New Caching Algorithm in Cognitive D2D Networks

Mustafa Enes ÇAKIR

December 31, 2018

Abstract

My algorithm uses count of contents on device that closer than R_d and their popularity. Device tries to distribute caches densities evenly looking nearest device's cache contents.

1 Introduction

Nowadays, most of the Internet traffic is consumed by video services. YouTube is very popular among teenagers. Generally, famous YouTubers have more than a million subscribers. So when he/she share a new video, thousands of people watch it at the same time. Centralized solutions create huge pressure on data centers. Video streamers try to store videos close to watchers because they want to reduce overall data traffic and give more smooth experience to their users. In addition to that, the popularity of old videos changes with new ones very quickly. When a video goes viral, billions of people watch it in a short period of time. So cache contents have to change dynamically. In addition, no need to wait to finish downloading for users to watch videos. They can start watching videos when packet started to come.

As it is seen, old web-caching solutions are not solving video stream caching issues. Because web-caching is designed to be centralized. The user has to download the whole object for using it. They store them. But videos are consumed in a comparatively short time span. Information-Centric Networking allows storing cache on every device that connected to network. Why we store every content in the cache? Because caching cost the money. We have limited cache capacity. The new replacement algorithm

2 Algorithm

First of all, I think people who have similar lifestyles live close to each other. Replacing old videos depend on local popularity looks like applicable to me. But we do not have a parameter for closer user choose similar videos. So I give up on this idea. On the other hand, I liked the idea of looking near devices. In addition to that, the difference between the average enhancement layer size and the average base layer size is considerable. If we want to replace an enhancement layer with a base layer, we have to delete five of them. It is not very efficient. In my algorithm, the device looks cache content of the device that are closer than R_d (see figure 1). We have N different contents. Each user checks average N other users. N/λ_{user} is the average area that N users are located. If we write the equation for R_d.

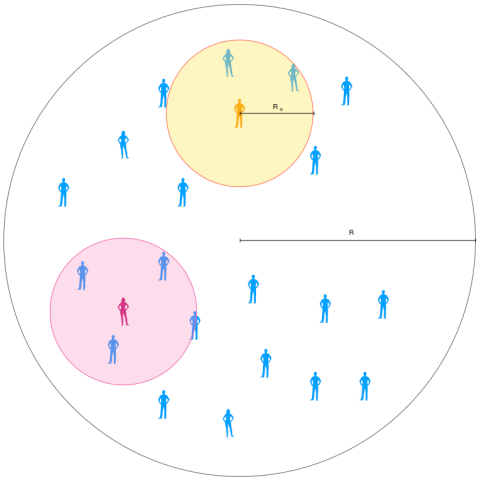

Figure 1: The diameter of device, R_d

$$R_d = \sqrt{\frac{N/\lambda_{user}}{\pi}}$$

For our model, R_d is approximately 146m. It counts each content separately. If newly arrived content is an enhancement, it looks for enhancement counts, vice verse for the base. If there is no enhancement, it uses base layers for replacement. Following logic is for choosing an item to be deleted. The algorithm divides content counts by their popularity for calculating their weight.

$$W_k = \frac{count_k}{popularity_k}$$

It replaces with the biggest weighted one. It uses division instead of multiplication because it gives an advantage to popular contents. A higher weight means the content has more cache in this area. The device can reach it with small distances. So the algorithm tries to balance the density of the cache in each area. This is not very homogeneous because of popular ones allowed to be denser. If the weight of videos are equal, remove less popular one.

Our device has approximately N neighbor devices and the device has M cached content. It will be look N devices for M content. So our time complexity is $\mathcal{O}(NM)$.

3 Related Works

Designing an efficient caching algorithm for Information-Centric Networks is a very important topic. Malthusian spectre (see figure 2) is a prediction that population growth will outpace agricultural production. If we think about our topic, we have limited hardware and information traffic that growing too fast. We need to move wisely. The other works are a more general solution than ours. We think about just the replacement part of the caching algorithm, but they include the probability of storing the cache of content. Sometimes they do not replace with old ones. On the other hand, we always saving new cached and replacing old ones. In addition, we are using Device-to-Device connections for transferring data. We don't think about hops counts. Also, they don't have a different quality version of the same content except [6].

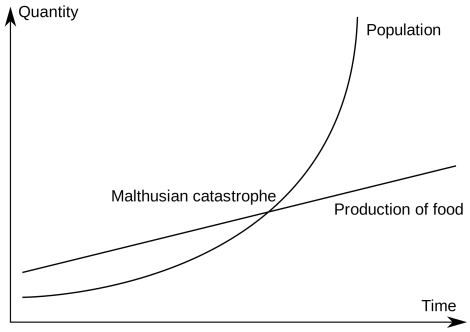

Figure 2: Malthusian spectre

[1] talks about ProbCache that is a probabilistic algorithm. It calculates caching capability of a given path per unit time. They try to decrease caching redundancy. They are not interested in the popularity of videos. They just care about the path that data follows. In contrast, our algorithm does not care path, because it's a D2D direct connection.

[2] is about chunk-based caching (CC) technique for videos. It's very smartish. While a user is watching videos, the next chunk has higher priority. It looks like obvious, but I had not thought that way. I like it. It uses the popularity of videos like our algorithm, but combine it with chunk scores. We do not chunk our data like them. They developed a more modular caching algorithm.

[3] proposed PopCache algorithm and [4] proposed WAVE algorithm. Both depend on the popularity of videos. Also, WAVE dynamically adjusts the number of chunks to be cached depending on the popularity of files. They just use general popularity, but we also look local popularity.

[5] has more focused solutions for 5G Systems. It looks like very hardware related. Both [4] and [5] use D2D. They more similar to our model than other papers. [5] has different quality versions for the same content like our model.

Also, I read [7] and [8] papers. [7] is about local popularity. It calculates expected local popularity of data based on universal popularity. [8] compares different replacement algorithms for video services.

References

[1] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proceedings of the Second Edition of the ICN Workshop on Information-centric Networking*, ser. ICN '12. New York, NY, USA: ACM, 2012, pp. 55–60.

[2] D. Hong, D. De Vleeschauwer, and F. Baccelli, "A chunk-based caching algorithm for streaming video," in *NET-COOP 2010 - 4th Workshop on Network Control and Optimization*, Gent, Belgium, Nov. 2010, session 05 : Streaming applications.

[3] K. Suksomboon, S. Tarnoi, Y. Ji, M. Koibuchi, K. Fukuda, S. Abe, N. Motonori, M. Aoki, S. Urushidani, and S. Yamada, "Popcache: Cache more or less based on content popularity for information-centric networking," in *38th Annual IEEE Conference on Local Computer Networks*, Oct 2013, pp. 236–243.

[4] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "Wave: Popularity-based and collaborative in-network caching for contentoriented networks," in *2012 Proceedings IEEE INFOCOM Workshops*, March 2012, pp. 316–321.

[5] 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," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131–139, February 2014.

[6] H. Zhu, Y. Cao, W. Wang, B. Liu, and T. Jiang, "Qoe-aware resource allocation for adaptive device-to-device video streaming," *IEEE Network*, vol. 29, no. 6, pp. 6–12, Nov 2015.

[7] Kumari Nidhi Lal,and Anoj Kumar, "A Cache Content Replacement Scheme for Information Centric Network", *Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016)*, Motilal Nehru National Institute of Technology, Allahabad 211 004, India

[8] Areej M. Osman, and Niemah I. Osman, "A Comparison Of Cache Replacement Algorithms for Video Services", *International Journal of Computer Science & Information Technology (IJCSIT)*, Vol 10, No 2, April 2018