Geographically-Distinct Request Patterns for Caching in Information-Centric Networks

Alireza Montazeri and Dwight Makaroff
Department of Computer Science, University of Saskatchewan
Saskatoon, Canada (Email: alm164@mail.usask.ca, makaroff@cs.usask.ca)

***Abstract*—The current Internet architecture follows a host-centric communication model, intended for machine to machine connection and message passing. Modern Internet users are mainly interested in accessing information by name, irrespective of physical location. Information centric networking (ICN) was developed to rethink Internet foundations. In-network caching is one of the main features of ICN. Studying the performance of different caching algorithms in ICN requires a good understanding of users' request distributions in such networks. Most studies use simplifying assumptions for user request patterns since ICNs are not yet deployed.**

Geographically localized and global request patterns have both been observed to possess Zipf-like properties, although the local distributions are poorly correlated with the global distribution. Several independent Zipf distributions combine to form an emergent Zipf distribution in real client request scenarios. We develop an algorithm that can generate realistic synthetic traffic for geographic regions that possesses Zipf power-law properties as well as a global Zipf distribution. Our simulation results show that the caching performance would have different behaviour based on users' requests distribution.

I. Introduction

Hierarchical caching and content replication results in shorter propagation delays, lighter network traffic and smaller load on the servers. Several overlay distribution architectures have been deployed in this regard, notably traditional centralized caches, Content Distribution Networks (CDN), distributed Peer-to-Peer (P2P) content delivery, and hybrid CDN-P2P networks [1].

Underlay in-network caching [2] has been offered as a mechanism whereby ICN routers are equipped with caches. Policy details of ICN caches have been the subject of performance modelling studies [3] as well as caching support for video streaming [4] and mobility [5]. Studying the performance of different caching algorithms in ICN requires an understanding of users' request distributions.

Global and local request patterns of Internet services have been studied independently and collectively. Studies show that the users' requests in both contexts follow Power law distributions [6], [7]. Observations also show that the geographically local distributions are poorly correlated with the global distribution [8]. This suggests that several independent Zipf distributions combine to form an emergent Zipf distribution in real client request scenarios. Correlation between regions is shown to vary with respect to geographic location and/or other demographic qualities [9].

In this paper, we develop an algorithm that can generate synthetic traffic for geographical regions that possesses Zipf power-law properties as well as a global Zipf distribution. We provide initial experimentation suitable for an hierarchical cache typical of deployment in an ICN.

II. Related Work

To make different request patterns in different regions, several techniques have been employed. We list a selection.

- Rossini *et al.* randomly generate an object i according to a global Zipf popularity distribution. Then, they map this new request for i to a random user u in the network, attached to an access ICN router $C(u)$. Once subsequent requests for i are generated, they bias the user extraction process so to favor the selection of users closer to $C(u)$. In other words, the request will then be mapped to user v with a probability $P(d)$ that monotonously decreases with d that is distance between $C(u)$ and $C(v)$.
- Fayazbakhsh *et al.* [10] explore the effect of spatial skew. A spatial skew of 0 means that all locations follow the same global popularity distribution. A spatial skew of 1, at the other extreme, implies that the most popular object at one location may become the least popular object at some other location.
- Traverso *et al.* proposed Shot Noise Model (SNM) [9] that captures temporal and geographical locality of content popularity. The basic idea is to represent the overall request process as the superposition of populations of independent inhomogeneous Poisson processes, each referring to a data item.

None of these approaches guarantee that both the global and local distributions follow Zipf distributions.

III. Traffic Generator Principles

Having notations in Table I and $\Gamma_r = \{1, 2 \dots N\}$, the popularity of data items in region r for users' traffic (exogenous traffic) follows a Zipf distribution with parameter α_r (i.e. $p_{r,i} \propto (1/\Pi_{r,i})^{\alpha_r}$). Consequently, the request rate for data item i in r is calculated as $\lambda_{r,i} = p_{r,i}\lambda_r$.

We suppose two subregions, u and v. The order of data items' popularity in either u or v is different from the order in region r and satisfies the following conditions: $\Pi_u \neq \Pi_r$, $\Pi_v \neq \Pi_r$, $\Pi_u \neq \Pi_v$, $\Gamma_r = \Gamma_u = \Gamma_v$, and $\lambda_u + \lambda_v = \lambda_r$. In particular, Π_u, Π_v, α_u, α_v, λ_u and λ_v are not known. Algorithm 1 divides the global set of data items' popularity into two subsets, each one providing a Zipf distribution. The algorithm uses S_r^{λ}, the Zipf parameter range (α_{-}/α^{-}), threshold t as input and returns the unknowns.

TABLE I: Notation

N	$\triangleq$	number of data items
R	$\triangleq$	set of all regions
C	$\triangleq$	the overall cache budge; $C = \sum_{\forall r \in R} C_r$
α_r	$\triangleq$	Zipf parameter in region r
$p_{r,i}$	$\triangleq$	item i's popularity in region r
λ_r	$\triangleq$	request rate in region r
$\lambda_{r,i}$	$\triangleq$	item i's request rate in region r
S_r^λ	$\triangleq$	set of $\lambda_{r,i}$, $1 \leq i \leq N$: $\{\lambda_{r,1}, \lambda_{r,2} \dots \lambda_{r,N}\}$
$\Pi_{r,i}$	$\triangleq$	rank of item i's popularity in region r
$\Lambda_{r,j}$	$\triangleq$	request rate of an item with rank j in region r
Π_r	$\triangleq$	function mapping item i to $\Pi_{r,i}$ $1 \leq i \leq N$
Γ_r	$\triangleq$	set of data items in region r
C_r	$\triangleq$	available cache at router in region r

Algorithm 1 Pseudo-Bisect Zipf distribution

1: **procedure** DIVIDE–ZIPF
2: **INPUT:**
3: $S_r^\lambda, \alpha_-, \alpha^-, t$
4: **OUTPUT:**
5: Π_v, Π_u, α_v, α_u, S_v^λ *and* S_u^λ
6: *Is-Zipf-like* $\leftarrow false$
7: **while** !*Is-Zipf-like* **do**
8: $T \leftarrow \{1, 2 \dots N\}$
9: $i \leftarrow Weight\text{-}Random(T)$;
10: $T \leftarrow T - \{i\}$; $\Pi_{u,i} \leftarrow 1$
11: *Randomly pick* $\lambda_{u,i}$: $0 < \lambda_{u,i} < \lambda_{r,i}$
12: $\lambda_{v,i} \leftarrow \lambda_{r,i} - \lambda_{u,i}$
13: *Pick up random* α_u, $\alpha_- < \alpha_u < \alpha^-$
14: $\theta \leftarrow \sum_{k=1}^{N} (1/k)^{\alpha_u}$
15: $p_{u,i} \leftarrow 1/\theta$
16: $\lambda_u \leftarrow \lambda_{u,i}/p_{u,i}$; $\lambda_v \leftarrow \lambda_r - \lambda_u$
17: **for** j *from* 2 *to* N **do**
18: $\Lambda_{u,j} \leftarrow \lambda_u\big((1/j)^{\alpha_u}/\theta\big)$
19: **end for**
20: $j \leftarrow 2$
21: **while** $T \neq \emptyset$ **do**
22: $k \leftarrow$ *Weight_Random*(T)
23: **if** $\Lambda_{u,j} \leq \lambda_{r,k}$ **then**
24: $\Pi_{u,k} \leftarrow j$; $\lambda_{u,k} \leftarrow \Lambda_{u,j}$; $T \leftarrow T - \{k\}$
25: $\lambda_{v,k} \leftarrow \lambda_{r,k} - \lambda_{u,k}$
26: $j \leftarrow j + 1$
27: **end if**
28: **end while**
29: *Is-Zipf-like* $\leftarrow Evaluate(S_v^\lambda, t, \Pi_u, \alpha_v)$
30: **end while**
31: **end procedure**

The algorithm starts with a weighted random selection of data item i, (the most popular data item in region *u* (line 9, 10)). Then, a portion of $\lambda_{r,i}$ is assigned to $\lambda_{u,i}$ (line 11). The remaining request rate for *i* is assigned to $\lambda_{v,i}$ (line 12). The algorithm then randomly selects α_u (line 13). Based on $\lambda_{u,i}$ and α_u, λ_u could be now calculated (lines 14-16). Having λ_u and α_u, the algorithm then calculates the request rate for other ranks in *u* (lines 17-19).

In the next stage, the algorithm assigns data items to ranks in $[2, N]$ for region *u* (lines 20-28). The algorithm starts with the second rank in *u* (line 20) since *i* is already assigned to the first rank. For the j^{th} rank in region *u* and among the data items in T with request rate larger than $\Lambda_{u,j}$, k is randomley selected and considered as the j^{th} most popular item in *u* (lines 22-27). Correspondingly, the request rate for item *k* in region *v* is obtained.

In the final stage, the algorithm evaluates the distribution of requests in *v* to determine how Zipf-like it is (line 29, details omitted for space reasons). Function *Evaluate* repeatedly creates *z* as a Zipf-like distribution. It then compares the popularity distribution of data objects in the region *v* with *z*, using the coefficient of determination. If the correlation exceeds a threshold *t*, we return success alongside α_v, as the parameter of closest Zipf-like distribution to users' requests in *v*, and Π_v. If *v*'s distribution is sufficiently Zipf-like, the algorithm ends. Otherwise, we find a new order for *u* by repeating lines 7-30.

To create k sub-request patterns, Algorithm 2 applies Algorithm 1 to the highest rate existing subregion. Table II shows some subregions for 1000 data objects, $k = 10$, $\lambda_r = 15$ and $\alpha_r = 1.2$. Different subsets have different λ_u, α_u and Π_u.

Algorithm 2 creating k Zipf-like sub-request patterns

1: **procedure** DIVIDE–ZIPF–TO–K
2: **INPUT:**
3: $S_r^\lambda, \alpha_-, \alpha^-, t, k$
4: **OUTPUT:**
5: $\Pi_1 \dots \Pi_k, \alpha_1 \dots \alpha_k, S_1^\lambda \dots S_k^\lambda$
6: $W \leftarrow \{\}$
7: **while** $|W| < k$ **do**
8: $Divide\text{-}Zipf(S_r^\lambda, \alpha_-, \alpha^-, t, \Pi_u, \Pi_v, \alpha_u, \alpha_v, S_u^\lambda, S_v^\lambda)$
9: $W \leftarrow W \bigcup u \bigcup v$
10: $r \leftarrow MaxRequestRate(W)$
11: **end while**
12: **end procedure**

TABLE II: A sample output of Algorithm 2.

x	λ_x	α_x	Π_x										
u_1	1.46	0.91	12	30	17	6	56	16	128	65	31	13	...
u_2	1.04	0.53	7	33	151	90	133	72	9	28	242	199	...
u_3	1.03	0.79	31	9	6	40	93	47	120	19	70	55	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
u_{10}	1.11	0.51	25	11	10	95	63	73	56	2	145	169	...
r	15	1.2	1	2	3	4	5	6	7	8	9	10	...

To study the independence properties of subregions, we perform a series of experiments on Algorithm 2. We also compute the following two correlations: (1) average global-region-correlation between the global traffic distribution and all regional traffic distributions; (2) average pairwise-region-correlation between the traffic distribution of all regions.

Figure 1 and Figure 2 demonstrate the correlations for $N = 20,000$ and $\alpha = 1.0$. Figure 1 depicts the correlations

based on σ_α, the variation of α_u between the regions. Neither correlation is affected by σ_α. Note that the global correlation is substantially higher than the local pairwise correlations. This is to be expected as we choose the most popular item in a subregion among the most popular items in the remaining largest region. Pairwise correlation is very low.

Figure 2 shows correlations over σ_α, the variation of λ_u between the regions. Larger σ_λ results in smaller correlation for both pairwise local and global distributions. Similar behaviour for both correlations over σ_α and σ_λ is observed for $\alpha_r = \{0.8, 1.2, 1.4\}$ and other datasets for populations of 1000, 5000, and 10,000.

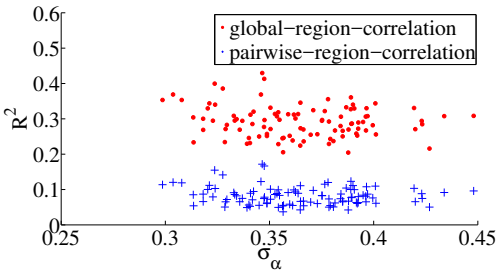


Fig. 1: Global/pairwise correlation vs. σ_α.

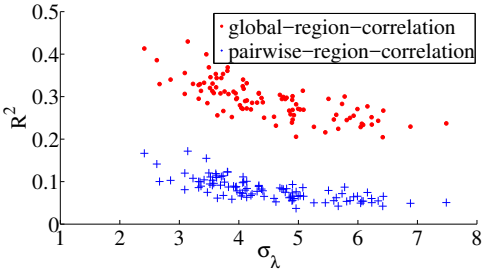


Fig. 2: Global/pairwise correlation vs. σ_λ.

We investigated the relationship between global-region-correlation and different values of N and α. Our observations show that the correlation decreases with N, since larger population sizes result in more possible permutations, producing sub-distributions with differing orders of data items. We also show that the correlation increases with α. A higher α generates fewer popular data items as choices for the "most-popular" data items in sub-distribution *u* (i.e a higher popularity *decay*); sub-distributions have similar content at the head of the distributions. Pairwise correlations show a similar trend.

A. Dividing the cache budget proportionally among regions

The following four policies/scenarios are considered to divide the cache budget:

1) Proportional-Edge-only (s_1): the cache size of an edge router is proportional to the its corresponding region's request rate ($C_u \propto \lambda_u$ if u is an edge region; otherwise $C_u = 0$).
2) Proportional-all-network (s_2): the cache budget is distributed among the all regions proportional to the each region's request rate.
3) Equal-Edge-only (s_3): ($C_u = C/k$ if u is an edge router; otherwise $C_u = 0$).
4) Equal-all-network (s_4): ($C_u = C/|R|$).

Our process to allocate the cache budget gets C, R and exogenous request rate for each data item i at region *r*, plus the structure of overlay ICN tree. The output will be the portion of cache budget dedicated to region r, $\forall r \in R$.

Having λ_r initially only for the edge routers, we divide C proportionally among those routers. The miss rates of items in region r, ($\lambda'_{r,i}$), for all the data items and edge routers are calculated depending on the replacement algorithm. Next, the algorithm calculates the request rate at the parents of edge routers. To complete a level in the tree, the parent nodes are then included in the cache distribution, since their request rates are now known approximately. Any intermediate router cache allocation will increase the miss rate at children caches. We continue calculating miss rates and cache sizes for routers at higher levels incrementally in the ICN tree.

IV. Performance Evaluation

In this section, we conduct a simple proof-of-concept cache performance experiment with identical and differential request patterns in the following four topologies: Geant, Tiger, Level3 and Dtelecom (see Table III).

TABLE III: Specification of topologies.

name	Number of Routers	Number of Edge Routers	depth
Level3	68	61	4
Dtelecom	46	41	3
Geant	22	10	5
Tiger	22	10	4

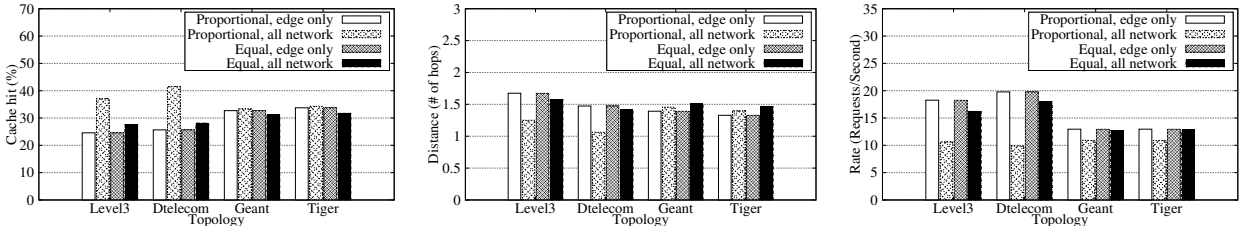
A. Experimental Methodology

We assume each node in the four topologies represents a different region, with one caching router. We also assume only one region has a content server with the original copy of the data item. We use ccnSim [11] as a ICN simulator and each simulation represents 10^6 seconds (11.6 days). The server is the root of an ICN overlay tree for all communication. Users are connected to edge regions only; edge routers receive exogenous traffic and intermediate routers receive endogenous traffic. For the geographical locality generation, k is the number of edge routers.

Three metrics are used to express the caching performance: (1) the hit ratio; (2) the request rate arrived at the content server; and (3) distance to the first copy of the data item (a measure of latency). 2-LRU is used in this paper as the caching replacement algorithm, since Garetto *et al.* showed it outperforms LRU [3]. We set $N = 20000$, global Zipf shape parameter $\alpha_r = 1.0$, the global request rate $\lambda_r = 40$, $\alpha_- = 0.5$, $\alpha^- = 2.0$.

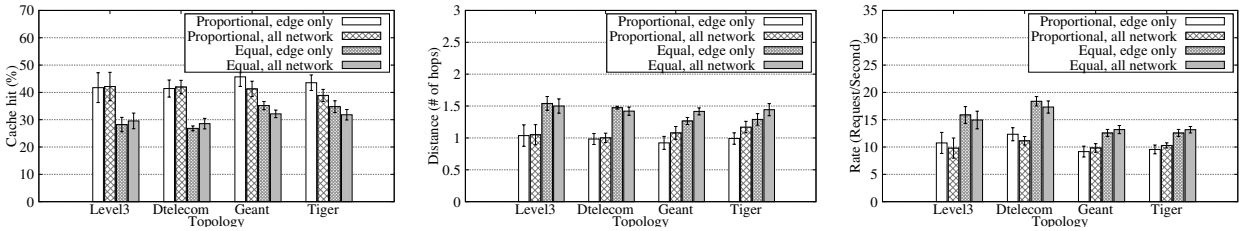
B. Baseline Configuration

As the baseline configuration, we consider identically distributed requests. Figure 3 shows the metrics for all four scenarios. For Dtelecom and Level3 topologies, proportional cache allocation in the entire network results



(a) Hit ratio (b) Distance (c) Request rate at the server

Fig. 3: Identical Zipf distribution for all regions.



(a) Hit ratio (b) Distance (c) Request rate at the server

Fig. 4: Geographical locality.

in better performance while all three metrics in Geant and Tiger topologies are always similar with a slight edge to proportional allocation in the entire network.

C. Influence of geographically localized traffic

Figure 4 depicts the cache performance with geographic locality. The bars represent the average of 5 runs, with error bars indicating 1 standard deviation. The three metrics for Level3 and DTelecom topologies for proportional allocation are very close since a high percentage of nodes are at the edges (refer to Table III).

Proportional allocation is effective in improving all three metrics for all four topologies, as to be expected since equal allocation ignores geographic locality in rate. On the other hand, dedicating cache budget to the intermediate routers in the network does not improve performance with geographical locality. Intermediate routers obtain only a $5-10\%$ hit ratio. Intermediate router traffic, (the aggregate of edge misses), does not have a Zipf-like distribution. This makes LRU-2 and LRU caching replacement algorithms inefficient. A different replacement policy for interior caches may make better use of resources is part of future work.

V. Conclusion and Future Work

In this paper, we proposed an algorithm to generate locally biased request patterns which follow different Zipf distributions for each region, and combine to form a Zipf distribution. Then, we studied the performance of the system in four different scenarios. Simulation results show that geographical locality causes the different system behaviour in the simple test scenarios.

We also confirm that caching in the core of network has little advantage compared to caching only in edge routers. We will examine alternate strategies for off-path caching, and on-path caching in terms of search strategy and replica placement. Hierarchical consideration of regional popularity will also be explored. This would be combined with potential correlations between geographically distant, but culturally similar demographics (university campuses).

References

[1] H. Yin, X. Liu, T. Zhan, V. Sekar, F. Qiu, C. Lin, H. Zhang, and B. Li, "Design and deployment of a hybrid CDN-P2P system for live video streaming: Experiences with LiveSky," in *ACM Multimedia*, Beijing, China, Oct. 2009, pp. 25–34.

[2] J. M. Wang, J. Zhang, and B. Bensaou, "Intra-AS cooperative caching for content-centric networks," in *SIGCOMM Workshop on Information-centric Networking*, Hong Kong, China, Aug. 2013, pp. 61–66.

[3] M. Garetto, E. Leonardi, and V. Martina, "A Unified Approach to the Performance Analysis of Caching Systems," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 3, pp. 12:1–12:28, May 2016.

[4] C. Anastasiades, A. Gomes, R. Gadow, and T. Braun, "Persistent caching in information-centric networks," in *IEEE LCN*, Clearwater Beach, FL, Oct. 2015, pp. 64–72.

[5] S.-E. Elayoubi and J. Roberts, "Performance and cost effectiveness of caching in mobile access networks," in *ACM ICN*, San Francisco, CA, Oct. 2015, pp. 79–88.

[6] L. A. Adamic and B. A. Huberman, "Zipf's Law and the Internet," *Glottometrics*, vol. 3, pp. 143–150, 2002.

[7] A. B. Downey, "Evidence for Long-tailed Distributions in the Internet," in *ACM SIGCOMM Workshop on Internet Measurement*, San Francisco, CA, Nov. 2001, pp. 229–241.

[8] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch Global, Cache Local: YouTube Network Traffic at a Campus Network: Measurements and Implications," in *SPIE Multimedia Computing and Networking*, San Jose, CA, Jan. 2008, pp. 1–13.

[9] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini, "Unravelling the impact of temporal and geographical locality in content caching systems," *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1839–1854, Oct. 2015.

[10] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable ICN," in *ACM SIGCOMM*, Hong Kong, China, Aug. 2013, pp. 147–158.

[11] R. Chiocchetti, D. Rossi, and G. Rossini, "ccnSim: An highly scalable CCN simulator," in *IEEE International Conference on Communications*, Budapest, Hungary, Jun. 2013, pp. 2309–2314.