Socially-Aware Multimedia Content Delivery for the Cloud

Irene Kilanioti
Department of Computer Science,
University of Cyprus,
Nicosia, Cyprus,
e-mail: ekoila01@cs.ucy.ac.cy

George A. Papadopoulos
Department of Computer Science,
University of Cyprus,
Nicosia, Cyprus,
e-mail: george@cs.ucy.ac.cy

***Abstract*—Most Content Delivery Networks (CDNs) are operated as a Software as a Service (SaaS): Many cloud providers build their custom CDNs to benefit from content users, as well as reduce demand on their own telecommunications infrastructure. More importantly, though, CDNs contribute to cloud adoption, as they can address network problems of cloud computing. With multimedia content providers requiring CDN services to enable the delivery of bandwidth-demanding media to end-users, and the growth of HTTP traffic due to media files circulating over Online Social Networks (OSNs), a social-awareness mechanism over a CDN becomes essential, to mitigate the considerable weight placed on bandwidth. A social awareness mechanism augmented to a stand-alone CDN traffic simulator addresses the issue of which content will be copied in the surrogate servers of a CDN infrastructure and to what extent. Hence, it ensures an optimized content diffusion placement. Herein, we further address the issue of temporal diffusion, related to the most efficient timing of the content placement. We exploit the knowledge of peak times for upload and download, so that content is prefetched in the hours with less traffic. We also incorporate other contextual information, such as the viewership within the media service, to ensure performance optimization. Our variations are experimentally proven to contribute toward maximization of CDNs' performance and minimization of content replication costs.**

***Keywords*-Cloud Computing tools, delivery networks and services, Cloud Applications: social networks, Big Data and Analytics, Social Video Sharing, Social Cascading, YouTube, Twitter, Internet Measurements, Content Delivery Networks**

I. Introduction

CDNs, often operated as SaaS in cloud providers (Amazon CloudFront, Microsoft Azure CDN, etc.) aim at addressing the problem of smooth and transparent content delivery. A CDN actually drives cloud adoption through enhanced performance, scalability and cost reduction. With the limitation for both CDNs and cloud services being the geographic distance between a user asking for content and the server where the content resides, cloud acceleration and CDN networks are both complementary to achieving a goal of delivering data in the fastest possible way. Although cloud mainly handles constantly changing and, thus, not easily cached dynamic content, utilization of CDNs in cloud computing is likely to have profound effects on large data download [1].

The general principle of CDNs is to replicate data dynamically in various places of the world as near as possible to the user that consumes it. They are, however, very dissimilar in terms of the services provided and their geographic coverage. The optimization of their overall efficiency, as far as user is concerned, is practically achieved with the automatic detection of the medium (either computer or mobile -smartphone / tablet-), optimised management of the browser cache, server load-balancing, the consideration of specific nature of the content of the media provider (video content may include video on demand, live videos, geo-blocked content, etc.) or features of certain operators, such as real-time compression, session management, etc.

In a manner that is complementary to the above, they address in general the major issues of (i) the most efficient placement of surrogate servers in terms of high performance and less infrastructure cost; (ii) the best content diffusion placement, namely the decision of which content will be copied in the surrogate servers and to what extent; and (iii) the temporal diffusion, related to the most efficient timing of the content placement [2].

Extended use of OSNs [3], [4], [5], and the increasing popularity of streaming media are the factors that determine the HTTP traffic growth [6]. The amount of traffic generated on a daily basis by online multimedia streaming providers is multiplied by the transmission over OSNs (with more than 400 tweets per minute including a YouTube video link [7] being published per minute). Hence, CDN users can benefit from an incorporated mechanism of social-awareness over the CDN infrastructure. In [2] Kilanioti incorporates a dynamic mechanism of preactive copying of content to an existing validated CDN simulation tool and proposes an efficient copying policy. The latter can be based on prediction of demand in social networks.

A. Why is our Approach Necessary: An Example

Let us consider Bob, located in London and assigned to the London CDN servers of an OSN service. Most of Bob's social friends are geographically close to him, but he also has a few friends in Europe and Australia assigned to their nearest servers. Bob logs into the OSN and posts a video that he wants to share. Pushing the video content to all other

978-0-7695-5697-0/15 $31.00 © 2015 IEEE
DOI 10.1109/UCC.2015.48

geographically distributed servers immediately before any requests occur would be the naive way to ensure that this content is as close as possible to all users. Aggregated over all users, pushing can lead to traffic congestion, and users would experience latency in accessing the content, which, moreover, could not be consumed at all. The problem of caching would be intensified when Alice, the only friend of Bob in Athens, would be interested in that content, and with many such Alices in various places.

Rather than pushing data to all surrogates, we can proactively distribute it only to friends of Bob likely to consume it and only at the time window that signifies a non-peak-time for the upload in London area and a non-peak-time for the download in Athens area, thus taking advantage of the timezone differences of our geo-diverse system. The content will be copied only under certain conditions (content with high viewership within the media service, copied to geographically close timezones where the user has mutual friends with high influence impact). This would contribute to smaller response times for the content to be consumed (for the users) and lower bandwidth costs (for the OSN provider).

B. Contributions

Herein, we perform experiments over a large corpus of YouTube videos. We use Twitter, one of the most popular OSNs centered around the idea of spreading information and propagating it via retweeting across multiple hops in the network [8]. This work extends the Social Prefetcher algorithm [2] to include information about peak-time of various timezones of our geo-diverse system, as well as contextual information about the viewership of video content within the media service. It implements extensions in two variations and incorporates them in a validated simulator for CDNs. A multitude of experiments shows improved metrics for performance measurement over content delivery.

A real dataset of User Generated Content (UGC) is used. It includes multimedia links over an OSN platform, thus social cascades are directly analyzed. Real restrictions of a CDN infrastructure (storage issues, network topology) are taken into account. The proposed algorithm also suggests a mechanism that overcomes the testing limitations of other existing CDN platforms, that either treat CDN policies as black boxes or need third users for experimentation.

Experimentation is conducted on a Twitter dataset containing geographic locations, follower lists and tweets for 37 million users, spreading of more than one million YouTube videos over this network, a corpus of more than 2 billions messages and approximately 1.3 million single messages with an extracted video URL. The wide popularity and massive user base of YouTube and Twitter allow us to obtain safe insights regarding user navigation behavior on other similar media and microblogging platforms, respectively.

In terms of performance, comparison with similar implementations such as [9] is not directly feasible. We note, though, that with the proposed policy, there is a significant improvement over their respective improvement (30%) in pull-based methods, that are employed by most CDNs. We also use a more refined topology of data centers and take storage issues into account. Despite the reductions in storage costs that cloud computing advancements have caused, storage costs still remain an important factor that can be reduced under certain conditions.

As for the main findings of our work, they can be exploited for future policies complementary to existing CDN solutions or incorporated to OSN providers mechanisms, to handle larger scale data. In this work we examine which parameters (number of timezones examined, time threshold duration) affect the CDN metrics the most. The optimization of our algorithm is proved in [2], whereas herein the incorporation of peak hours and popularity of circulating objects information are examined to furthermore enhance its performance.

The remainder of this paper is organized as follows. Section II reviews previous related work. Section III formally describes the addressed problem. The proposed algorithm is described in Section IV. Section V gives an outline of the methodology, along with the preparation of the employed datasets. Our main findings are presented in Section VI. Section VII concludes the paper and discusses directions for future work.

II. Related Work

The algorithm [2] gives a near-optimal solution to the problem of content delivery and addresses memory usage issues related to the very large graph dataset accommodated. Efforts to incorporate the information extracted from OSNs in the way that users share content have various research goals: the decision for copying content, improvement of policy for temporary caching, etc. These goals along with phenomena related to bandwidth-intensive media content and its outspread via OSNs, as well as measurement studies on OSNs that support CDN infrastructure decisions for replicating the content, are described in [10]. Other systems that leverage information from OSNs include [11], [9], and [12]. Traverso et al. in [9] improve QoS by exploiting time differences among sites and the access patterns that users follow. Rather than naively pushing UGC immediately, and unnecessarily contributing to a traffic spike in case the content is not consumed, the system follows a pull-based approach, when the first friend of a user in a Point of Presence (PoP) asks for the content. It also considers the traffic peaks of the regions where the user and the friend are located.

In [2] the parameters of a CDN infrastructure are taken into account and heuristics introduced from more recent works are applied. A real dataset from multimedia links spread over an OSN platform is used to directly analyze social cascades and access to social profiles is not conducted

via a third-party page [11]. The proposed algorithm suggests a mechanism added to a CDN simulator that overcomes the testing limitations of other existing CDN platforms, such as the blackbox treatment of CDN policies or the need for the participation of third users.

Another branch of the bibliography studies users' behaviors in different media services. The traffic characterization of YouTube is described in several studies ([13], [6], [14], [15], [16]), with emphasis on the characteristics of YouTube content, such as file size, bitrate, usage patterns and popularity. In [15], the authors study the YouTube workload to discover that there are many similarities between traditional Web and media streaming workloads. The authors in [17] find a strong correlation among YouTube videos, because the links to related videos generated by uploaders depict small-world characteristics. In [18] the authors analyze how the popularity of individual YouTube videos evolves.

III. Problem Description

We aim at improving the performance of the CDN infrastructure in terms of reducing the response time, improving the hit ratio of our request, as well as restricting the cost of copying from the origin server to surrogate servers. We consider the network topology, the server location, and restrictions in the cache capacity of the server. Taking as input data from OSNs and actions of users over them, we aim at recognizing objects that will eventually be popular in the realm of the OSN platform.

We search a policy such that given a graph $G(V, E)$, a set of R regions, where the nodes of the social network are distributed, and the posts P of the nodes, it recognizes the set of objects O that will be popular only in a subset of the regions (Table I). There is the content likely to be copied. The policy is represented by the function $Put(n_i, Predict(G, P, R, O))$, which takes as input a surrogate server $n_i \in N$ and the results of function $Predict$ (set of g objects that will be globally popular and λ objects that will be locally popular), such that:

$$\frac{Q_{hit}}{Q_{total}} \tag{1}$$

is maximum, whereas constraint

$$\sum_{\forall i \in O} S_i f_{ik} \leq C_k \tag{2}$$

is fulfilled, where:

$$f_{ik} = \begin{cases} 1 & \text{if object } i \text{ exists in the cache of surrogate server } k \\ 0 & \text{if object does not exist} \end{cases} \tag{3}$$

It returns the set of objects $o \in O$ that have to be placed in surrogate server $n_i \in N$. For $x, 1 < x < w$, objects that will be copied in the surrogate server and the capacity Cn_i of n_i, (2) has to be fulfilled:

Table I: Notation Overview

$G(V, E)$	Graph representing the social network
$V = \{V_1, ..., V_n\}$	Nodes representing the social network users
$E = \{E_{11}, ..., E_{1n}, ..., E_{nn}\}$	Edges representing the social network connections, where E_{ij} stands for friendship between i and j
$R = \{r_1, r_2, ..., r_\tau\}$	Regions set
$N = \{n_1, n_2, ..., n_u\}$	The surrogate servers set. Every surrogate server belongs to a region r_i
$C_i, i \in N$	Capacity of surrogate server i in bytes
$O = \{o_1, o_2, ..., o_w\}$	Objects set (videos), denoting the objects users can ask for and share
S_i, $o_i \in O$	Size of object i in bytes
Π_i	Popularity of object i, $i \in O$
$q_i = \{t, V_\psi, o_x\}, 1 < x < w, 1 < \psi < n$	Request i, consists of a timestamp, the id of the user that asked for the object, and the object id
$P = \{p_{12}, p_{13}, ..., p_{nw}\}$	User posts in the social network, where p_{ij} denotes that node i has shared object j in the social network
$pts_i, pte_i, 1 < i < \tau$	peak time start and peak time end for each region in secs
$Q = \{q_1, q_2, ..., q_\zeta\}$	Object requests from page containing the media objects, where q_i denotes a request for an object of set O
Q_{hit}, Q_{total}	Number of requests served from surrogate servers of the region of the user/ total number of requests
$X, Y \in R$	Closest timezones with mutual followers/ with highest centrality metric (HITS) values

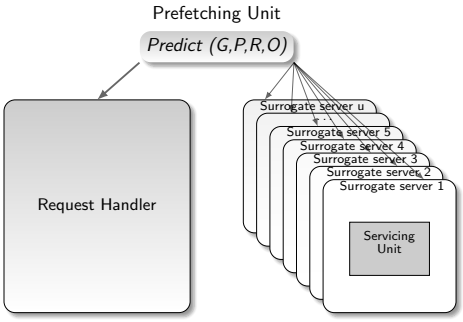

Figure 1: The Prefetching Unit.

$$S_1 + S_2 + S_3 + ... + S_x \leq Cn_i(1) \quad (4)$$

IV. Proposed Dynamic Policy

The proposed algorithm encompasses an algorithm for each new request arriving in the CDN and an algorithm for each new object in the surrogate server. Internally, the module communicates with the module processing the requests and each addressed server separately (Fig. 1).

A. For Every New Request in the CDN:

The main idea is to check whether specific time has passed after the start of the cascade, and then define to what extent the object will be copied. Initially, we check whether it is the first appearance of the object. The variable *o.timestamp* depicts the timestamp of the last appearance of the object in a request and helps in calculating the timer related to the duration of the cascade. If it is the first appearance of the object, the timer for the object cascade is initialized and *o.timestamp* takes the value of the timestamp of the request. If the cascade is not yet complete (its timer has not surpassed a threshold), we check the importance of the user applying the Hubs Authorities (HITS) algorithm and checking its authority score, as well as the viewership of the object in the media service platform (*Variation-1*, Fig. 2). In *Variation-2* (Fig. 3) we check the importance of the user, as well as if the time of the transmission is not within the peak-time range of the region of the user ([19]).

For users with a high authority score, we copy the object to all surrogate servers of the user's timezone and to the surrogate servers serving the timezones of all followers of the user (global prefetching). Otherwise, selective copying includes only the surrogates that the subpolicy decides (local prefetching).

Centrality is measured with the HITS algorithm, described in Section V. Subpolicy (Fig. 4) checks the X closest timezones where a user has mutual friends and out of them, the Y with the highest value of the centrality metric as an average. Highest value of the metric means that the object is likely to be asked for more times. Copying is performed to the surrogate servers that serve the above timezones.

1: **if** $o.timestamp$ == 0 **then**
2: $o.timer$ = 0;
3: $o.timestamp$ = $request\_timestamp$;
4: **else if** $o.timestamp$!= 0 **then**
5: $o.timer$ = $o.timer$ + ($request\_timestamp$ - $o.timestamp$);
6: $o.timestamp$ = $request\_timestamp$;
7: **end if**
8: **if** $o.timer > time\_threshold$ **then**
9: $o.timer$ = 0;
10: $o.timestamp$ = 0;
11: **else if** $o.timer < time\_threshold$ **and** $user.authority\_score > authority\_threshold$ **then**
12: copy object o to surrogate that serves user's V_i timezone;
13: **for all** user V_y that follows user V_i **do**
14: find surrogate server n_j that serves V_y's timezone;
15: copy object o to n_j;
16: **end for**
17: **else if** o.timer $<$ time_threshold **and** $o.\Pi_i > \Pi_i\_threshold$ **then**
18: copy object o to surrogates n_j that Subpolicy I decides;
19: **end if**

Figure 2: Variation-1 - Algorithm for every new request ($timestamp$, V_i, o) in the CDN

B. For Every New Object in the Surrogate Server:

For both variations, in the case that the new object does not fit in the surrogate server's cache, we define the *time_threshold* as the parameter for the duration that an object remains cached. We find the oldest objects and delete them. In the case that there are no such objects, we delete those with the largest timestamp in the cascade. In all other cases, the Least Recently Used (LRU) policy is applied for the removal of objects. The above are depicted in Fig. 5.

The heuristics applied in our approach are based on the following observations [2]: Users are more influenced by geographically close friends, and moreover by mutual followers, with the most popular users acting as authorities. Social cascades have a short duration (about 80% of the cascades end within 24 hours, with 40% ending in less than 3 hours). In our prefetching algorithm, we take into account the observation that the majority of cascades end within 24 hours. However, we introduce a varying time threshold for the cascade effect and the time that an object remains in cache. Values given in the time threshold variable also include 48 hours, as well as threshold covering the entire percentage of requests. The idea is to check whether specific time has passed after the start of cascade and, only in the case that the cascade has not ended, define to what

```
1: if o.timestamp == 0 then
2:   o.timer = 0;
3:   o.timestamp = request_timestamp;
4: else if o.timestamp != 0 then
5:   o.timer = o.timer + (request_timestamp - o.timestamp);
6:   o.timestamp = request_timestamp;
7: end if
8: if o.timer > time_threshold then
9:   o.timer = 0;
10:  o.timestamp = 0;
11: else if o.timer < time_threshold and user.authority_score > authority_threshold then
12:  copy object o to surrogate that serves user's V_i timezone;
13:  for all user V_y that follows user V_i do
14:    find surrogate server n_j that serves V_y's timezone;
15:    copy object o to n_j;
16:  end for
17: else if o.timer < time_threshold then
18:  if o.timestamp ∋ (pts_{r_{V_i}}, pte_{r_{V_i}}) and o.timestamp ∋ (pts_{r_{n_j}}, pte_{r_{n_j}}) then
19:    copy object o to surrogates n_j that Subpolicy I decides;
20:  end if
21: end if
```

Figure 3: Variation-2 - Algorithm for every new request ($timestamp$, V_i, o) in the CDN

```
1: find X timezones where (user V_i has mutual followers and they are closer to user's V_i timezone);
2: find the Y ⊆ X that (belong to X and have the highest HITS score);
3: for all timezones that belong to Y do
4:   find surrogate server n_j that serves timezone;
5:   copy object o to n_j;
6: end for
```

Figure 4: Subpolicy I

```
1: if o.size + current_cache_size ≤ total_cache_size then
2:   copy object o to cache of surrogate n_k;
3: else if o.size + current_cache_size > total_cache_size then
4:   while o.size + current_cache_size > total_cache_size do
5:     for all object o' in current_cache do
6:       if (current_timestamp - o'.timestamp) + o'.timer > time_threshold then
7:         copy o' in CandidateList;
8:       end if
9:       if CandidateList.size>0 and CandidateList.size != total_cache_size then
10:        find o' that o'.timestamp is maximum and delete it;
11:      else if CandidateList.size==0 or CandidateList.size==total_cache_size then
12:        use LRU to delete any object o ∈ O;
13:      end if
14:    end for
15:  end while
16:  put object o to cache of surrogate n_k;
17: end if
```

Figure 5: Algorithm for every new object o in the surrogate server n_k

Table II: Simulation Characteristics

Number of nodes in the topology	3500
Redirection Policy	Cooperative Environment (closest surrogate)
Number of origin servers	1
Number of surrogate servers	423
Number of users	162
Bandwidth	100 Mbit/sec

extent the object will be copied (*algorithm for every new request*). This check is also performed in *algorithm for every new object*, where we define the *time_threshold*. The latter roughly expresses the average cascade duration, as it defines the duration that an object remains cached.

V. Experimental Evaluation

For the experimental evaluation, we used the CDNsim simulator for CDNs [20]. The configuration of the simulation values is shown in Table II. We conducted a multitude of experiments (110, 55 for each variation), in which the time thresholds varied. For the extraction of reliable output, we had to conclude to a specific network topology, as well as make assumptions regarding the input dataset. The simulator takes as input files describing the underlying CDN and the traffic in the network, and provides an output of statistical results, discussed in the next Section.

A. Network Topology

There follows a short description of the process to define the nodes in the topology. These nodes represent the surrogate servers, the origin servers, and the users making the object requests (Fig. 6). For an in-depth analysis you can refer to [2].

To simulate our policy and place the servers in a real geographical position, we used the geographical distribution

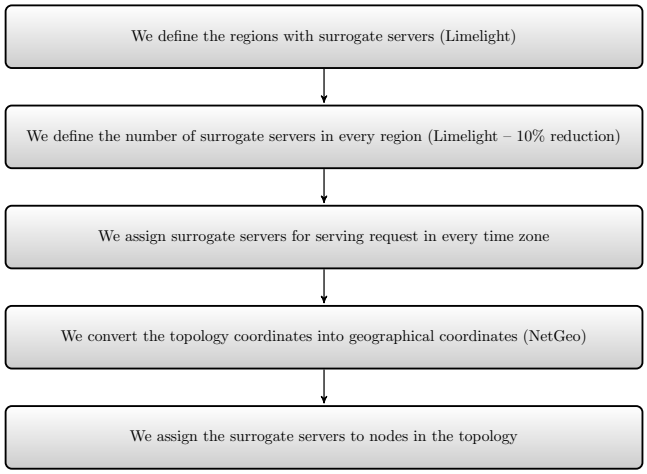

Figure 6: Methodology followed

Table III: Distribution of Servers over the World for the Experimental Evaluation

City	Servers	City	Servers
Washington DC	55	Toronto	12
New York	43	Amsterdam	20
Atlanta	11	London	30
Miami	11	Frankfurt	31
Chicago	37	Paris	12
Dallas	19	Moscow	10
Los Angeles	52	Hong Kong	8
San Jose	37	Tokyo	12
Seattle	15	Changi	5
Phoenix	3	Sydney	1

of the Limelight network [21]. For the smooth operation of the simulator the number of surrogate servers was reduced by the ratio of 10%, to ultimately include 423 servers (Table III). Depending on which surrogate region of the 20 the Limelight network defines is closer to each of the 142 Twitter timezones, we decided where the requests from this timezone will be redirected. The population of each timezone was also taken into consideration. The INET generator [6] allowed us to create an AS-level representation of the network topology. Topology coordinates were converted to geographical coordinates with the NetGeo tool from CAIDA, a tool that maps IP addresses and Autonomous System (AS) coordinates to geographical coordinates [22], and surrogate servers were assigned to topology nodes.

After grouping users per timezone (due to the limitations the large dataset imposes), each team of users was placed in a topology node. We placed the users in the nodes closer to those comprising the servers that serve the respective timezone requests, contributing this way to a realistic network depiction.

B. Number of Requests

1 million requests were considered sufficient, as CDNsim handles satisfyingly up to so many in general, with the number of objects being the dominant factor increasing the memory use of the tool. Also similar concept approaches use similar number of requests ([9] on a daily basis and [12]), and same number of distinct videos for generation of requests. With the requests generated from the generator following a long-tail distribution, 15 % of the whole catalog size was considered to be sufficient.

C. Threshold Values

Experimenting was conducted for time thresholds of 24 hours and 48 hours, as well as for the time threshold that covered all the requests. The threshold value for media service viewership was moderately chosen as 402408 (average media viewership in the dataset). The authority threshold score was tested for various values (0.006 / 0.02 / 0.04).

D. Influence Measurement Metrics

HITS algorithm [23] is a link analysis algorithm that rates web pages. Twitter uses a HITS style algorithm to suggest to users which accounts to follow [24], as well. A so-called good hub represents a page that points to many other pages, and a so-called good authority represents a page that is linked by many different hubs. We had to address memory usage issues for the very large graph dataset accommodated, and HITS was calculated using the MapReduce technique.

VI. Main Findings

The statistic reports produced by the simulator are used to evaluate the proposed policy. A short explanation of the metrics used in our experiments for extracting statistical results follows. They are described in detail in [20], along with various other metrics.

A. Metrics Used

1) Client Side Measurements: They refer to activities of clients, i.e. the requests for objects.

- *Mean Response Time*: indicates how fast a client is satisfied. It is defined as $\frac{\sum_{i=0}^{M-1} t_i}{M}$, where M is the number of satisfied requests and t_i is the response time of the i^{th} request. It starts at the timestamp when the request begins and ends at the timestamp when the connection closes.

2) Surrogate Side Measurements: They are focused on the operations of the surrogate servers.

- *Hit Ratio*: is the percentage of the client-to-CDN requests resulting in a cache hit. High values indicate high quality content placement in the surrogate servers.

Table IV: Average Metric Values for $X = 10$ Timezones of Close Mutual Friends

	Mean response time (Avg, $10^{-}2$ sec.)	**Hit ratio (Avg, %)**	**Active servers**	**Mean utility (Avg, %)**
Variation-1 - 24-h	1.1383	32.81	326	96.01
Variation-1 - 48-h	1.1352	33.08	326	96.01
Variation-1 - all-h	1.1172	34.58	325	96.04
Variation-2 - 24-h	1.1411	32.13	325	95.98
Variation-2 - 48-h	1.1376	32.43	326	96.00
Variation-2 - all-h	1.1174	34.38	324	96.03
Social Prefetcher 24-h	1.1412	32.12	325	95.98
Social Prefetcher 48-h	1.1377	32.42	326	96.00
Social Prefetcher all-h	1.1181	34.16	325	96.01

3) Network Statistics: They run on top of TCP/IP and concern the entire network topology.

- *Active Surrogate Servers*: refers to the servers being active serving clients.
- *Mean Surrogate Servers Utility*: is a value that expresses the relation between the number of bytes of the served content against the number of bytes of the pulled content (from the origin server or other surrogate servers). It is bounded to the range [0, 1] and provides an indication about the CDN performance. High net utility values indicate good content outsourcing policy and improved mean response times for the clients.

After conducting a multitude of experiments (55 for each variation) with varying threshold values, we reached the following conclusions.

Table IV presents the average values of four parameters for six cases of testing. The lowest mean response times appear for the cases of the time threshold covering all requests for both variations. In general, we observe a better performance in terms of mean response times and hit ratios achieved for the Variation-1, where the viewership within the YouTube platform is considered. Both variations perform better than the Social Prefetcher approach.

Comparison with the experimental results of [9] is not directly feasible because the authors do not address the storage issue and because the response times of a CDN infrastructure do not coincide with the download times for buffering stage of videos. In terms of performance, though, we note that with the policy proposed herein, there is a significant improvement over their respective improvement (30%) in pull-based methods employed by most CDNs, as we surpass the Social Prefetcher performance.

B. Impact of Time Threshold Duration

- *on Mean Response Time*:
  As the time threshold increases from 24 to 48 h and to hours covering the entire set of requests, we observe that the mean response time decreases steadily. Here, we present indicative values for the 10 closest timezones of mutual followers and varying subsets of 1, 5 and 10 timezones with the highest influence metric, respectively, where copying will ultimately be performed (Fig. 7) for both variations.

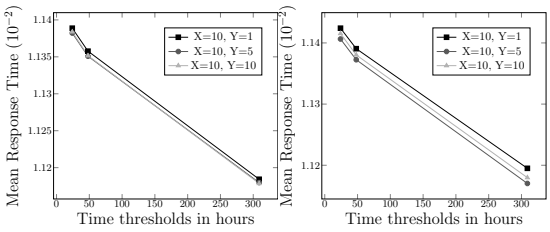

Figure 7: Effect of time threshold duration on mean response time for X closest timezones with mutual followers and Y timezones with the highest metric, where copying is ultimately performed for (i)Variation-1 and (ii)Variation-2

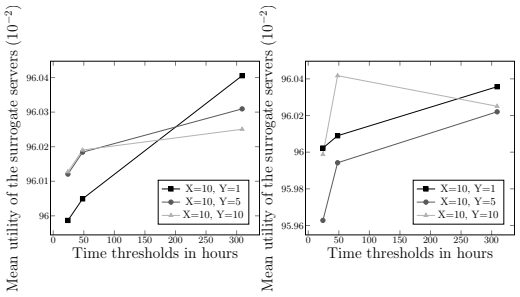

Figure 8: Effect of time threshold duration on mean utility of the surrogate servers for X closest timezones with mutual followers and Y timezones with the highest metric, where copying is ultimately performed for (i)Variation-1 and (ii)Variation-2

- *on Mean Utility of the Surrogate Servers*:
  With the exception of time threshold of 48 h for Variation-2, the mean utility of the surrogate servers shows a peak for both variations for the hours covering the entire set of requests. Here, we present indicative values for the 10 closest timezones of mutual followers and varying subsets of 1, 5 and 10 timezones with the highest influence metric (Fig. 8).
- *on Hit Ratio*: As the time threshold increases from 24 to 48 h and to hours covering the entire set of requests, we observe that the hit ratio steadily increases for both variations. This result is not unexpected because more requests are examined and more copies are likely to be performed (Fig. 9).

C. Impact of the Number of Timezones

- *on Active Servers*:
  For a fixed number of 10 closest timezones with mutual followers Variation-1 appears to use less active servers after the first timezone of highest centrality in the 24-h scenario. In the 48-h scenario Variation-1 depicts higher values than Variation-2 for the cases of 1, 6 and 10 timezones examined (Fig. 10).
- *on Mean Response Time*: The trade-off between the reduction of the response time and the cost of copying

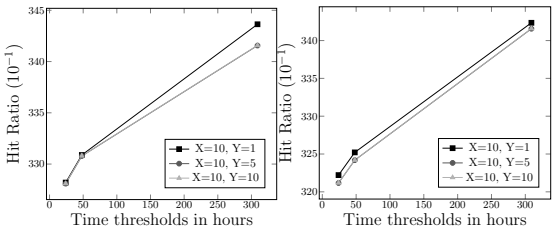

Figure 9: Effect of time threshold duration on hit ratio for X closest timezones with mutual followers and Y timezones with the highest metric, where copying is ultimately performed for (i)Variation-1 and (ii)Variation-2

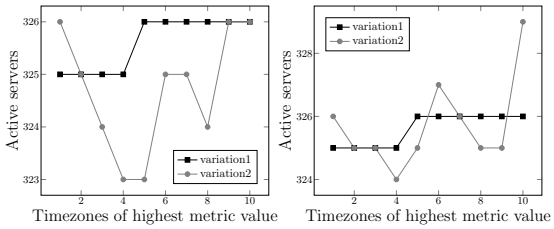

Figure 10: Effect of timezones used as Y on active servers for $X = 10$ closest timezones with mutual followers for (i)24-h and (ii)48-h

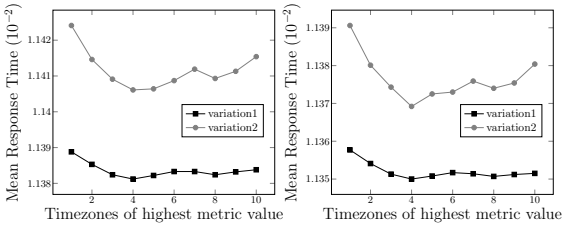

Figure 11: Effect of timezones used as Y on mean response time for $X = 10$ closest timezones with mutual followers for (i)24-h and (ii)48-h

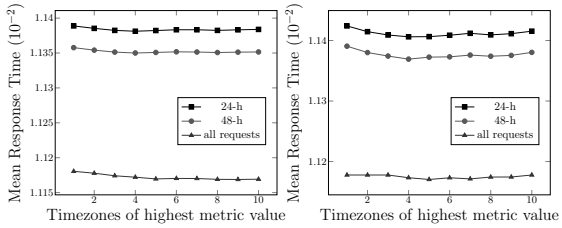

Figure 12: Effect of timezones used as Y on mean response time for $X = 10$ closest timezones with mutual followers for (i)Variation-1 and (ii)Variation-2

in servers is expressed with a decrease of the mean response time as the timezones increase, and a point after which the mean response time starts to increase again (Fig. 11 and Fig. 12).

For both variations this decrease in the mean response time occurs with approximately 4 timezones out of the 10 used (for a fixed number of closest timezones with mutual followers). After this point the slight increase in the mean response time is attributed to the delay for copying content to surrogate servers.

The cost for every copy is related to the number of hops among the client asking for it and the server where copying is likely to be made, according to the Put function. This point-of-change is also depicted for our variations in Fig. 13 for the most representative case of all requests.

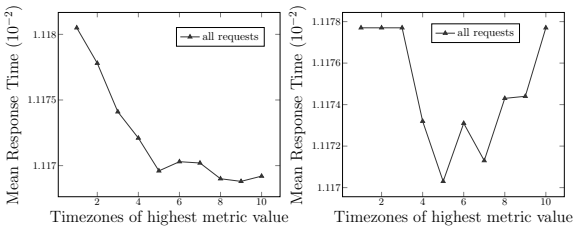

Figure 13: Mean response time for X=10 closest timezones with mutual followers and all possible Y values, $Y \in [1, 10]$ for (i)Variation-1 and (ii)Variation-2

VII. Conclusions

In this work, we further extended a dynamic policy of OSN content prefetching with temporal and other contextual parameters, depicting how OSNs can affect the content delivery infrastructure. We have presented how geo-social properties of users participating in social cascades prove to be of great importance toward improving the performance of CDNs and cloud, in the long-term. Bandwidth-intensive multimedia delivery over a CDN infrastructure is experimentally evaluated with realistic workloads, that many works in the related literature lack.

Whereas our study is limited to a specific OSN and media service, our results are generally applicable with a potentially high impact for large-scale systems with traffic generated by online social services and microblogging platforms. As the number of internet users increases dramatically and OSNs open new perspectives in the improvement of Internet-based content technologies, new issues in the architecture, design and implementation of existing CDNs arise. Our research agenda includes the generalization of proposed policy to deal with multiple OSN platforms and mobile CDN providers.

Acknowledgments

For the development of algorithms and to conduct the accompanying experiments, the cloud infrastructure of the Department of Computer Science of the University of Cyprus, as well as Amazon Web Services, were used.

REFERENCES

[1] Y. Li, Y. Shen, and Y. Liu, "Utilizing content delivery network in cloud computing," in *Computational Problem-Solving (ICCP), 2012 International Conference on*, Oct 2012, pp. 137–143.

[2] I. Kilanioti, "Improving multimedia content delivery via augmentation with social information. The Social Prefetcher approach." *Multimedia, IEEE Transactions on*, vol. 17, no. 9, pp. 1–1, 2015. [Online]. Available: http://bit.ly/1fUm7nD

[3] D. A. Easley and J. M. Kleinberg, *Networks, Crowds, and Markets - Reasoning About a Highly Connected World*. Cambridge University Press, 2010. [Online]. Available: http://www.cambridge.org/gb/knowledge/isbn/item2705443/?site_locale=en_GB

[4] E. Bakshy, I. Rosenn, C. Marlow, and L. A. Adamic, "The role of social networks in information diffusion," in *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, 2012, pp. 519–528. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187907

[5] K. Chard, S. Caton, O. Rana, and K. Bubendorfer, "Social Cloud: cloud computing in social networks," in *IEEE International Conference on Cloud Computing, CLOUD 2010, Miami, FL, USA, 5-10 July, 2010*, 2010, pp. 99–106. [Online]. Available: http://dx.doi.org/10.1109/CLOUD.2010.28

[6] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. B. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement 2007, San Diego, California, USA, October 24-26, 2007*, 2007, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/1298306.1298309

[7] A. Brodersen, S. Scellato, and M. Wattenhofer, "YouTube around the world: geographic popularity of videos," in *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, 2012, pp. 241–250. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187870

[8] T. Rodrigues, F. Benevenuto, M. Cha, P. K. Gummadi, and V. A. F. Almeida, "On word-of-mouth based discovery of the web," in *Proceedings of the 11th ACM SIGCOMM Conference on Internet Measurement, IMC '11, Berlin, Germany, November 2-, 2011*, 2011, pp. 381–396. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068852

[9] S. Traverso, K. Huguenin, I. Trestian, V. Erramilli, N. Laoutaris, and K. Papagiannaki, "Tailgate: handling long-tail content with a little help from friends," in *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, 2012, pp. 151–160. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187858

[10] I. Kilanioti, C. Georgiou, and G. Pallis, "On the impact of online social networks in content delivery," in *Advanced Content Delivery and Streaming in the Cloud*, M. Pathan, R. Sitaraman, and D. Robinson, Eds. Wiley, 2014.

[11] N. Sastry, E. Yoneki, and J. Crowcroft, "Buzztraq: predicting geographical access patterns of social cascades using social networks," in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems, SNS 2009, Nuremberg, Germany, March 31, 2009*, 2009, pp. 39–45. [Online]. Available: http://doi.acm.org/10.1145/1578002.1578009

[12] S. Scellato, C. Mascolo, M. Musolesi, and J. Crowcroft, "Track globally, deliver locally: improving content delivery networks by tracking geographic social cascades," in *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, 2011, pp. 457–466. [Online]. Available: http://doi.acm.org/10.1145/1963405.1963471

[13] S. Mitra, M. Agrawal, A. Yadav, N. Carlsson, D. L. Eager, and A. Mahanti, "Characterizing web-based video sharing workloads," *TWEB*, vol. 5, no. 2, p. 8, 2011. [Online]. Available: http://doi.acm.org/10.1145/1961659.1961662

[14] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao, "YouTube everywhere: impact of device and infrastructure synergies on user experience," in *Proceedings of the 11th ACM SIGCOMM Conference on Internet Measurement, IMC '11, Berlin, Germany, November 2-, 2011*, 2011, pp. 345–360. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068849

[15] P. Gill, M. F. Arlitt, Z. Li, and A. Mahanti, "YouTube traffic characterization: a view from the edge," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement 2007, San Diego, California, USA, October 24-26, 2007*, 2007, pp. 15–28. [Online]. Available: http://doi.acm.org/10.1145/1298306.1298310

[16] Z. L. P. Gill, M. Arlitt and A. Mahanti, "Characterizing user sessions on YouTube," in *ACM/SPIE Multimedia Computing and Networking Conference (MMCN '08), San Jose, USA, 2008*, 2008.

[17] X. Cheng, C. Dale, and J. Liu, "Statistics and social network of YouTube videos," in *16th International Workshop on Quality of Service, IWQoS 2008, University of Twente, Enskede, The Netherlands, 2-4 June 2008.*, 2008, pp. 229–238. [Online]. Available: http://dx.doi.org/10.1109/IWQOS.2008.32

[18] F. Figueiredo, F. Benevenuto, and J. M. Almeida, "The tube over time: characterizing popularity growth of YouTube videos," in *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, 2011, pp. 745–754. [Online]. Available: http://doi.acm.org/10.1145/1935826.1935925

[19] "Center for Applied Internet Data Analysis." [Online]. Available: https://www.caida.org

[20] K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, and Y. Manolopoulos, "CDNsim: A simulation tool for content distribution networks," *ACM Trans. Model. Comput. Simul.*, vol. 20, no. 2, 2010. [Online]. Available: http://doi.acm.org/10.1145/1734222.1734226

[21] J. L. C. Huang, A. Wang and K. Ross, “Measuring and evaluating large-scale cdns,” in *Internet Measurement Conference (IMC), 2008*, 2008.

[22] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafò, and S. G. Rao, “Dissecting video server selection strategies in the YouTube CDN,” in *2011 International Conference on Distributed Computing Systems, ICDCS 2011, Minneapolis, Minnesota, USA, June 20-24, 2011*, 2011, pp. 248–257. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2011.43

[23] J. M. Kleinberg, “Authoritative sources in a hyperlinked environment,” *J. ACM*, vol. 46, no. 5, pp. 604–632, 1999. [Online]. Available: http://doi.acm.org/10.1145/324133.324140

[24] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh, “WTF: the who to follow service at Twitter,” in *22nd International World Wide Web Conference, WWW ’13, Rio de Janeiro, Brazil, May 13-17, 2013*, 2013, pp. 505–514. [Online]. Available: http://dl.acm.org/citation.cfm?id=2488433