FGPC: Fine-Grained Popularity-based Caching Design for Content Centric Networking

Dung Ong Mau, Min Chen
School of Computer Science and Technology
Huazhong University of Science and Technology, China
omdung@gmail.com, minchen2012@hust.edu.cn

Tarik Taleb
NEC Europe Labs, Germany
talebtarik@ieee.org

Xiaofei Wang, Victor C. M. Leung
Dept. of Electrical and Computer Engineering
The University of British Columbia, Canada
{xfwang, vleung}@ece.ubc.ca

ABSTRACT

Content Centric Networking (CCN) is a content name-oriented approach to disseminate content to edge gateways/routers. In CCN, a content is cached at routers for a certain time. When the associated deadline is reached, the content is removed to cope with the limited size of content storage. If the content is popular, the previously queried content can be reused for multiple times to save bandwidth capacity. It is, therefore, critical to design an efficient replacement policy to keep popular content as long as possible. Recently, a novel caching strategy, named Most Popular Content (MPC), was proposed for CCN. It considers the high skewness of content popularity and outperforms existing default caching approaches in CCN such as Least Recently Used (LRU) and Least Frequency Used (LFU). However, MPC has some undesirable features, such as slow convergence of hitting rate and unstable hitting rate performance for various cache sizes. In this paper, a new caching policy, dubbed Fine-Grained Popularity-based Caching (FGPC), is proposed to overcome the above-mentioned weak points. Compared to MPC, FGPC always caches coming content when storage is available. Otherwise, it keeps only most popular content. FGPC achieves higher hitting rate and faster convergence speed than MPC. Based on FGPC, we further propose a Dynamic-FGPC (D-FGPC) approach that regularly adjusts the content popularity threshold. D-FGPC exhibits more stability in the hitting rate performance in comparison to FGPC and that is for various cache sizes and content sizes. The performance of both FGPC and D-FGPC caching policies are evaluated using OPNET Modeler. The obtained simulation results show that FGPC and D-FGPC outperform LRU, LFU, and MPC.

MSWiM'14, September 21–26, 2014, Montreal, QC, Canada.

http://dx.doi.org/10.1145/2641798.2641837 .

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

NDN;Future Internet;CCN;Caching Policy

1. INTRODUCTION

With the convergence of cloud computing, social media, and mobile communications, the types of data traffic are becoming more diverse and the community of Internet users is exponentially growing. Millions of multimedia files (e.g., pictures, voices and videos) are generated and shared by producers and consumers. This trend has posed high requirements on network bandwidth and data storage, congesting networks and overloading servers.

In order to alleviate the problem of bandwidth scarcity, Content Centric Networking (CCN) has been proposed to effectively distribute popular content to a potential number of users [1] [2]. To maximize the probability of content sharing while ensuring minimal upstream bandwidth demand and lowest downstream latency, routers/gateways should cache exchanged content as long as possible. Caching decision and replacement policies play a crucial role in CCN's overall performance. The Least Recently Used (LRU) and Least Frequently Used (LFU) replacement policies were originally proposed for CCN. However, LRU and LFU suffer from low efficiency and that is due to the following reason: LRU and LFU make replacement decision based on only existing content located on the cache, i.e., LRU uses a time stamp of the content while LFU counts delivery frequency of the content. It should be noted that the name of content in CCN has the same form with the Uniform Resource Locator (URL), e.g., ccnx://root/prefix1/prefix2/../. In both schemes, content popularity is overlooked, which causes inaccuracy in identifying the popular level of a newly arriving content. Thus, inadequate content replacement may happen: old popular content may be replaced by new unpopular one.

In order to address the above limitation and improve the precision of content replacement decision, a novel caching strategy, named Most Popular Content (MPC), was proposed in [3]. In MPC, every router/gateway counts the local number of requests for each content name, and stores the pair (content name; popularity count) into a content popularity table. Once the popularity of a content object reaches a predetermined threshold in a caching node, it is tagged as a popular content and is stored in the cache. By storing only popular content, MPC caches less content, saves resources and reduces the number of cache operations, which makes it achieve a higher hitting ratio in comparison to LRU and LFU.

Although some memory space is saved in MPC, the utilization of the cache is typically not high. Based on our observation, higher hitting rates can be achieved by intelligently utilizing vacant cache memory to contribute to a certain amount of hitting ratio. For example, when there is room in available memory, unpopular content can be stored. When the cache becomes full, unpopular content are removed, yielding space for popular ones. Furthermore, such strategy of caching less content, adopted by MPC, results in a slow convergence of MPC in terms of hitting rate performance.

As a remedy to the above limitations, we propose a novel replacement policy, dubbed Fine-Grained Popularity-based Caching (FGPC). Similar in spirit to MPC, FGPC maintains a large table to generate three kinds of statistic information, namely *i*) content names, *ii*) popularity levels of content by counting the frequency of appearances of a content name, and *iii*) time stamp of used content located in a cache. In order to quickly achieve high hitting rate, the cache always stores new coming content when it has available memory. To avoid inadequate replacement decision, once the cache is overflowed, FGPC checks the counting value of new content names. If the counter reaches a predetermined popularity threshold value, the new content will be stored using the LRU replacement policy; otherwise the new content is simply deleted.

Compared with the existing LRU/LFU or MPC solutions, the FGPC scheme has the following unique features:

- In FGPC, we carefully investigate the characteristics of CCN where content name includes prefixes. Thus, the high skewness of content popularity is the main reason to make MPC/FGPC ourperform LRU. We add a content popularity table to handle content name, content counter, and time stamp.
- The way Internet users request content is fine tuned with *Pareto* principle; that is 20% popular content are requested by 80% number of users.
- Popular content always appear with a high probability and vice versa. These common characteristics also indicate that the FGPC algorithm is feasible, practical, and compliant with the current Internet object access behavior.

Moreover, an enhanced version of FGPC, dubbed Dynamic FGPC (D-FGPC), is proposed. Because the number of content can be stored in a cache and the number of arrival requests to a gateway/router is changing dynamically over time, the popularity threshold value should be adaptively adjusted on-the-fly over time, too. For this reason, the enhanced version of FGPC is called D-FGPC.

Table 1: Notation

Symbol	Definition
F	File size by default
F^P	Practical file size
$\alpha = \frac{F^P}{F}$	The variety factor of F
$\lvert F\rvert$	File number
$\lvert F\rvert F$	Catalog size
C_{size}	Cache size
$C^R_{size} = \frac{C_{size}}{\lvert F\rvert F} * 100\%$	Relative cache size

From the background of CCN [2], we import CCN strategy to all network elements on top of IP layer in the OPNET simulator [4] [5]. The existing LRU, MPC and our proposed FGPC and D-FGPC policies are successfully constructed in CCN nodes. Our simulation results prove that CCN is a good solution to existing challenges of traditional IP networks. The results indicate that FGPC and D-FGPC outperform LRU and MPC with highly effective caching.

For the sake of better readability, Table I lists up the notations used in this paper. The remainder of this paper is organized as follows. Section II provides some related research work. Section III introduces the two proposed content replacement algorithms, FGPC and D-FGPC. Section IV portrays the simulation setup and discusses the simulation results. Finally, Section V concludes this paper.

2. RELATED WORK

Recently, CCN has become a hot research area, and several projects and prototypes have applied CCN [6]- [11]. CCN is a network architecture, built on the Internet Protocol (IP) engineering principle, but treats content as a primitive. Further details on the CCN architecture can be found in [2].

Along with the overwhelming library of applications and services, millions of content exist on the Internet nowadays; an important portion of it being “User Generated Content” (UGC) [12]. To understand the nature and impact of UGC systems, the work conducted in [12] analyzes YouTube and Daum, the world’s largest UGC Video on Demand (VoD) systems. The results show 10% of the top YouTube popular videos account for nearly 80% of views, while the rest 90% of the YouTube videos account for small number of requests. Daum data also reveals a similar behavior. In fact, the skewness in content popularity had been considered for a long time ago [13] [14] [15]. At this moment, the robustness of data traffic becomes critical and researches in ICT need to find out optimal solutions such as in-network caching to offload data traffic [16] [17].

Similarly, data traces are collected from two UGC sites in China, namely Youku and 6CN in [19]. The results indicate that top 5% videos contribute to over 80% views, which demonstrate higher skewness in video popularity in comparison to YouTube or Daum. An interesting implication of this skewed distribution is that, by storing only 5% to 10% of long-term popular videos, a cache can serve 80% of requests [18].

Edge caching is of vital importance for CCN. CCN becomes highly efficient with intelligent caching at edge routers/gateways. This efficiency degrades as caching occurs far from end-users [20]. However, global CCN caching, supported with Self-Organized Networking (SON) functions among caches can offer benefits well beyond only edge caching in sub-networks [21]. A variety of cache sizes and popularity skewness are often considered to have a positive effect on the performance of different replacement policies. In [22] and [23], the simulation results show that the overall network performance improves in case of large cache sizes or high skewness factors.

3. FGPC ALGORITHM

In MPC, the drawback of LRU is identified: the CCN default caching (as known as LRU/LFU) strategy always stores content at all nodes on the delivery path. This approach could replace popular content by unpopular ones. MPC is proposed with a caching less approach to solve the limitations of LRU/LFU. In the MPC, they key idea is to cache only the most popular content and that is in order to achieve high performance and save resources. However, on the other hand, caching less strategy causes two other problems. First, despite the fact that the probability appearance of unpopular content is small, they still help to contribute to a certain amount of hitting ratio. Moreover, in CCN standards, it is specified that to maximize the content-sharing probability with minimal upstream bandwidth demand and lower downstream latency, Content Stores (CS) should keep arriving data packages (DataPks) as long as possible. Second, with cashing less strategy, hitting rate slowly converges to a steady state. Hereunder, we introduce FGPC highlighting how it copes with the issues of LRU, LFU and MPC. A new variant of FGPC is portrayed afterwards.

3.1 FGPC strategy

In FGPC, each CCN node maintains a table containing statistics about the popularity of a content name and that is in the form of a content counter, along with a time stamp. Indeed, FGPC keeps track of popular content by locally counting the frequency of appearances of each content name. As shown in Fig. 1, there are three main operations conducted by FGPC:

- FGPC constantly updates three kinds of statistic information in the popularity table, i.e., content name, content counter and time stamp when receiving a content from upstream or delivering a content downstream.
- FGPC always stores newly arriving content (regardless their popularity) when CS has available space.
- When a CS is about to get full and a new content arrives, FGPC compares the popularity level of the new content (P_X) to a predefined popularity threshold value (P_{th}).If Px exceeds P_{th}, FGPC adopts the LRU policy to store the new content into CS. Otherwise, FGPC ignores the content and does not cache it.

In FGPC, CCN nodes achieve effective caching when they recognize the popularity levels of all content and keep popular content for longer times than other less popular content items. The

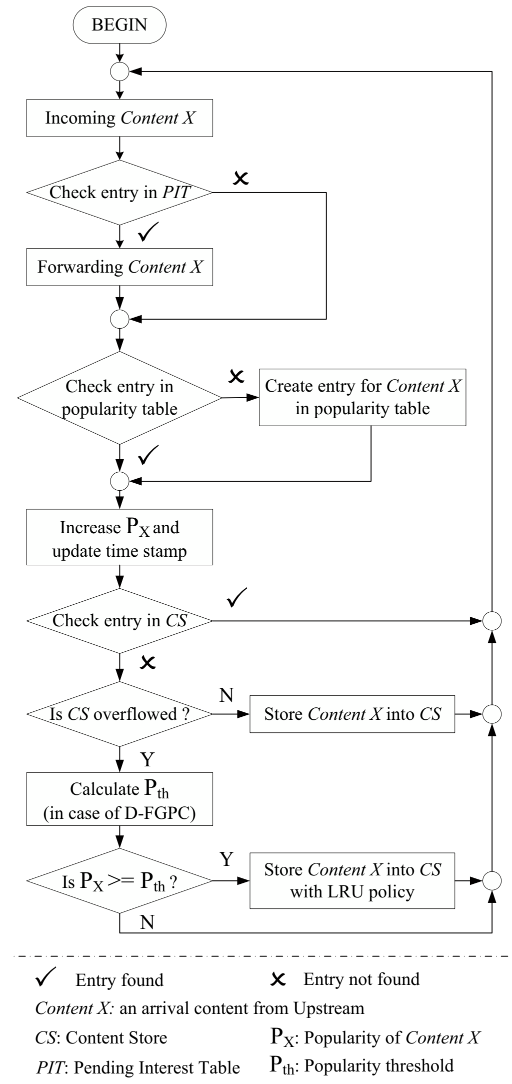

Figure 1: FGPC and D-FGPC flowchart.

trade-offs between performance and space availability as well as with computational complexity should be taken into account. In order to significantly reduce the overhead of the popularity table, algorithms such as the message-digest (MD5) Hash algorithm or mapping content names to digital numbers could be effective. For instance, to keep one million content names, and given the fact that a MD5 Hash value uses $16Bytes$ (i.e., $1B$ for counter and $3B$ for time stamp [3]), a CCN node would need an additional memory of merely $20 * 10^6 B$ or $19.0735MB$ for the popularity table. Nowadays, there is a clear technical trend that network devices are provided powerful packet processing and large memory. This makes no extra cost impact to the provider or to the users for in-network caching with FGPC approach.

3.2 D-FGPC strategy

In the basic FGPC scheme, the popularity threshold (P_{th}) is fixed for filtering popular content. Admittedly, this is not realistic as in real-life implementations, the popularity of a content changes with time. In addition, the number of arriving interest packages ($n_{interest}$), the practical file sizes (F^P) as well as available cache sizes (C_{size}) of different routers/gateways dynamically change. In this section, we define D-FGPC, as a new variant of FGPC with dynamic P_{th} adjustment. For the sake of better illustration, Fig.1 shows the main operations of both the FGPC and D-FGPC schemes. Hereunder, we discuss the relationship between the parameters P_{th}, $n_{interest}$, F^P and C_{size} as follows:

- Given a fixed F^P, when C_{size} increases, available memory of CS increases too. Then, P_{th} must decrease to relax the popularity-based content filtering and help fill up the available memory of CS as soon as possible. For this reason, D-FGPC fully utilizes the cache and converges faster than FGPC in case of large C_{size}.
- Given a fixed C_{size}, when F^P increases, the number of practical files stored on CS decreases. Then, P_{th} must increase to improve the popular content filtering. It should be noted that content in the memory are constructed by a lot of chunk files, and users normally request for completed content which include a group of chunk files together.
- Given a fixed C_{size}, when $n_{interest}$ increases, the number of newly coming content will also increase. In order to efficiently use the limited cache size and to accommodate the largest number of content requests, P_{th} must set up to higher values to enhance the popular content filtering.

Similarly, we have reverse situations and operations when C_{size}, F^P and $n_{interest}$ decrease, respectively. As above discussion about the relationship between the parameters P_{th}, $n_{interest}$, F^P and C_{size}, a dynamic setting of the popularity threshold can be achieved using Eq.1, whereby β is a constant that reflects the content filtering factor.

$$P_{th} = \beta * \frac{n_{interest} * F^P}{C_{size}} \tag{1}$$

In the MPC and FGPC schemes, P_{th} is fixed at five [3]. In case of D-FGPC scheme, β should make the threshold value around five for a fair comparison with the MPC/FGPC performance. For example, when we set F^P and C_{size} to constant default values, e.g., the middle values in the range, $n_{interest}$ can be estimated every minutes, then β is determined when P_{th} is set at five. With the calculated β, when either F^P or C_{size} varies, P_{th} will vary around five too.

4. SIMULATION AND RESULTS

4.1 Network architecture

To evaluate the performance of FGPC and D-FGPC, we implemented CCN and conducted simulations using the OPNET Modeler 16.0 [4] [5]. In the simulations, CCN is overlayed over the IP layer. Indeed, we integrated the CCN processing modules into all network elements, such as routers, PCs, servers and IP Cloud. Fig.2 shows the envisioned OPNET model for CCN node processor which is integrated into routers.

With every intention to consider a typical Internet network topology, we envision the network topology as shown in Fig.3. Almost

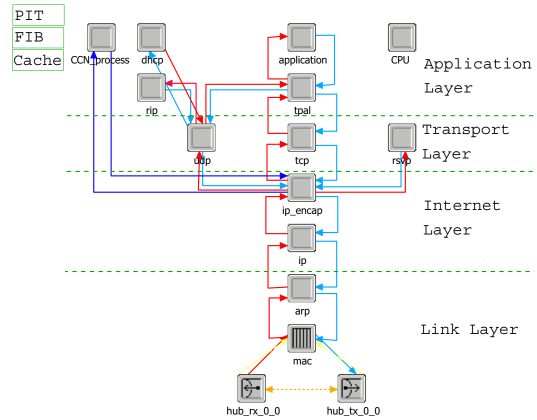

Figure 2: Procesor model of a simulated CCN node.

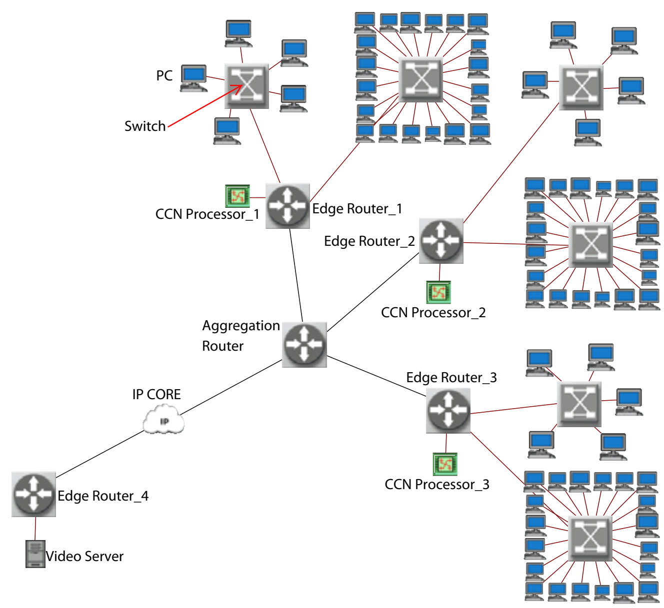

Figure 3: Envisioned network architecture.

all networks are built with three layers, i.e. *i*) the core layer provides optimal transport between core routers and distribution sites, *ii*) the distribution layer provides policy-based connectivity, peer reduction and aggregation, and *iii*) the access layer provides common group access to the internetworking environment. The simulated network includes three clusters of end users, distributed over a wide area network (WAN). Each cluster has an edge router, a CCN processor node, a switch and 25 PCs. All PCs request video content from a video server following a *Pareto* distribution: 20 PCs (80% traffic) request popular video content while the remaining five PCs (20% traffic) request unpopular video items. Videos are streamed from the video server through an IP core and then via aggregation routers, edge routers and finally received by CCN processors. After executing underlying video caching/replacement policies, CCN processors supply requesting PCs with requested video content when available at CCN nodes. In this paper, we apply FGPC and D-FGPC to edge routers to achieve the benefits of CCN [20] [21].

Table 2: Simulation parameters.

Element	Attribute	Value
WAN	Link between routers	OC-24 data rate
	Link for server	1000 BaseX
	Link for PCs	100 BaseT
Server	CCN root	ccnx://hust.edu.cn/epic/video/
	Publish root's name interval	100 seconds
	Number of video files ($\|F\|$)	500000 files
	A video size by default (F)	1 MB
	Practical video size (F^P)	0.5/ 1/ 1.5/ 2/ 2.5/ 3 MB
	The variety factor of F (α)	0.5/ 1/ 1.5/ 2/ 2.5/ 3
	Packet size	1024 bits
PCs	CCN directory	root/prefix1/.../prefix5
	File based popularity	*Pareto* principle
	Start time	100 + random(10) seconds
	Stop time	20000 seconds
	IntPk inter-arrival time	5 + random(2) seconds
	DataPk time-out	2 seconds
CCN node	Relative cache size (C^R_{size})	0.05/ 0.1/ 0.15/ 0.2/ 0.25/ 0.3%
	Replacement policy	LRU/ MPC/ FGPC/ D-FGPC
	The popularity threshold (P_{th})	5 (in case of MPC/FGPC)
	Hitting rate results sampling rate	0.1 Hz

The performance evaluation is conducted considering the impact of two metrics on the hitting rate; they are namely the relative cache size (C^R_{size}) and the factor of default file size (α) [22] [23]. The hitting rate for the three simulated edge routers is calculated as the ratio of the number of Interest packages (*IntPks*) satisfied at edge routers to the total number of *IntPks* from all PCs. Table II lists the values of important parameters considered in the simulations. These values were selected to reflect real-world implementations of in-network caching and that is considering prior research work [3] [22] [24] [25]. The simulations were run multiple times and the presented results are an average of these runs.

4.2 Simulation results

We first evaluate the performance of the different content caching/replacement policies for different cache sizes. The simulation duration is set to 20000 seconds. In the simulations, all PCs start sending *IntPks* from the 100^{th} second. In Fig.4, the hitting rate results are obtained for each 1000 seconds.

In Fig.4, the relative cache size (C^R_{size}) at CCN node is increased by 0.05%, 0.1%, 0.15%, 0.2%, 0.25% and 0.3%. It should be noted that in the simulations, the catalog size is large (500000 files) while the cache size is small (equal to or less than 1500 files). For this reason, the relative cache size is equal to or less than 0.3%. In Fig.4, the obtained simulation results show that high hitting rates can be achieved for high cache sizes and that is for all simulated policies. From the figure, it becomes apparent that this increase in the hitting rate is not linear to increase in the cache volume. The figure also indicates that when the relative cache size is equal to 0.25%, the cache can handle most requests for popular content. However, increasing the relative cache size to 0.3% degrades the performance gain. There is thus a tradeoff between cache volume (cost) and performance and there is consequently need to retrieve a suitable cache size.

Fig.4 also shows that in case of LRU, the hitting rate quickly increases when the cache becomes full. When the cache becomes overflowed, drawbacks of LRU happen and the hitting rate decreases gradually. This performance is mainly attributable to the intrinsic nature of LRU whereby caches store every new content. In contrast to LRU, MPC caches less and requires time to collect information about the popularity of content. Thus, we notice a slow increase of the hitting rate in case of MPC till it reaches a steady state. Incorporating the nice features of both LRU and MPC, the proposed FGPC scheme always stores new content when CCN nodes have available space. This explains the quick increase of FGPC's hitting rate. When the cache becomes overflowed, FGPC behaves similar to MPC and requires time to collect enough information about the popularity level of content. This feature results in a temporal decrease of the hitting rate (for a short time) before it continues increasing to the final state. Given the dynamic setting of the popularity threshold in D-FGPC, its hitting rate tends to be more stable, in comparison to FGPC, and that is during the entire simulation time as well as for several relative cache sizes.

Fig.5 compares among the performances of the four schemes when the relative cache size is set to 0.1%. The figure shows that D-FGPC outperforms all the other schemes, followed by FGPC, MPC and LRU, respectively. The figure also shows that LRU reaches higher cache hit ratio than others in the beginning of the simulations (i.e., until 6400*s*). The performance of LRU then degrades while other schemes achieve higher hitting rates. This is mainly due to the fact that D-FGPC, FGPC and MPC need time to assess the popularity of content before they start caching popular content.

Fig.6 further compares among the four schemes for different relative cache sizes. D-FGPC and FGPC exhibit always higher hitting

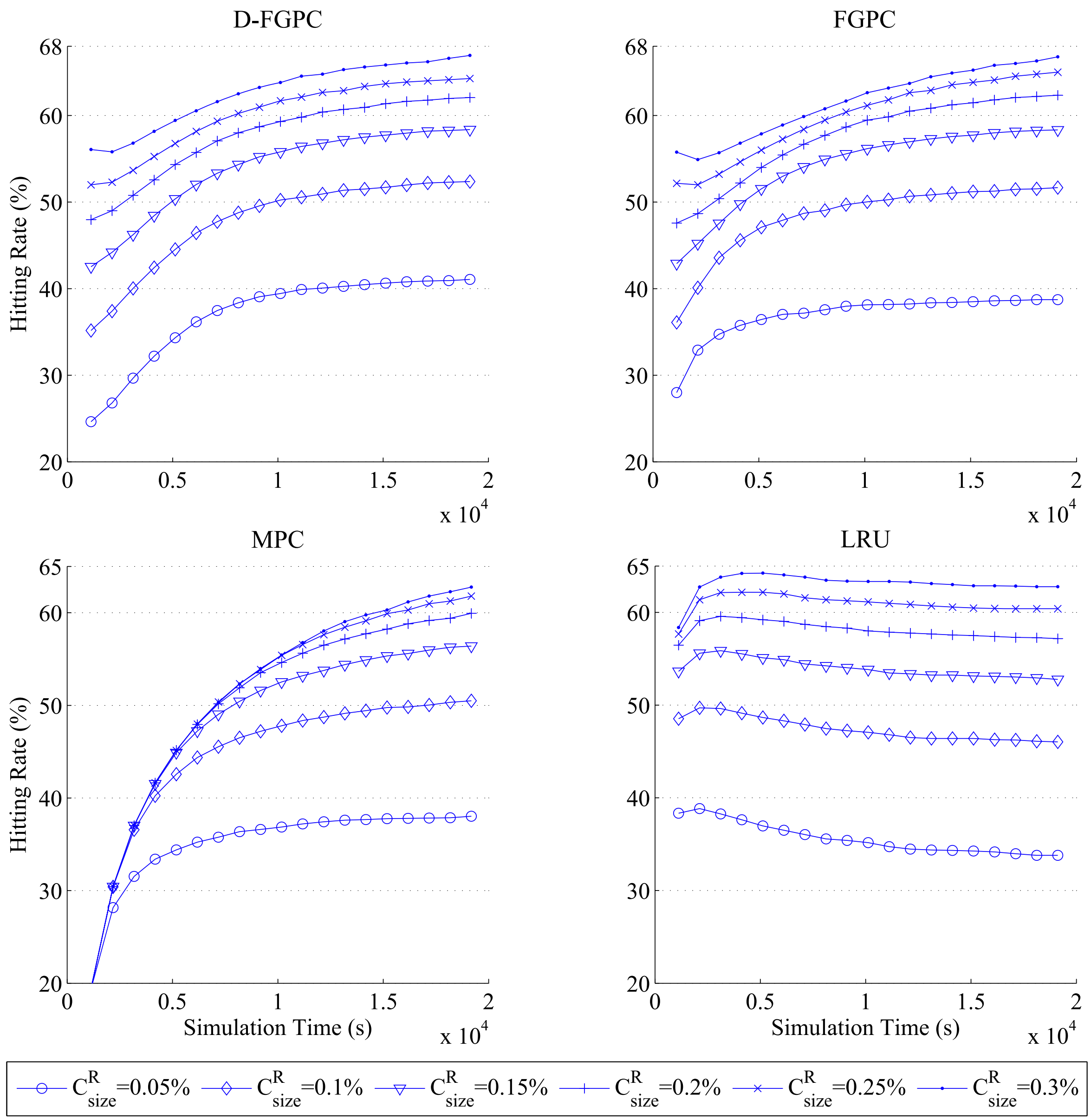

Figure 4: Performance of D-FGPC, FGPC, MPC and LRU for varying relative cache sizes.

rates in comparison to the conventional MPC and LRU schemes and that is in all situations. Furthermore, the figure reveals two important observations. First, when the relative cache size reaches 0.3%, the MPC performance quickly degrades and its hitting rate is just slightly higher than LRU. Second, with a dynamic threshold popularity value, the performance of D-FGPC is independent from the cache sizes. For example, when the relative cache size is small (e.g., 0.05%), the value of the popularity threshold is not optimal, making FGPC perform poorly in comparison to MPC. On the other hand, D-FGPC maintains always the highest hitting rate.

Fig.7 illustrates the effect of the factor of the default file size (α) on the hitting rate of the four simulated schemes and that is for a fixed relative cache size (0.15%). In this scheme, F^P is set to 0.5/1/1.5/2/2.5/3 MB in turn at the beginning of the simulation. It should be noted that despite the fact F^P may vary in real-life, we do not capture this in the envisioned simulations and that is for the sake of simplicity. D-FGPC exhibits the best performance, followed by FGPC, MPC and LRU, respectively. From the figure, it becomes clear that for a small file size (e.g., α= 0.5), the cache stores higher numbers of content. This trivially yields higher hitting rates. Furthermore, another reason is related to the number of arriving *IntPks* from all PCs. Indeed, in case of a stable data transmission rate, CCN nodes may take shorter times to satisfy requests for small-size files than in case of requests for large files.

5. CONCLUSION

In this paper, we introduced two variants of a new cache decision and replacement policy for CCN that takes into account content popularity. The performance of the proposed policy was eval-

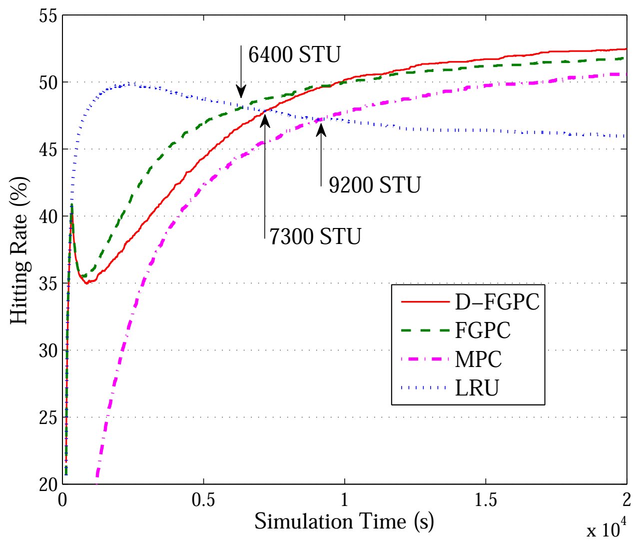

Figure 5: Multiple replacement policies with relative cache size (C^R_{size}) 0.1%.

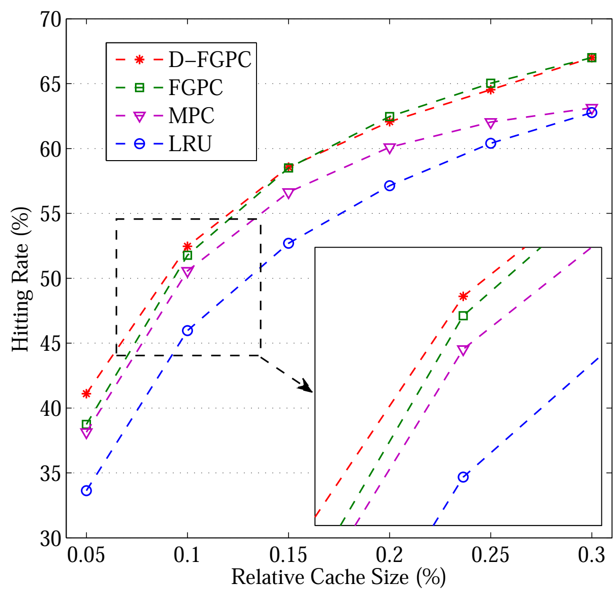

Figure 6: Final state of multiple replacement policies for varying C^R_{size}.

uated using computer simulations. The obtained results demonstrated the good performance of the proposed policy in achieving higher hitting rates and that is under different working conditions (e.g., cache size and file size). The performance of the proposed policy is expected to be largely improved with an efficient co-operation/contextual information (e.g., content popularity) sharing among neighboring CCN nodes to achieve self-organized networking of the CCN nodes and their caching. This defines one of the authors' future research directions with regard to CCN.

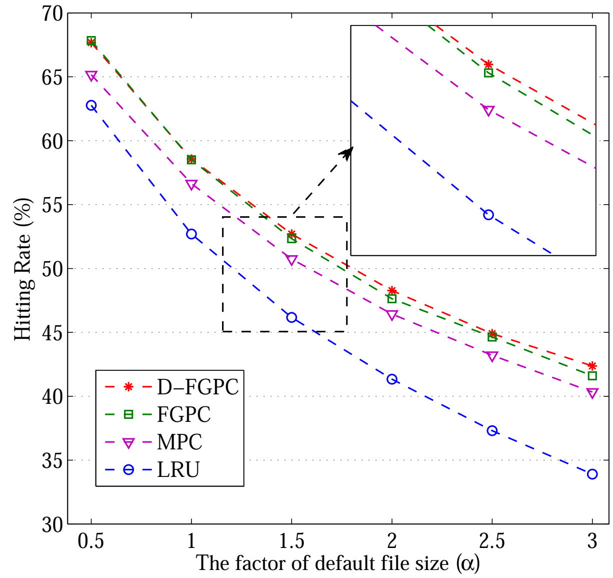

Figure 7: Final state of multiple replacement policies for varying values of α.

6. REFERENCES

[1] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content", *Communication of the ACM*, Vol. 55, No. 1, pp. 117-124, Jan. 2012.

[2] CCNx protocol. Available: http://www.ccnx.org

[3] C. Bernardini, T. Silverston, and O. Festor, "MPC: Popularity-based Caching Strategy for Content Centric Networks", *IEEE International Conference on Comminication (ICC2013)*, pp. 2212-2216, Budapest, Hungary, Jun. 2013.

[4] OPNET Modeler. Available: www.opnet.com

[5] M. Chen, "OPNET Network Simulation", Press of Tsinghua University, ISBN 7-302-08232-4, 2004.

[6] Named Data Networking. Available: http://www.nameddata.net

[7] NS-3 based Named Data Networking (NDN) simulator. Available: http://ndnsim.net/index.html

[8] B. Han, X. Wang, T. Kwon, Y. Choi, and N. Choi, "AMVS-NDN : Adaptive Mobile Video Streaming with Offloading and Sharing in Wireless Named Data Networking", *IEEE INFOCOM*, *NOMEN Workshop*, Turin, Italy, April, 2013.

[9] X. N. Nguyen, D. Saucez, and T. Turletti, "Efficient caching in Content-Centric Networks using OpenFlow", *INFOCOM 2013 Workshop Proceedings (2013)*, pp. 1-2, Feb. 2013.

[10] M. Xie, I. Widjaja, and H. Wang, "Enhancing Cache Robustness for Content-Centric Networking", *INFOCOM 2013 Workshop Proceedings (2012)*, pp. 2426-2434, Mar. 2012.

[11] Y. Li ,H. Xie, Y. Wen, and Z. Zhang, "Coordinating In-Network Caching in Content-Centric Networks: Model

and Analysis", *Distributed Computing Systems (ICDCS) 2013*, pp. 62-72, Jul. 2013.

[12] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S.Moon, "Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems", *IEEE/ACM Transaction on Networking*, Vol. 17, No. 5, pp. 1357-1370, Oct. 2009.

[13] N. Eisley, L. Peh, and L. Shang, "In-Network Cache Coherence", *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 321-332, Dec. 2006.

[14] I. Psaras, W. K. Chai, and George Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks", *ICN '12 Proceedings of the second edition of the ICN workshop on Information-centric networking*, pp. 55-60, Aug. 2012.

[15] T. Taleb, N. Kato, and Y. Nemoto, "Neighbors-Buffering Based Video-on-Demand Architecture", *in Signal Processing: Image Communication J.*, Vol. 18, No. 7, Aug. 2003, pp. 515-526.

[16] T. Taleb, Y. Hadjadj-Aoul, and K. Samdanis, "Efficient Solutions for Data Traffic Management in 3GPP Networks", *in IEEE Systems J.* (to appear)

[17] K. Samdanis, T. Taleb, and S. Schmid, "Traffic Offload Enhancements for eUTRAN", *in IEEE Communications Surveys & Tutorials J.*, Vol. 14, No. 3, Third Quarter 2012. pp. 884-896.

[18] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V.C.M. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems", *in IEEE Communications Magazine*, Vol. 52, No. 2, Feb. 2014. pp.131-139

[19] G. Li, M. Wang, J. Feng, L. Xu, B. Ramamurthy, W. Li, and X. Guan, "Understanding User Generated Content Characteristics: A Hot-Event Perspective", *IEEE Communications Conference (IEEE ICC2011)*, pp. 1-5, Kyoto, Japan, Jun. 2011.

[20] G. Tyson, S. Kauney, S. Miles, Y. El-khatibz, A. Mauthez and A. Taweel, "A Trace-Driven Analysis of Caching in Content-Centric Networks", *IEEE Conference on Computer Communications and Networks (ICCCN)*, Munich, Germani, 2012.

[21] A. Ghodsi, T. Koponen, B. Raghavan, S. Shenker, A. Singla and J. Wilcox, " Information-Centric Networking: Seeing the Forest for the Trees", *Procedure of 10th ACM Workshop Hot Topics in Networks*, Nov. 2011.

[22] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong, "Popularity-driven Coordinated Caching in Named Data Networking", *ACM/IEEE symposium on Architectures for networking and communications systems (ANCS)*, pp. 15-26, Oct. 2012.

[23] Z. Ming, M. Xu, and D. Wangy, "Age-based Cooperative Caching in Information-Centric Networks", *IEEE Conference on Computer Communication Workshops (INFOCOM)*, pp. 268-273, Mar. 2012.

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

[25] D. Rossi, G. Rossini, "A dive into the caching performance of Content Centric Networking", *Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2012 IEEE 17th International Workshop on*, pp. 105-109, Sept. 2012.