the farthest node: nodes with high GC have short distances to all other nodes in the graph. It is defined as the invert of the maximum of all geodesic distances from a node to all other nodes in the network, i.e., $CC(n) = 1/\max_{\forall s \in V} d(n,s)$ where $d(n,s)$ is the length of the shortest path from n to s.
- **Eccentricity Centrality: EC(n)** reflects how far, at most, is node n from every other node It is defined as the largest geodesic distance $EC(n) = \max_{\forall s \in V} d(n,s)$ and thus mirrors the GC definition.

TABLE I
Network topologies.

	Segment	$\lvert V\rvert$	$\lvert E\rvert/\lvert V\rvert$	CoV	$\delta[ms]$	D
(a) Abilene	Core	11	2.54	0.19	11.3	8
(b) Tiger2	Metro	22	3.60	0.17	0.11	5
(c) Geant	Aggr	22	3.40	0.41	2.59	4
(e) Level3	Core	46	11.65	0.86	8.88	4
(d) DTelekom	Core	68	10.38	1.28	17.21	3

TABLE II
Correlation coefficient E[ρ] among centrality metrics.

E[ρ]	DC	BC	CC	EC	GC	SC
DC	1	0.802	0.846	-0.523	0.538	0.903
BC	.	1	0.884	-0.655	0.674	0.949
CC	.	.	1	-0.790	0.794	0.906
EC	.	.	.	1	-0.984	-0.633
GC	.	.	.	.	1	0.652
SC	.	.	.	.	.	1

Note: E[ρ] is averaged over the 6 topologies: the standard deviation is std(ρ) < 0.167 for any metric-pair, and the coefficient of variation is always below std$(\rho)/E[\rho] < 0.310$

The relationship among the different criticality scores is expresses in Tab. II, that reports the average over all topologies of the correlation coefficient between any two sets of centrality values[1]. Though this list is non exhaustive, as other metrics (e.g., the lobby index, etc.) could be included as well, it is nevertheless already fairly representative. It could be rather objected that it is redundant to consider so many centrality metrics, especially since some of them are similar (e.g., BC and SC) or strongly negatively correlated (e.g., EC and GC). Still, we point out that each metric has its own pros and cons. Consider for instance EC and GC. On the one hand, it could be argued to give more cache space to high-GC nodes, as they have short distance to all other nodes in the graph, and may thus act as "shared hubs" for content requests passing by. Conversely, it could also be argued to give more cache space to high-EC nodes exactly since they are faraway in the topology: otherwise, their requests may induce higher load on many CCN routers in the path, unless they could be filtered out by having access to a larger amount of cache.

Let us define C_{tot} as the overall size of cache in the topology. In the case of homogeneous network, we fix the size of the individual caches to $C_i = 10\,$GB, that [2] points out to be about nowadays technological limit due to line speed requirement. In other words, CCN routers must be able to service each interest packet by doing a lookup for content in real time (similarly to IP longest prefix matching lookup for addresses). Hence, memory access speed (and cost) limits the

[1] To simplify the visual presentation, we report only the matrix values above the diagonal: we point out that the correlation coefficient is invariant to the order of the vectors, and hence the matrix is actually symmetrical.

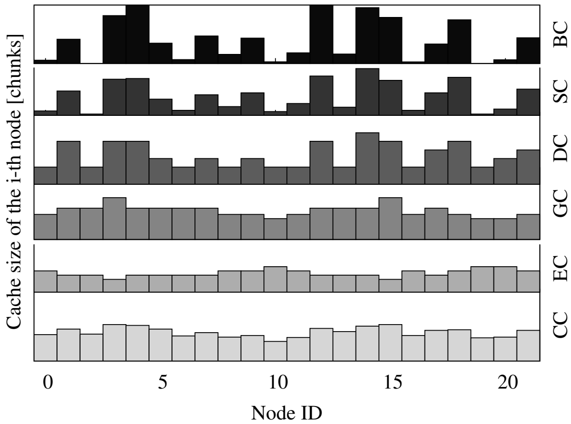
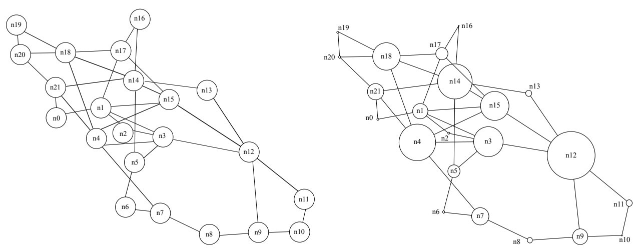


Fig. 1. Example of cache size for different nodes ranking metrics, Geant topology



(a) Homogeneous (b) Heterogeneous

Fig. 2. Pictorical representation of cache sizing, with CCN content store size proportional to the size of node in the corresponding graph. The picture reports (a) homogeneous and (b) heterogeneous cache sizes (in the latter case, proportional to the betweenness centrality BC index).

practically achievable content store size [2].

In the case of homogeneous networks, $C_{tot} = |V|C_i$ with $|V|$ the number of nodes in the network. In the case of heterogeneous networks, we exploit the centrality scores as follows. Consider a generic metric $X \in \{CC, GC, DC, EC, SC, BC\}$, where we denote by $X(i)$ the value of X for node $i \in V$ for the considered topology. We then adopt two criteria for cache sizing:

$$C_X^P(i) = C_{tot} \frac{X(i)}{\sum_{j \in V} X(j)}, \forall i \quad (1)$$

$$C_X^Q(i) = \max\left(c, \lceil C_X^P(i)/c \rceil c\right) \quad (2)$$

Notice that (1) corresponds to a perfectly *proportional* criterion, where the cache size $C_X^P(i)$ is distributed to the i-th node proportionally to the metric $X(i)$ normalized over the sum of the $X(i)$ score over the whole graph. An example of the (1) strategy computed for all centrality metrics on the Geant network is reported in Fig. 1 (where the centrality metrics X are sorted top to bottom by decreasing coefficient of variation of the score $\sigma(X)/E[X]$). For the sake of illustration, Fig. 2 depicts nodes of variable size, whose radius is proportional to the cache size $C(i)$, to visualize *where* in the network the cache resource have been allocated. Specifically, Fig. 2 contrasts an homogeneous Geant network where $C(i) = C_{tot}/|V|$ for all nodes (left plot) with the heterogeneous $C_{BC}^P(i)$ allocation (right plot) corresponding to the largest skew in the cache resource allocation (as seen in the top plot of Fig. 1). Clearly, the other centrality metrics will provide allocation skews in between these two extremes.

While (1) is an ideal strategy that allows to gauge the relative importance of the centrality score, we acknowledge that it may be hardly feasible in practice: indeed, CCN content store modules will be quantized in multiples of a unit module c, as it happens for nowadays RAM memory. As such, we also consider a *quantized* strategy (2), where the size of individual caches $C_X^Q(i)$ is multiple of $c = 1\,\text{GB}$ units. The model assumed by (2) is that will ISPs invest in a fixed number of memory modules C_{tot}/c that they can then arbitrarily deployed in the network. The viewpoint we adopt in this work is that an ISP may wish to reallocate the C_{tot}/c cache modules at its disposal (e.g., moving from an homogeneous setup to an heterogeneous one), so to optimize the achievable performance without incurring in further capital expenditure.

Notice that the quantization process induces an error so that the overall cache amount is now $C_{tot}(1+\epsilon)$ with $\epsilon \in \mathbb{R}$ the error induced by the quantization process. This error is due to two contrasting operations[2] in (2) that somehow compensate (the average error is $E[\epsilon] = 2.3\%$). It could be argued that any difference in terms of performance (e.g., cache hit gain) may be due to discrepancy in the quantized cache size (e.g., when $\epsilon > 0$). To rule out this possibility, we verify the absence of correlation between these discrepancies. More formally, denote H_{Const} the cache hit probability of the homogeneous network, with overall cache size C_{tot}. Denote then by $H_X^Q = (1+\gamma)H_{Const}$ the cache hit of an heterogeneous network with quantized cache allocation strategy according to the centrality measure X, with overall cache size $\sum_{i \in V} C_X^Q(i) = (1+\epsilon)C_{tot}$. The correlation coefficient $\rho_{\gamma,\epsilon}$ between the series of (γ, ϵ) pairs gathered over all networks and centrality measures, equals $\rho_{\gamma,\epsilon} = 0.05$, ruling out any correlation between these errors. As such, performance differences in the following solely depend on the centrality metrics.

Notice that Fig. 1 and Fig. 2 highlights variation in the cache allocation according to different centralities for the same topology. In Fig. 3 we finally reports a complementary view, showing the same centrality index (namely, BC) for all topologies. The x-axis represents the normalized node ID $i/|V|$, sorted by increasing BC index; the y-axis instead reports the cumulated fraction of content store size $\sum_{j=0}^{i} C_{BC}^P(i)/C_{tot}$, where we consider a proportional allocation strategy for the sake of illustration. It can be seen that in the DTelekom and Level3 network, BC index defines a very skewed allocation, with few nodes taking the most of the cache. Conversely, Geant, Abilene and Tiger provide a more balanced allocation (approaching the constant allocation depicted as a reference).

[2] The max operation imposing a minimum cache size $C_X^Q(i) \geq c \quad \forall X, i$ increases ϵ, while the ceiling operation reduces ϵ

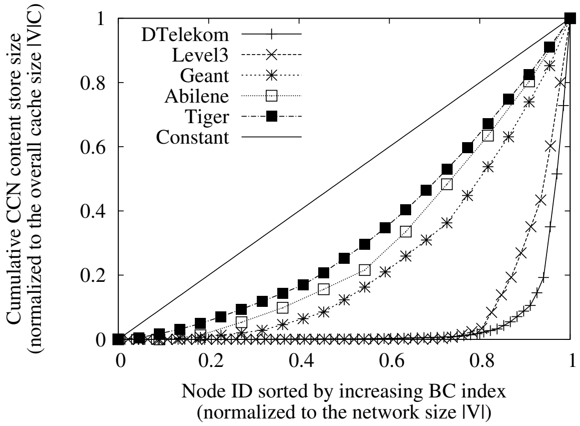


Fig. 3. Network topologies

III. Performance evaluation

A. Simulation scenario

To gather relevant results, we study CCN caching performance under realistic topologies (up to 60 nodes), large cache sizes (individual content stores of 10 GB), and YouTube-like Internet catalog (10^8 files for 1PB video data). Notice that our simulator, which we make available as open source software at [1], manages to achieve a much larger scale with respect to current literature [2]–[4], [7], [13], [16], [17].

We repeat simulations 20 times, with each simulation lasting for the minimum between the convergence of the cache hit and 3600 seconds after caches fill. Since we simulate 5 topologies, 1 homogeneous plus 6x2 heterogeneous strategies for content store sizing, for each popularity setting (i.e., value of the zipf catalog α) *we simulate nearly 1200 hours of CCN dynamics* (or, nearly 50 days worth of traffic).

While the topologies have been discussed in the previous section, we consider as in [2] that the size of the CCN content store is upper bounded due to nowadays memory access speeds, that must be able to follow network link capacity. A detailed analysis let authors in [2] to quantify to $C(i) = 10\,\text{GB}$ the size of individual caches achievable nowadays, that we also consider in this work.

The YouTube catalog size is estimated by [5] to amount to 10^8 video files, having geometrically distributed size with 10 MB average [11], yielding to a 1 PB catalog. Notice thus that each cache can only store a very small portion (10GB/1PB) of the whole catalog.

Concerning the catalog, an important parameter is the popularity of individual items, which is usually assumed to be Zipf, and also experimentally verified by [5] for the YouTube case. Clearly, the value of the Zipf exponent α plays a paramout role in determining the caching performance. Yet, there is no consensus for setting α that varies in the CCN literature between 1 [7] and 2.5 [3] (the latter taken from [5]). Notice however that [5] reports a $\alpha = 2.5$ fitting for a part of the slope the YouTube popularity: we point out that the $\alpha \geq 2$ values in [5] only fit part of the tail, while the body appears to follow a Mandelbrot-Zipf law. As we discuss in the following, considering such a large value for α could oversimplify the problem.

We studied the system performance under a large α range in [8], [10], of which we summarize here the main message concerning the impact of α. Let us recall that the Zipf probability distribution is defined as $P(X = i) = \frac{1/i^\alpha}{C}$ with $C = \sum_{j=1}^{|F|} 1/j^\alpha$ where i is the rank of the i-th most popular file in a catalog of size $|F|$ files. Thus, $\sum_{i=1}^{k} P(X = i)$ corresponds to the percentage of requests directed to the set of k most popular objects. Now, we observe that for large values of α, the caching problem becomes trivial: e.g., at $\alpha = 2$, out of the 10^8 files catalog, only 120MB of cache are required to cache the $k = 12$ files responsible for at least 95% of the user requests (as $\sum_{i=1}^{12} P(X = i) > 0.95$ for $\alpha = 2$). A sharp transition phase of the value of k corresponding to the 95% of requests happens between $\alpha = 1.25$ and $\alpha = 1.5$ where the bulk of requests concerns respectively about $k = 50,000$ files (or 500TB storage space) and $k = 234$ files (about 2.4GB). Comparing these sizes to that of individual caches $E[C(i)] = 10\,\text{GB}$, we expect these two α values to correspond to large variation of the caching performance. We argue that the $\alpha \in [1.25, 1.5]$ boundaries are representative of a wide spectrum of CCN performance (from [8], we expect the absolute cache hit to vary between 50% and 80%), and we consider both values in the following.

B. Simulation results

The aim of our wide-range simulation campaign is (i) to assess whether heterogeneous cache sizing can provide performance benefits over an homogeneous network and (ii) if performance gain are consistent across all topologies for some allocation metric $X \in \{CC, GC, DC, EC, SC, BC\}$.

We express caching performance with two output metrics. We consider *cache hit* probability H as the usual network-centric metric. As user-centric metric we consider the *path stretch* $d/|P|$ as the number of CCN backbone hops d that the data chunk has actually traveled in the network, normalized over the path length $|P|$ until the content originator (i.e., without caching). Note that $d = 0$ when users find the content at the edge CCN router and $d = |P|$ when the content is not cached by any CCN router, so that $d/|P| \in [0, 1]$.

First, we observe that *quantization* actually plays a beneficial role. Let us consider the cache hit metric H and as before denote by H_X^P (H_X^Q) the cache hit metric achieved for a given topology and popularity settings by using a proportional (quantized) allocation according to centrality metric X. We then define the relative error induced by quantization on cache hit as $(H_X^Q - H_X^P)/H_X^P$, which is depicted for all topologies and metrics in a monotonously increasing fashion in Fig. 4. In the picture, a negative value (gray shaded zone) correspond to cases (i.e., metric and topology combinations) where proportional allocation would yield better cache hit results. Interestingly, Fig. 4 shows that though proportional allocation yields better performance in some cases (left part), however the performance difference with the corresponding

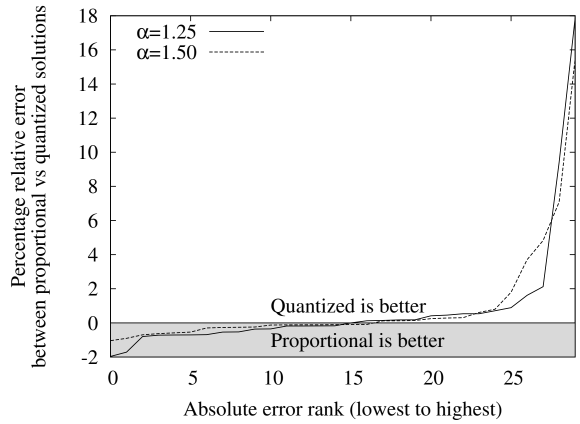


Fig. 4. Percentage relative error $(H_X^Q - H_X^P)/H_X^P$ for the cache hit metric of quantized vs proportional allocation, all topologies and allocation metrics

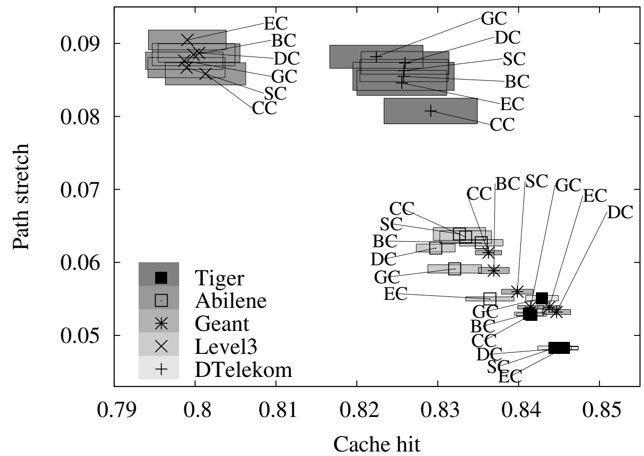


Fig. 5. Scatter plot of cache hit versus path stretch for different ranking (explicitly labeled) and topologies (represented with different points and box colors) when $\alpha = 1.5$. Each box is centered around the the average hit and stretch performance, with box width and height extending to the standard deviation of the above respective metrics.

quantized allocation is minimal (below 2%). Conversely, there are cases in which quantization can bring almost up to 20% gain with respect to a proportional allocation. Essentially, this is due to the fact that some metrics (especially, BC and SC, recall Fig. 1) may allocate a very low amount of cache space to some nodes, which is "corrected" by having a minimum amount of cache in the quantization process. In the following, we thus consider only quantized allocations: this choice is both robust (as we avoid outliers due to skew in the centrality metrics) and realistic (as a perfectly proportional allocation is not directly applicable).

To represent at a glance the impact of network topology and allocation criterion on the system performance, we depict in Fig. 5 a scatter plot of the cache hit versus path stretch for different criterion and topologies (reporting only $\alpha = 1.5$ for the sake of brevity). Each box is centered around the the average cache hit and stretch performance, while box width and height represent their standard deviations respectively. For the sake of illustration, each topology is represented using different points and box colors, while centrality metrics are explicitly labeled in the figure.

Notice that for Level3 and DTelekom, the cache hit is only modestly affected by centrality metrics. Notice that happens despite a significant unbalance in the network topology, with the vaste majority of the cache resources possibly allocated to few nodes only (as per Fig. 3). For instance, for Level3 all boxes roughly spans along the same x-axis support, meaning that heterogeneous allocation can hardly make any difference on the cache hit performance. Conversely, a slightly more pronounced separation is visible along the y-axis, entailing that the number of hops traveled before a cache is hit can instead be affected by the ranking function. Similar considerations hold for DTelekom, where performance gaps are only slightly more noticeable.

Centrality metrics have a larger impact instead on the Tiger, Abilene and Geant topologies, for both cache hit and path stretch metrics: notice indeed that boxes now separate across centrality metrics. Due to the cache hit and path stretch semantic, the bottom-right part of the plot correspond to better performance, so that allocation criteria resulting in bottom-right boxes for all topologies should be preferred. Due to the previously observed correlation in the size of the resulting caches (Tab. II) it is not surprising to observe similarity across allocations of different centrality metrics. Yet, there is no clear winner that stands out from the plot – such as an optimal strategy, furthermore yielding consistently superior results over all[3] topologies.

We stress the Zipf popularity settings may further alter the results for a given topology. We exemplify the situation by taking into account the relative error between the cache hit gained by a quantized allocation strategy induced by metric X over a constant homogeneous allocation, i.e., $(H_X^Q - H_{Const})/H_{Const}$ according to our previous terminology. The relative error is depicted in Fig. 6 for $\alpha = 1.5$ (top) and $\alpha = 1.25$ (bottom) and for all topologies (points) and centrality metrics (x-axis).

It can be seen that, e.g., eccentricity centrality EC may have an opposite behavior depending on whether the popularity is highly skewed or not: apparently, putting more cache capacity to nodes that are somehow "confined" in faraway networks region can be a reasonable choice only for highly skewed content ($\alpha = 1.5$) as it otherwise can yield to cache hit reduction. More generally, even for a single performance metric, it is again hard to find a metric robust under all networks and popularity settings – *with the exception of degree-based $C_{DC}^Q(i)$ allocation.*

Notice indeed that, for both $\alpha = 1.5$ and $\alpha = 1.25$, the DC metrics yield to cache hit improvements for all

[3] We prefer to ensure that gain hold over a larger number of topologies that simply requiring a larger *average* gain, as the latter may hide performance drops for some topologies.

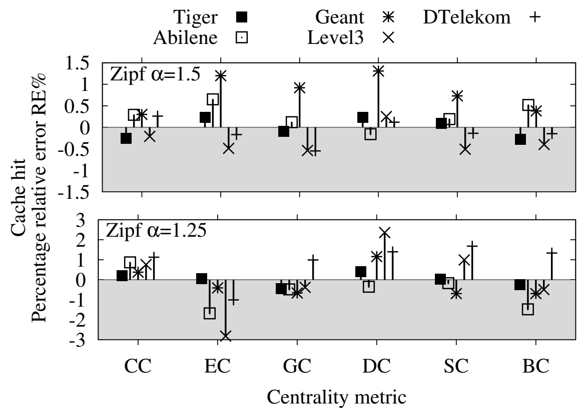


Fig. 6. Cache hit under quantized ranking allocation strategies, for different topologies and Zipf settings. Y-axis reports the percentage relative error $(H_X^Q - H_{Const})/H_{Const}$ with respect to a baseline constant allocation. Dark shaded zone corresponds to a performance loss for heterogeneous allocation.

topologies except Abilene, where the performance loss keeps however very limited (i.e., performance are very close to the baseline constant allocation for Abilene). This can be explained with the fact that, in the Abilene topology, the degree does not significantly vary across nodes, so that a degree-proportional allocation practically degenerate into an (almost) homogeneous allocation. Conversely, in all other cases $(H_{DC}^Q - H_{Const})/H_{Const} > 0$ so that some gain can be gathered from a heterogeneous allocation.

Overall, it seems that a simple quantize degree-based allocation policy C_{DC}^Q is the most robust across all topologies. On the one hand, this is positive, since very simple operational rules of thumb can be defined (e.g. "if you add a line card, add a content store module as well"). On the other hand, we also observe that a degree-based selection may tradeoff with technological constraints, such as the ability to perform memory lookups at line speed *when both the memory size and the node network capacity grows*. As a result, further architectural analysis beyond [2] may be needed to assess whether the quantized degree-based allocation policy is also feasible, and thus relevant.

Finally, from a higher level viewpoint, we notice that performance gain is upper-bounded by a modest 2.5% *in the best case* (Level3 topology, $\alpha = 1.25$ in the bottom plot of Fig. 6). In reason of such a limited gain over homogeneous cache sizes, it seems as there may be no real incentive in using heterogeneous cache allocation policies altogether.

IV. Conclusions

This work contrasts the performance of CCN under homogeneous vs heterogeneous cache sizes. Specifically, we consider that a given amount of memory resources need to be allocated across a network of arbitrarily connected CCN nodes. We then define several allocation criteria resorting to standard graph centrality metrics – such as betweenness, closeness, stress, graph, eccentricity and degree centralities. Our criteria are either simplistic (i.e., proportional to the centrality metrics) or also incorporate a first degree of technological constraints (i.e., memory quantization in multiple of a given amount).

To gather relevant results, we study large cache sizes up to 10 GB, realistic topologies up to 60 nodes, and a YouTube-like Internet catalog consisting of 10^8 files for 1 PB of video data. A thorough simulation campaign allow us to conclude that the simplest metric, namely degree centrality (i.e., allocating memory proportionally to the number of links), proves to be a robust allocation criterion, yielding (modest) cache hit gain under a large number of topologies and popularity settings.

At the same time, we also observe that in reason of the modest gain, there may be no real incentive in using heterogeneous cache allocation policies altogether.

Acknowledgements

This work was funded by the ANR Project *Content-Oriented Networking: a New Experience for Content Transfer (CONNECT)*.

References

[1] ccnSim homepage. http://www.infres.enst.fr/~drossi/ccnSim.

[2] Somaya Arianfar and Pekka Nikander. Packet-level Caching for Information-centric Networking. In *ACM SIGCOMM, ReArch Workshop*, 2010.

[3] G. Carofiglio, M. Gallo, and L. Muscariello. Bandwidth and Storage Sharing Performance in Information Centric Networking. In *ACM SIGCOMM, ICN Worskhop*, pages 1–6, 2011.

[4] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, and Diego Perino. Modeling Data Transfer in Content-Centric Networking. In *ITC*, 2011.

[5] M. Cha, H. Kwak, P. Rodriguez, Y.Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *ACM IMC*, pages 1–14, 2007.

[6] H. Che, Y. Tung, and Z. Wang. Hierarchical web caching systems: Modeling, design and experimental results. *IEEE Journal on Selected Areas in Communications,*, 20(7):1305–1314, 2002.

[7] J. Choi, J. Han, E. Cho, T. T. Kwon, and Y. Choi. A Survey on Content-Oriented Networking for Efficient Content Delivery. *IEEE Communications Magazine*, pages 121–127, 2011.

[8] G. Rossini D. Rossi. Caching performance of content centric networks under multi-path routing (and more). Technical report, Telecom ParisTech, 2011.

[9] Rodrigo Fonseca, Mark Crovella, and Bruno Abrahao. On the Intrinsic Locality Properties of Web Reference Streams. In *IEEE INFOCOM*, 2003.

[10] D. Rossi G. Rossini. A dive into the caching performance of content centric networking. Technical report, Telecom ParisTech, 2011.

[11] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. Youtube traffic characterization: a view from the edge. In *ACM IMC*, pages 15–28, 2007.

[12] Van Jacobson, Diana K Smetters, Nicholas H Briggs, James D Thornton, Michael F Plass, and Rebecca L Braynard. Networking Named Content. In *ACM CoNEXT*, 2009.

[13] Konstantinos Katsaros, George Xylomenos, and George C Polyzos. MultiCache : An overlay architecture for information-centric networking. *Computer Networks*, pages 1–11, 2011.

[14] Nikolaos Laoutaris, Sofia Syntila, and Ioannis Stavrakakis. Meta Algorithms for Hierarchical Web Caches. In *IEEE ICPCC*, 2004.

[15] Stefan Podlipnig and Laszlo B Osz. A Survey of Web Cache Replacement Strategies. *ACM Computing Surveys*, 35(4):374–398, 2003.

[16] Ioannis Psaras, Richard G Clegg, Raul Landa, Wei Koong Chai, and George Pavlou. Modelling and Evaluation of CCN-Caching Trees. *IFIP Networking*, 2011.

[17] Elisha J. Rosensweig, Jim Kurose, and Don Towsley. Approximate Models for General Cache Networks. *IEEE INFOCOM*, pages 1–9, 2010.

[18] N. Spring, R. Mahajan, and D. Wetherall. Measuring isp topologies with rocketfuel. In *ACM SIGCOMM*, 2002.