# Clustered Content Replication for Hierarchical Content Delivery Networks


Lazaros Gkatzikis
Huawei,
France.
lazaros.gkatzikis@huawei.com

Vasilis Sourlas
UCL,
UK.
v.sourlas@ucl.ac.uk

Carlo Fischione
KTH,
Sweden.
carlofi@kth.se

Iordanis Koutsopoulos
AUEB and CERTH,
Greece.
jordan@aueb.gr

György Dán
KTH,
Sweden.
gyuri@kth.se



***Abstract*—Caching at the network edge is considered a promising solution for addressing the ever-increasing traffic demand of mobile devices. The problem of proactive content replication in hierarchical cache networks, which consist of both network edge and core network caches, is considered in this paper. This problem arises because network service providers wish to efficiently distribute content so that user-perceived performance is maximized. Nevertheless, current high-complexity replication algorithms are impractical due to the vast number of involved content items. Clustering algorithms inspired from machine learning can be leveraged to simplify content replication and reduce its complexity. Specifically, similar items could be clustered together, *e.g.,* according to their popularity in space and time. Replication on a cluster-level is a problem of substantially smaller dimensionality, but it may result in suboptimal decisions compared to item-level replication. The factors that cause performance loss are identified and a clustering scheme that addresses the specific challenges of content replication is devised. Extensive numerical evaluations, based on realistic traffic data, demonstrate that for reasonable cluster sizes the impact on actual performance is negligible.**



***Keywords*—cache, clustering, content replication, Radio Access Network.**


## I. Introduction

Content Delivery Networks (CDNs) are responsible for 36% of the Internet traffic [1]. In order to efficiently deliver such vast amounts of data traffic, CDN providers deploy cache servers worldwide, where the most popular content can be replicated. Each content request is redirected to the closest replica rather than being served by the origin CDN server. Thus, replication facilitates local servicing of content requests, and consequently both the user Quality of Experience (QoE) is improved and the core network traffic is minimized. In parallel, a 10-fold increase of mobile devices traffic is expected by 2018 [1]. Thus, it has been proposed that also Radio Access Network (RAN), namely base stations, should be enhanced with caching capabilities.

The problem of optimal content replication and placement in a network with distributed caches has received significant interest lately [2], [3]. Existing caching schemes assume that caching is performed at item-level. However, given the trillions of items transferred over CDNs, deriving at each time the cache placement is a task of prohibitive computational complexity. Thus, some form of content aggregation is the only viable solution. The dimensionality of the problem is further amplified when fragmentation of items into equally sized chunks is applied, which is a requirement of many replication mechanisms [2], [4].

Ideas from data mining can be used to reduce the complexity of content replication algorithms. Specifically, clustering of items into groups based on a similarity metric can significantly reduce the input size (dimension) of the replication problem, and hence it may enable the application of existing replication schemes in our context. However, this comes at the cost of reduced performance, namely QoE, compared to the fine-grained but impractical item-level replication. Thus, the derivation of the optimal clusters in terms of size and contents is a challenging task that needs to address this inherent tradeoff.

In this paper, we consider the interaction of content clustering and replication in a hierarchical network of caches, covering both RAN- and CDN-level, as the one depicted in Fig. 1. Any request that cannot be satisfied locally results in a cache miss and is forwarded to the next higher layer, which introduces additional communication delay. Due to the high complexity of replication at item granularity, *we apply clustering of items and replication of clusters*. Thus, clusters can be updated offline during periods of low traffic, *e.g.,* during the night, based on popularity statistics that have been already collected, whereas cache contents can be updated more frequently.

### A. Related work

The problem of content caching in CDN and RAN-level was considered in [5], where no coordination among caches is assumed, *i.e.,* each cache decides its content independently. The offline policy of caching the most popular items and in-network caching policies, similar in rational to Least Recently Used (LRU), are considered. In contrast, we address the feasibility of coordinated content replication by clustering content items based on their popularity.

Coordinated replication and placement of content in a set of caches so as to optimize network performance, *e.g.,* minimize average latency, is an NP-hard problem [6]. However, several caching schemes of polynomial complexity have been proposed in the literature [2], [4], [7], [8]. For example, work [4] demonstrates that for generic topologies an $e/(e-1)$-approximation algorithm of complexity $O(N^8V^8)$ can be derived, where $N$ is the number of content items and $V$ the number of caches. Given that an extremely large number of items $N$ is circulated over CDNs and that our system consists of numerous caches $V$, even such approximation algorithms of polynomial complexity cannot be directly applied. Instead, we propose the use of clustering so as to reduce the dimensionality

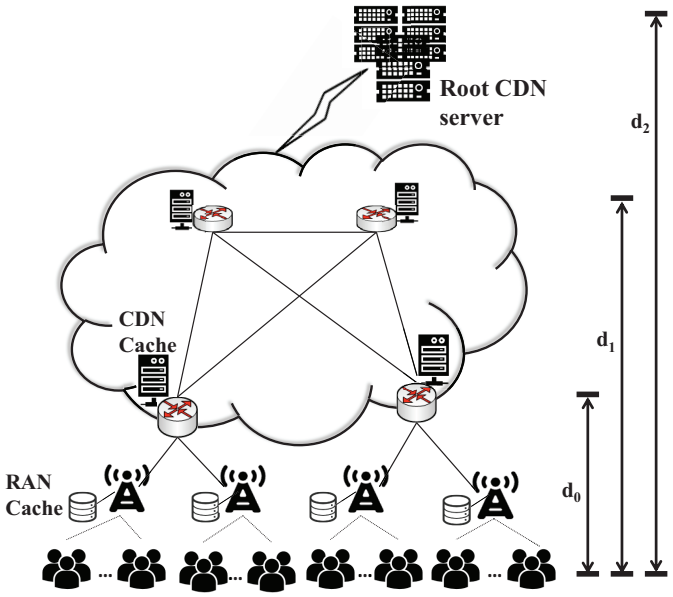



Fig. 1. Architecture of the hierarchical-cache network.

of the problem. An overview of existing clustering schemes can be found in [9].

Clustering of web content based on popularity and replication at cluster-level was first considered in [10]. The validity of such an approach was demonstrated through extensive numerical evaluations of existing clustering and replication algorithms. In particular, the $K$-split algorithm was used to minimize the maximum distance within a cluster. Clustering for replication purposes has also been considered in the context of grid computing in [11]. Items are clustered together whenever they are frequently accessed by the same process within a small period of time. The problem of clustering is cast as a graph partition problem, and a greedy algorithm is proposed. Once the clusters have been determined, the problem is cast as an integer linear programming (ILP) instance, which is solved numerically.

The aforementioned schemes deal with clustering and replication as two independent problems. Clustering of content is performed according to existing clustering schemes and replication of clusters follows. Although existing schemes provide tangible evidence that content clustering is a promising approach, a clustering scheme addressing the specifics of content replication is still missing. In this direction, we suggest that clusters should be formed such that performance loss induced by replication at cluster-level is minimized.

An alternative use of clustering is proposed in [12], where the interaction of user association and caching at RAN-level is considered. In particular, users are clustered together and associated to small cells based on similarity of their content requests. Such an approach is complementary to our work, given that our scheme relies on aggregate content popularity estimates.

### B. Our contribution

The key contributions of this paper are as follows:

- We identify the *spatial distribution* of requests as the most prominent feature of content items in order to aid replication.
- We formulate the joint clustering and replication problem and we analyze explicitly the inherent complexity - performance tradeoff.
- We propose a replication-aware clustering scheme and different metrics for the calculation of content similarity that capture the spatial diversity of content popularity.
- We use realistic traffic data to analyze the impact of cluster size on actual performance.

The rest of the paper is organized as follows. In Section II, we present the system architecture and formulate the problem of joint clustering and replication. In Section III, the problem is decomposed into a replication-aware clustering scheme and an adaptive replication strategy. Numerical results quantifying the performance loss caused by clustering are presented in Section IV. Finally, Section V concludes our study and provides pointers for future work.

Throughout the paper we will use calligraphic letters to denote sets and the corresponding capitals for cardinality; for example $|\mathcal{V}| = V$.

## II. System Model and Problem Formulation

We consider the hierarchical-cache network depicted in Fig. 1, which can be represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Let $\mathcal{V}$ denote the set of caches and $\mathcal{E}$ the set of communication links connecting them. Root $v_0$ corresponds to the origin CDN server where all content items are stored, the first layer consists of CDN caches and each CDN cache is directly accessible by a set of RAN caches. Let $C_v$ be the storage capacity (in bits) of cache $v \in \mathcal{V}$.

Let $\mathcal{N}$ denote a given fixed set of $N$ content items that have to be delivered over the network and $s^n$ the size (in bits) of item $n$. Content requests are generated by users, with each user accessing the network through a RAN node. Requests that can not be served locally are forwarded to higher cache layers. As a result, content requests are generated with rate $\boldsymbol{r}_v = \{r_v^1, \ldots, r_v^N\}$, where $r_v^n$ denotes the aggregate incoming request rate (in requests per second) at cache $v$ for item $n$. Vector $\boldsymbol{r}_v$ is an estimate of the actual request pattern based on observed, historical content access data (within a given time window). This estimate is used as a prediction for the future number of requests addressed to each cache.

Access requests that cannot be satisfied locally, trigger a procedure for the transfer of the requested item from a remote cache. Thus, in terms of traffic, a request for item $n$ generated at node $i$, would incur a cost equal to $s^n d_{ij}$, if served by node $j$. Alternatively, parameter $d_{ij}$ could capture the latency of accessing content that is located in a distant node $j$. In this work, we consider latency as the performance metric of interest and we assume that it is symmetric *i.e.,* $d_{ij} = d_{ji}$.

Clustering of $N$ content items into $M$ clusters (*i.e.,* groups), with $M \ll N$, facilitates replication at cluster-level. In addition, clustering can tune the complexity of replication by determining the exact number of clusters to be formed. Here, we consider the scenario where replication complexity imposes a hard constraint on the maximum number of clusters

$\bar{M}$ that can be supported, *i.e.,* $M \leq \bar{M}$. Parameter $\bar{M}$ is a positive integer that is determined by $i$) the time required to calculate a cache assignment and $ii$) the dynamics of aggregate request pattern $\boldsymbol{r}_v$. The former depends on the complexity of the replication algorithm applied, whereas the latter is the period of time in which estimated request rate vectors can be considered sufficiently accurate.

Let binary matrix $\boldsymbol{X} \in \{0,1\}^{N \times M}$ denote clustering decisions, such that $x_{nm} = 1$ if item $n$ is assigned to cluster $m$. Similarly, matrix $\boldsymbol{Y} \in \{0,1\}^{M \times V}$ denotes caching decisions $y_j^m \in \{0,1\}$ on whether cluster $m$ should be cached at node $j$. Let $S^m = \sum_{n \in \mathcal{N}} x_{nm} s^n$ be the size of cluster $m$ and $R_i^m = \sum_{n \in \mathcal{N}} x_{nm} r_i^n$ be the total request rate for items of cluster $m$ at node $i$. Notice that we use the small letter to denote the item-level parameters and the corresponding capital letters to denote the cluster-level parameters. The problem of joint clustering and caching towards minimizing average latency can be formally stated as

$$\min_{\boldsymbol{X},\boldsymbol{Y},M} \quad \sum_{m=1}^{M} \sum_{j \in \mathcal{V}} \sum_{i \in \mathcal{A}_j^m} R_i^m d_{ij} \tag{1}$$

$$\text{s.t.} \quad x_{nm} \in \{0,1\} \qquad \forall n \in \mathcal{N}, m = 1 \dots \bar{M}, \tag{1a}$$

$$\sum_{m=1}^{M} x_{nm} = 1 \qquad \forall n \in \mathcal{N}, \tag{1b}$$

$$M \leq \bar{M}, \tag{1c}$$

$$\sum_{m=1}^{M} y_j^m S^m \leq C_j \qquad \forall j \in \mathcal{V}, \tag{1d}$$

$$y_j^m \in \{0,1\} \qquad \forall j \in \mathcal{V}, m = 1 \dots \bar{M}, \tag{1e}$$

where $\mathcal{A}_j^m$ is the set of nodes retrieving items of cluster $m$ through its replica at cache $j$, constraints (1a)-(1c) correspond to assignment of items to clusters. Constraint (1b) ensures that clusters are disjoint, whereas constraint (1d) captures the limited capacity of each cache.

Notice that relaxing constraint on the maximum number of clusters (1c) results in the well known NP-hard problem of item-level replication [6]. Thus, Problem (1) is NP-hard.

## III. Efficient Algorithms for Replication-aware Clustering and Caching

To deal with the NP-hardness of optimal cluster-level replication and the extreme number of RAN caches, we propose that the original Problem (1) should be decomposed into two subproblems: $i$) an in-network caching problem at RAN caches and $ii$) a replication-aware clustering subproblem, where the derived clusters can be subsequently replicated at CDN caches. Any cache miss at RAN is forwarded and served by the closest CDN cache.

The proposed decomposition of the original problem is also motivated by various practical constraints. First, capacity of RAN caches is significantly smaller than that of CDN caches. Thus, cooperative clustering and caching at CDN and RAN level would significantly limit the size of clusters that can be supported, which is implicitly determined by constraint (1d). Second, given also the limited number of caches that can be handled due to complexity constraints [4], the joint consideration of the two cache layers could lead to degraded performance since a lower-complexity algorithm would have to be applied. Finally, RAN and CDN are generally operated by different entities, which would make coordination extremely difficult. Thus, we propose that cluster-level caching should be applied only at CDN caches, whereas item-level opportunistic caching is a suitable option for the RAN caches.

We assume that each RAN cache applies a Least Frequently Used (LFU) caching strategy, where in case of a full cache the item that has been accessed the least is evicted. Thus, the incoming request rate at CDN node $j$, $\boldsymbol{r}_j$, is the result of cache misses. On the CDN side, clustering requires first to identify the features that differentiate content items regarding replication. In our scenario, content popularity is the main feature. Notice that according to the problem definition (1), replication decisions are also dependent on the corresponding costs $d_{ij}$. However, these are characteristics of the underlying network and hence cannot be considered as features. Instead, we propose that the impact of this communication cost should be addressed by the cluster replication algorithm.

### A. Characterizing performance loss due to clustering

In traditional clustering problems, once the features have been identified, a set of representatives has to be selected so that a loss function, *e.g.,* average distance of cluster items from the closest representative, is minimized. Instead, in the replication scenario considered here, we are only interested in how clustering of items translates into suboptimal replication decisions. Next, we present the two reasons that cause performance loss in comparison to item-level replication.

• **Slack loss: Non-integral multiple of cluster size.** This loss arises when part of the storage capacity of a cache remains unallocated, since no uncached cluster fits there. If item-level caching was applied instead, a subset of the items of a cluster would have been cached. Unallocated space is upper-bounded by the size of the smallest uncached cluster. The corresponding loss is equal to the cost of fetching those items from the closest replica.

• **Diversity loss: Miss-classification of items due to spatial variation of popularity and coarse-grained replication.** This loss results from two main characteristics of the specific application scenario, namely spatial variation of content popularity and diversity of latency costs $d_{ij}$. Content clustering is performed according to the request pattern $\boldsymbol{r}_j$ across all caches. Thus, any two items that are clustered together in the same cluster, must be cached together. There may be places (caches) though, that it would be preferable to split a cluster, so as to cache only a part of it along with other items. Diversity loss is generally decreasing in the number of clusters $M$.

### B. Replication-aware clustering of content items

The problem of clustering a set of items in $M$ clusters is generally NP-hard [13]. Next, we derive a low complexity clustering scheme that adequately addresses the aforementioned losses. In order to address "Slack loss" we suggest that all clusters should be of equal size, *i.e.,* $S^m = S \ \forall m$, and cluster size $S = \sum_{n \in \mathcal{N}} s^n / M$ should be a common divisor

of all server capacities. Thus, we introduce the following set of constraints on the selected number of clusters

$$M\frac{C_j}{\sum_{n\in\mathcal{N}} s^n} \in \mathbb{N} \quad \forall j \in \mathcal{V}. \tag{2}$$

This constraint ensures that clusters perfectly fit to caches and hence no cache capacity remains unused. Any approximate divisor could also be used leading to limited loss. The imposed set of constraints, along with constraint (1c), eventually determine the number of clusters $M$. Notice that in CDNs all servers can be considered of equal capacities and hence (2) reduces to a single constraint. Given that optimal clustering is NP-hard, it has been shown that introducing specific constraints in cluster cardinality generally improves performance of heuristic approaches [14]. Introducing this constraint facilitates also the application of swapping-based replication algorithms which require all items to be of equal size [7], [8]. On the other hand, "Diversity loss" is an inherent characteristic of clustering and hence cannot be totally avoided. Nevertheless, we propose a clustering approach that pursues to minimize the corresponding loss.

For a given feature set, a similarity index $I$ (metric) has to be derived so as to cluster items together. In its simplest form, the overall popularity of each item $p^n = \sum_{j\in\mathcal{V}} r_j^n$ can be considered as a single dimensional feature for clustering. In this case, similarity of any two items $n_1$ and $n_2$ is captured by their absolute distance as

$$I_{\text{pop}}(n_1, n_2) = \left(|p^{n_1} - p^{n_2}|\right)^{-1}. \tag{3}$$

Generally, calculating content similarity over a more detailed feature set captures content characteristics more precisely and enables us to address spatial variation of request rates $\boldsymbol{r}_j$. In this direction, we consider a similarity metric based on pairwise Euclidean distance, *i.e.,*

$$I_{\text{eucl}}(n_1, n_2) = \left(\sqrt{\sum_{j\in\mathcal{V}} \left(r_j^{n_1} - r_j^{n_2}\right)^2}\right)^{-1}. \tag{4}$$

Finally, we consider a more conservative similarity metric, which is based on Chebychev distance, *i.e.,* the maximum distance of content popularity over all coordinate dimensions

$$I_{\max}(n_1, n_2) = \left(\max_{j\in\mathcal{V}} |r_j^{n_1} - r_j^{n_2}|\right)^{-1}. \tag{5}$$

The aforementioned pairwise similarity metrics can be then used to construct clusters. Although any of the existing similarity-based clustering algorithms could be applied, here we devise a greedy clustering scheme that specifically addresses both Slack and Diversity loss. Initially, a cluster that contains the two most similar items is formed. Then, more items are added to the cluster one by one until the selected size of cluster according to constraints (1c) and (2) is reached. Each item assigned to a cluster is excluded from the candidate set. Next, similarity of each item with the set of items already clustered is calculated as the average distance over all of them. Thus, clusters are created and filled sequentially until all the items have been assigned to a cluster.

### C. Cluster-level replication

Once the clusters have been determined, they have to be optimally replicated over the CDN caches. The proposed clustering scheme guarantees that any of the existing item-level replication schemes can be directly applied. In this work, we assume that 2-approximation algorithms such as greedy placement [15] or the lower complexity swapping algorithm of [7], [8] are applied. Notice that the performance guarantees of existing item-level approximation algorithms hold also in the cluster domain, since here clusters can be simply considered as items of larger size and higher popularity.

The greedy replica placement algorithm initially assumes empty caches and at each iteration replicates the cluster to the cache that yields the maximum latency gain. In particular, in the first round the algorithm evaluates the latency gain if each of the $M$ clusters is cached in each of the $V$ caches. Out of the $MV$ available options, the cluster-cache pair that yields the maximum latency gain is selected. Given the previous step decision, in the second round, an additional cluster-cache pair has to be selected, namely the one that yields the maximum latency savings. The greedy algorithm is repeated until all the available storage capacity has been used and is characterized by a complexity of $O(M^2V^2)$. Thus, if $M = N/100$, clustering reduces replication complexity by four orders of magnitude.

An alternative approach would be to apply an iterative algorithm inspired by hierarchical clustering. Initially, two clusters are formed and latency under greedy replication is calculated. Next, the clusters are split such that constraint (2) is not violated and performance is calculated for the new clusters under greedy replication. Such an approach does not require the number of clusters to be predetermined and it is repeated as long as the latency improvement is above a predefined threshold. Notice that this approach comes at the cost of increased execution complexity, since multiple replication problems have to be solved. Due to limited space we do not elaborate on this approach.

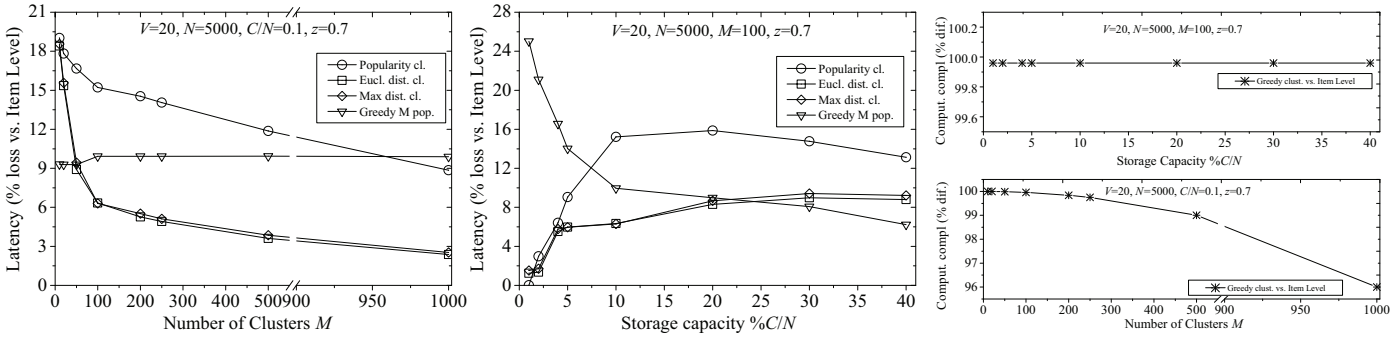
## IV. Numerical Evaluation

### A. Evaluation setup

We use simulations to evaluate the performance of the proposed clustering scheme, when combined with greedy replication of the resulting clusters. We also consider the performance of item-level greedy replica placement (mentioned in figures as *item-level*), so as to quantify the performance loss arising from clustering. Greedy replication algorithm initially assumes empty caches and in each iteration caches the item to the cache that yields the maximum latency gain [15].

Throughout this section, we assume that all items are of unit size $s^n = s = 1, \forall n \in \mathcal{N}$ and we consider a hierarchical-cache network with $V = 20$ (CDN caches), where all caches have the same storage capacity ($C_v = C$, $\forall v \in \mathcal{V}$) and hence each can hold up to $C$ different unit sized items.

As shown in Fig. 1 we assume three layers of caching. If the requested content exists in the directly accessible RAN cache, zero latency is induced. We denote by $d_0$ the average latency of serving a request by the user's closest CDN cache, whereas $d_1$ denotes the average latency of serving a request




(a) (b) (c)

Fig. 2. The performance of the proposed clustering schemes vs. the total number of clusters and the the storage capacity of each cache.

from a peer cache in the given CDN network. Finally, $d_2$ represents the average latency of fetching contents from the origin CDN server, where all content items are stored (root $v_0$). Similarly to [16], when the requested content is not cached in the closest CDN cache, it is fetched from a peer cache in the same administrative domain, whereas the root CDN server might be outside this domain.

Generally, condition $d_0 < d_1 < d_2$ holds. Typical latency values are 10-30 ms for $d_0$ in cable and ADSL access networks. The latency between caches in the same administrative domain, $d_1$, typically ranges from a few up to 20 ms larger than $d_0$, depending on the geographical coverage of the network. Finally, $d_2$ typically ranges from $100-200$ ms. Throughout our simulations, we assume that $d_{ij} \in \{25, 50, 100\}$ ms, depending on the type of caches $i$ and $j$.

Request rates for content items at each node are determined by their *popularity*. We approximate item popularity by a Zipf law distribution of exponent $z$, since it has been shown that file popularity in the Internet follows the Zipf distribution [17]. In general, the popularity of each content item may differ from place to place, due to locality of interest (a.k.a. *spatial locality* or *spatial skew*). In our experiments, this is captured through a localized request generation model, where aggregate request pattern $\boldsymbol{r}_v$ is different across locations $v$. We assume $V$ different regions each served by a CDN cache. All regions are characterized by the same value for the Zipf distribution exponent which captures the local popularity of items. In each location the ranking/order of the items within the Zipf distribution may be different, given that different items are cached in each of the RAN caches. The latter captures the impact of RAN-level caching. We assume that in each region a total of 10 requests per second are generated. Thus, the request rate for each item at each region varies from 0-10 reqs/sec depending on item's popularity and ranking. We consider a scenario where $N = 5000$ content items have to be clustered in $M$ clusters.

### *B. Quantifying the impact of number of clusters and cache storage capacity*

First, we study the performance of the proposed clustering scheme for each of the similarity metrics. Fig. 2(a) depicts the impact of the number of clusters $M$ on latency. We observe that clustering based on Euclidean and Max relative distance outperform Popularity-based clustering. The former two perform only $3\% - 18\%$ worse than item-level replica placement algorithm, requiring at the same time $96\%$ less computations for the completion of the replica assignment as depicted in Fig. 2(c). On the other hand, popularity-based clustering exhibits significantly worse performance even for a large number of clusters ($\approx 9\%$ worse than item-level), since this scheme does not capture the spatial variation of content, but only the aggregate request rate for each item over the entire network. The tradeoff of performance over computational complexity is definitely against item-level replication, since in most cases it requires more than $96\%$ additional computations so as to achieve a negligible latency improvement of $3\% - 6\%$.

In Fig. 2(b) we depict the impact of available storage capacity, expressed as the fraction of content items that can be stored in a CDN cache. We observe that location-aware similarity metrics always perform better than Popularity-based clustering and at most $8\%$ worse than item-level replication. In the most realistic regime of small enough storage capacity ($1\% - 10\%$ of all items), they perform even better ($\approx 5\%$ worse than item-level) having 99.96 less computational complexity, as depicted in Fig. 2(c).

From the results above, it should be clear that the computational complexity of item-level replication is enormous when compared to cluster-level replication. An alternative approach of comparable complexity to the proposed scheme would be an item-level greedy replication considering only the $M$ most popular items (denoted as *Greedy $M$ pop.* in the figures). In particular, the $M$ network-wide most popular items are placed according to the item-level greedy replication algorithm and the remaining storage capacity of each cache is used to store the locally most popular items. *Greedy $M$ pop*. algorithm is in line with the coordinated caching scheme used in [16], where a portion of the available cache is used in a coordinated manner and the rest is used to cache the locally most popular items.

Fig. 2 reveals that the *Greedy $M$ pop.* algorithm performs better than the Popularity-based clustering, but leads to $2 - 3$ times higher performance loss in comparison to the proposed clustering scheme that exploits the spatial characteristics of content popularity. It is only for large enough storage capacity that *Greedy $M$ pop.* outperforms cluster-level replication. In this case, all $M$ items fit in each cache and hence are replicated in each of them.

### *C. Impact of content popularity*

In the above scenarios we assumed a specific value for the Zipf exponent of the items' popularity. Measurement-based

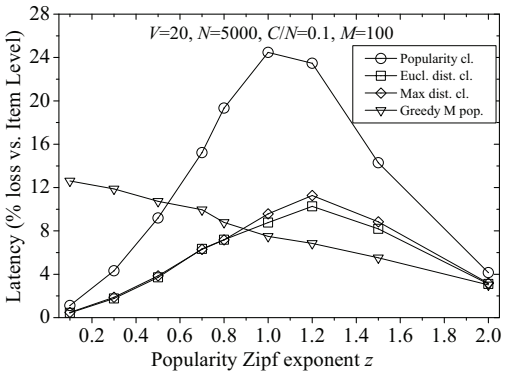



Fig. 3. The performance of the proposed clustering schemes vs. the exponent of content popularity.

studies suggest that the Zipf exponent for web traffic lies in the range of $0.64 - 0.84$, while other types of traffic (*e.g.,* P2P or video) may follow different popularity patterns [17]. Since in this work we do not assume a specific application, we depict in Fig. 3 a wider range of values for the Zipf distribution. As expected, exploiting spatial variation of content popularity is of utmost importance and it results into a latency increase of $10\%$ over item-level replication. On the other hand, Popularity-based clustering may result up to a $24\%$ latency increase, for large values of parameter $z$, since in such scenarios the proposed clustering scheme groups together items that are very popular, but they have large spatial skewness, resulting in higher Diversity loss.

On the other hand, *Greedy $M$ pop.* algorithm is outperformed by clustering schemes for small values of $z$ and only when $z > 1$ is the situation reversed. When $z > 1$ the $M$ most popular items account for the largest portion of requests and thus Greedy $M$ pop. algorithm outperforms cluster-level replication. In addition, when $z > 1$ and due to the different popularity ranking of the items among the nodes of the network, items tend to have large spatial skewness and the composed clusters have large intra-clustering distance increasing the Diversity loss as well.

## V. Conclusions

This work is a first step towards a better understanding of the interaction of clustering and replication in hierarchical architectures that bring together caching at CDN- and RAN-level. We devised a clustering scheme that takes into account the specific characteristics of content replication. Our analysis demonstrated that applying the proposed scheme, if content is clustered according to spatial distribution of popularity, can efficiently reduce replication complexity up to four orders of magnitude at the cost of less than $5\%$ performance degradation in comparison to item-level replication.

In this paper, we focused mainly on the impact of spatial distribution of content popularity in a static scenario. Content popularity though may exhibit significant temporal variations as well. Thus, the temporal dimension of the problem, *i.e.,* how often clustering and replication decisions should be updated, is an interesting topic for future study. In addition, stream clustering approaches could be considered for the online dynamic update of clusters when new popular items appear in the network in a streaming fashion.

## Acknowledgment

This work was conducted while L. Gkatzikis was a research associate at KTH Royal Institute of Technology. V. Sourlas work has been supported by the European Commission through the FP7-PEOPLE-IEF INTENT program, under contract 628360. I. Koutsopoulos acknowledges the support of the ERC08-RECITAL project, co-financed by Greece and the European Social Fund through the Education and Lifelong Learning Operational Program of the Greek National Strategic Reference Framework 2007-2013.

## References

[1] "Cisco visual networking index: Forecast and methodology," http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.

[2] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *IEEE INFOCOM*, 2010, pp. 1–9.

[3] S. Gitzenis, G. Paschos, and L. Tassiulas, "Enhancing wireless networks with caching: Asymptotic laws, sustainability and trade-offs," *Computer Networks*, vol. 64, no. 0, pp. 353 – 368, 2014.

[4] K. Shanmugam, N. Golrezaei, A. Dimakis, A. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[5] H. Ahlehagh and S. Dey, "Hierarchical video caching in wireless cloud: Approaches and algorithms," in *IEEE ICC*, 2012, pp. 7082–7087.

[6] I. D. Baev and R. Rajaraman, "Approximation algorithms for data placement in arbitrary networks," in *ACM-SIAM Symposium on Discrete Algorithms*, 2001, pp. 661–670.

[7] V. Sourlas, P. Flegkas, L. Gkatzikis, and L. Tassiulas, "Autonomic cache management in information-centric networks," in *IEEE NOMS*, 2012, pp. 121–129.

[8] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas, "Distributed cache management in information-centric networks," *IEEE Transactions on Network and Service Management*, vol. 10, no. 3, pp. 286–299, 2013.

[9] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *Journal of Intelligent Information Systems*, vol. 17, no. 2-3, pp. 107–145, 2001.

[10] Y. Chen, L. Qiu, W. Chen, L. Nguyen, and R. Katz, "Efficient and adaptive web replication using content clustering," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 979–994, 2003.

[11] H. Sato, S. Matsuoka, and T. Endo, "File clustering based replication algorithm in a grid environment," in *9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, 2009, pp. 204–211.

[12] M. ElBamby, M. Bennis, W. Saad, and M. Latva-aho, "Content-aware user clustering and caching in wireless small cell networks," in *Wireless Communications Systems (ISWCS)*, 2014, pp. 945–949.

[13] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hardness of Euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.

[14] S. Zhu, D. Wang, and T. Li, "Data clustering with size constraints," *Knowledge-Based Systems*, vol. 23, no. 8, pp. 883 – 889, 2010.

[15] J. Kangasharju, J. Roberts, and K. W. Ross, "Object replication strategies in content distribution networks," *Comput. Commun.*, vol. 25, no. 4, pp. 376–383, 2002.

[16] Y. Li, H. Xie, Y. Wen, and Z.-L. Zhang, "Coordinating in-network caching in content-centric networks: Model and analysis," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2013, pp. 62–72.

[17] G. Dán and N. Carlsson, "Power-law Revisited: Large Scale Measurement Study of P2P Content Popularity," in *9th IPTPS*, 2010.