Towards Popularity-Based Caching in Content Centric Networks

Cesar Bernardini*†, Thomas Silverston*† and Olivier Festor†
*Université de Lorraine, LORIA, UMR 7503, Vandoeuvre-les-Nancy, F-54506, France
†Inria, Villers-les-Nancy, F-54600, France
Email: {name.surname}@inria.fr

I. Introduction

Today's Internet was created aiming to share resources. The chosen model was therefore a conversation mechanism between two machines: the messages are exchanged between two host according to source and destination addresses of IP datagrams. This led to a model with strong coupling between *where* and *what*, even when most of the users value the Internet for *what* content it contains. As well location dependence makes configuration and implementation of network services more complex. Alternative architectures such as CDN and P2P networks provide high availability and improve performance but they are not-supported natively by the current Internet. In order to solve these issues, Information-Centric Network architectures (ICN) have been proposed such as PSIRP[7], NetInf[1], or CCN[2]. These architectures are thought as clean-state designs for the Future Internet.

Content Centric Networking (CCN) is a communication architecture based on named data where packet address name content, not location. The host notion does not exist anymore. The communication paradigm relies on two primitives: *Interest* and *Data*. A consumer requests content by broadcasting its interest all over the network; any node hearing the request and having the data can issue a response with a data message. In addition, CCN offers routing-by-name, caching, integrity checks based on signatures, encryption, multicast support, flow-control at each hop, protection against Denial of Service attacks.

Caching consists in storing data messages at a node. In CCN, data messages are cached at every intermediate nodes from a source to a destination. A content is therefore replicated and is available at a closer distance for future requests. An important feature for CCN is to manage the cache of nodes by defining replacement policy, cache policy and cache strategy[4].

Most of the research about information centric networks, have been carried out on CCN. For instance, Kurose et al.[3] show that cache replacement policies can be grouped in the same equivalence class. Rossi et al.[6], [5] study cache sizes and compare several caching strategies. They also argue that popularity of content have a strong impact on CCN performances[4].

In this extended abstract, we propose a new caching strategy based on popularity. Section II describes our proposal in detail. Then, Section III relates to our ongoing work and finally, we end up with future work at Section IV.

II. Popularity-Based Caching Strategy

As we explained before, every time a CCN node issues an interest message, it is broadcasted through the network until a node hearing the request is able to reply. Instead of the CCN caching strategy where all the nodes cache every content message that has passed by them, we believe that caching only popular content will help to save resources and to improve the overall performances of the CCN network. To this end, we propose a new caching strategy called *Most Popular Content* (MPC), where nodes will cache only popular content. Purging unpopular content from caches will help in several ways: more resources will be available for popular content; unpopular content will not increase the workload of replacement caching policies; Unpopular content will not be cached, and for these, CCN network will act as IP does in the current Internet where a request is forwarded to the node owning the content.

A content is defined as popular when it has been requested (through interest messages) a certain number of times reaching a *popularity threshold*. Every node maintains therefore a local popularity table and counts all the interest messages it receives for any particular content. When a content become popular, the CCN node holding the content suggests his neighbor nodes to cache the content. Besides, after receiving suggestion messages, neighbor nodes may accept to cache the content or not. Moreover, the popularity of a content can decrease with time, as it may be less popular in the future.

More precisely, our new caching strategy works as presented in Figure 1. Figure 1(I) describes a typical scenario, and the final state of cache at every node according to our strategy, Fig.1(II), or CCN, Fig.1(III). There is a popular content *d1*, initially stored at node D, and an unpopular content *e1* at node E. In our scenario, the popular content *d1* will be requested by three nodes A, B, C. The unpopular content *e1* will be requested only once by node A.

Figure 1(II) shows how our strategy works with the proposed scenario. When node A sends an Interest Message for content *e1*, e1's popularity is increased in the popularity tables of every node along the path [A, C, D, E]; in other words, *e1* content popularity at nodes A, C, D and E is now set to 1. In the same way, when node A sends an Interest Message for

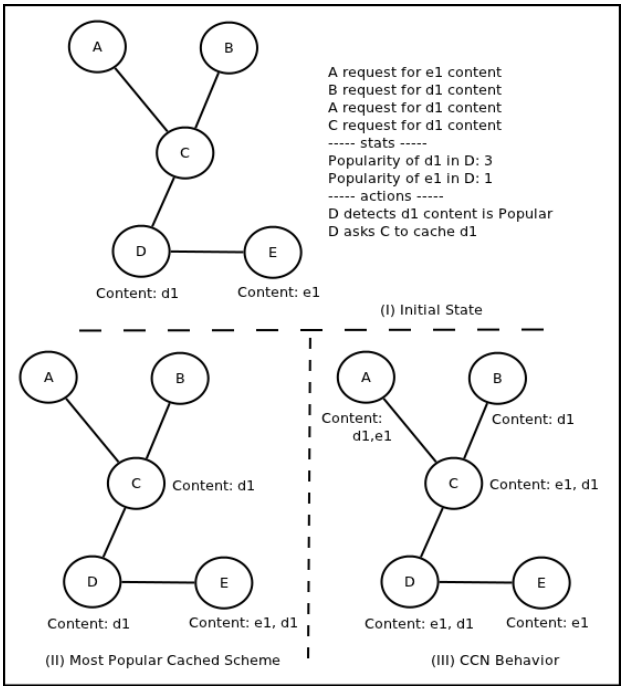

Fig. 1. (I) represents an example of a network and a representative scenario; (II) depicts our proposed caching strategy with the scenario; (III) shows the normal CCN behavior with the same scenario.

content *d1*, the popularity of *d1* is set to 1 along the path [A, C, D]. Since content *d1* is a popular one, B and C send in turn an Interest Message for content *d1*. The interest message from B will increase d1's popularity to 2 at C and D (through the path [C, D] to D). Finally, C requests the content *d1*, increasing d1's popularity to 3 at node C and D.

At this moment, node D is the only node to hold the content d1. D spreads suggestions messages to his neighbors C and E. Node C and E will therefore cache the content and will be able to transmit the content after upcoming requests. For instance, if there are interested nodes close from A, their request will be redirected directly to C and not D, reducing the number of hops to get the content.

Figure 1(III) describes the CCN case. When the first request is sent by node A, content *e1* will be cached at every nodes along the path [A, C, D, E]. Then, according to our scenario, content *d1* will be cached around [A, C, D] after the first request from A. The request from B will be directed only through C and *d1* will be cached into B. Finally, node C has already the content since it has been stored from previous requests.

The main difference between our strategy and the CCN's one is that the CCN behavior caches all content, and it will overload resources at every node, even though that content is not popular. For instance on Fig. 1(III), content *e1* is replicated at nodes A, C and D and content d1 is on nodes A, C and B. Differently, with our strategy on Fig 1(II), *e1* is only present at node E and *d1* is replicated but only to nodes C and E. Clearly, our proposed strategy (Fig.1(II)) is saving resources such as memory, bandwidth or the number of caching replacement operations compared with the CCN strategy.

III. Ongoing Work

In the previous section, we presented MPC, our new caching strategy for CCN network. We are currently evaluating MPC by implementing it into the ccnSim simulator[4]. Our implementation includes all the features to support our proposed strategy, as for example, popularity tables and *suggestion* messages. In order to evaluate our strategy properly, we are investigating its key parameters such as popularity threshold, replacement policy for popular content, or popularity table size. We are also refining our strategy according to the feedback we have from our simulations. For instance, we could adjust the behavior of nodes when receiving a suggestion message and define an acceptance policy if nodes cache the content or not.

We intend to analyze this strategy thoroughly with optimal parameters values and as well to compare it fairly to current cutting-edge CCN strategies. Thus, we will introduce new metrics such as the percentage of claimed cached elements, the number of hops between the content and the receiver, the number of replicas in the entire network, or the lifetime of a content cached in a node. By using ccnSim and our new strategy, we will be able to compare different caching strategies and to point out that popularity caching may outperform other strategies and is well suited with CCN.

IV. Future Work

We stand strongly for MPC, and we expect that it could be a startup for studying name-based routing. Since it is a suggestion based mechanism, it could be adapted to manage content among nodes, predict popularity and routing content to destination. Beyond, the popularity of content is also a piece of information that can be obtained from social networks. This feature may help improving the accuracy of our strategy.

References

[1] C. Dannewitz, E. Bauer, M. Becker, F. Beister, N. Dertmann, M. Kionka, M. Mohr, F. Steffen, S. Stey, and S. Weber. OpenNetInf documentation design and implementation. Technical report, June 2010.

[2] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, 2009. ACM.

[3] Elisha Rosensweig, Daniel Menasche, and Jim Kurose. On the Steady-State of Cache Networks. Technical report, July 2011.

[4] Dario Rossi and Giuseppe Rossini. Caching performance of content centric networksunder multi-path routing (and more). Technical report, Telecom ParisTech, 2011.

[5] Dario Rossi and Giuseppe Rossini. A dive into the caching performance of content centric networking. Technical report, Telecom ParisTech, 2011.

[6] Dario Rossi and Giuseppe Rossini. On sizing ccn content stores by exploiting topological information. In *NOMEN 2012*, 2012.

[7] M. Särelä, T. Rinta-aho, and S. Tarkoma. RTFM: Publish/subscribe internetworking architecture. ICT Mobile Summit, 2008.