Network Research Workshop
Proceedings of the Asia-Pacific Advanced Network 2014 v. 38, p. 55-59.
http://dx.doi.org/10.7125/APAN.38.8   ISSN 2227-3026

GA for Popularity Based Cache Management in ICN

Alaa Mohammed A.A. [1] ,and Koji Okamura[2]

1 Alaa Mohammed A.A. / Department of Advanced Information Technology, Graduate school of Information Science and Electrical Engineering, Kyushu University, Japan.
2 Koji Okamura / Research Institute for Information Technology, Kyushu University, Japan.

E-Mails: alaa_83moh@yahoo.com, oka@ec.kyushu-u.ac.jp

Tel.: +81-80-8396-9151

Abstract: Information Centric Networks (ICNs) is a new architecture for the Future Internet to deliver content at large-scale. It relies on named data and caching features, which consists of storing content across the delivery path to serve forthcoming requests. In this paper, we study the problem of finding the optimal assignment of popular contents in the available caches storage in ICN. We formulate this problem as a combinatorial optimization problem. Metaheuristic methods are considered as effective methods for solving this problem. We will adapt cache management system based on GA for solving the considered problem in order to minimize overall network overhead.

Keywords: Information Centric Networks (ICNs); Genetic Algorithm (GA); Network of Information (NETINFO).

---

1. Introduction

Initially, the Internet was designed for host-to-host communication, users are only interested in actual content rather than source location. Hence, new Information-Centric Networking architectures (ICN) have been proposed in these architectures the communications are more centered on content without caring much about where it is physically located [1].One of those architectures is NetInf [2], there are two models for retrieving content, via name resolution and via name-based routing. If an NRS is available, a receiver can first resolve an NDO name into a set of available locators and can subsequently retrieve a copy of the data from the “best” available source(s). Alternatively, the receiver can directly send out a GET request with the NDO name, which will be forwarded toward an available NDO copy using name based routing. As soon as a copy is reached, the data will be returned to the receiver. Our proposed algorithm will work with the architectures that are depend on installing local and global servers for name resolution service (NRS) like PURSUIT and NetInf architectures.

Related work: The author in [3] depend on the strategy of cache everything in everywhere. In [4] the author define random caching strategy, “Rdm+LRU” which simply caches randomly at only one intermediate node along the delivery path. Finally in [5] the author design “Most popular content” MPC as a new cache management strategy, where nodes cache only popular content. As some of proposed algorithms for cache management take cache decision locally and surely if the cache decision is made at large scale it will be better than local decision [5]. So, our GA algorithm will be a type of central cache management algorithms for finding near optimum locations for caching popular contents. The rest of the paper is organized as follows. Section 2 presents GA. While in Section 3, we discussed results. Finally, Section 4 conclude our work.

2. GA for cache management in ICN

2.1. Problem Formulation

We consider a network where the nodes are the Autonomous Systems (ASs). For simplicity, we assume that one AS is the same as one ISP node. We have N of AS in the network and AS_i, i 1, 2,……..,N, has k_i bytes of storage capacity. The storage capacity of AS_i is:

$$k_i = \sum_{i=1}^{L} s_i \tag{1}$$

Here L: the number of routers in ASi and Si: the cache size of router i. And the total storage capacity k is the summation of storage capacity of each AS. The objective of this algorithm is to find optimum cache locations of m different content files (popular ones) such that network overhead can be minimized. The network overhead can be computed as follows:

$$O_{net} = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, d_{ij} \tag{2}$$

Here p_{ij} is the popularity of the file j at node i, and d_{ij} is the shortest distance between node I and requested file j. in this algorithm we will suppose that the distance is measured as the number of hops between the source and destination nodes. The subject is to minimize the overall network overhead given in equation 2, in order to satisfy the following constraints:

1- The number of copies of each object content stored at any node should be either 0 or1.

2- Each object content should be cached on at least one node (AS)

2.2. The proposed algorithm

Our proposed algorithm is based on using NetInf architecture [4]. The popularity of objects is calculated based on the number of request at each local NRS, so global NRS can find global objects popularity. The proposed GA will ran at global NRS in offline mode by this way we can save the time to process for a large number of content objects. After each run of our algorithm the popularity counter set to reset value and after the new popular contents reach to predefine threshold our GA work again, by this method we can satisfy the dynamic change in the popularity.

2.2.1. *GA overview*

Genetic algorithms generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover Figure [1] show the steps of GA. The solutions of the proposed GA can be formulated as a matrix shown in Figure [2], where the element x_{ij}=l if the file i is stored at the node j, else 0. For evolution we used fitness function. The fitness function for each chromosome is given by the Inverted of function presented in equation [2]. Crossover and mutation operations are shown in figure [3] and figure [4]. Tournament Selection is used as selection operation in our GA.

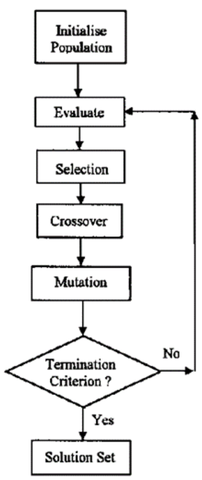

Figure 1. Overall GA

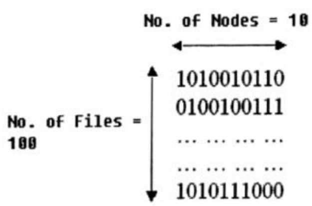

Figure 2. Representation of the solution

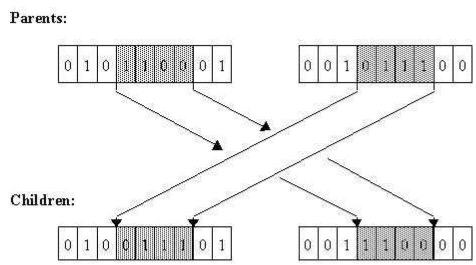

Figure 3. Crossover operation

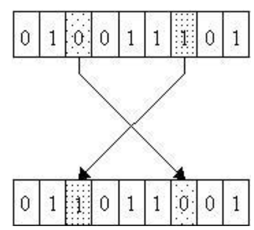

Figure 4. Mutation operation

2.2.2. How GA select good solution by example

For simplicity, it is supposed the case of caching one file for network of 6 node. And we supposed two randomly solution and try to show if GA can decide which one is better than other one in order to minimize the total network overhead. Equation [2] used to calculate the network overhead. For the first solution in figure 5.1 the network overhead is O_{net} = 40, but in the second solution in figure 5.2 it is found that O_{net} =50. We can see that the first solution is better than the second solution.

3. Results

In our simulation we will use the NetInfo open source implementations "OpenNetInf [6]". We assume that the network consist of 20 router divided to 10 AS_s. And there are 50 client connected request each object of M popular content, the request order is random. We measure all traffic passing through the respective router interfaces including any management traffic and other overhead. As we can see form figure 6 that cache in everywhere is better when the number of content is small because in this case the content will cache closer to client and the live time of content will be longer, but with increase the number of objects we observed that our GA is better than because the live time of contents in the first algorithm will be smaller than our algorithm and surly the content will be retrieve from other locations more than our GA that will have lc

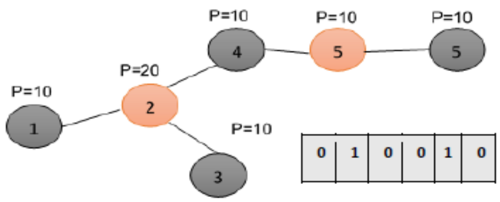

Figure 5. (1) This solution means that this content will be cached at nodes 2, 5

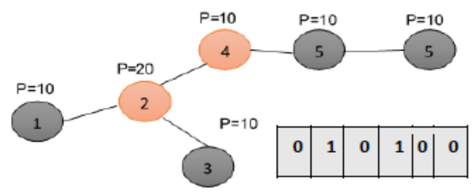

Figure 5. (2). this solution means that this content will be cached at nodes 2, 4

// M: represent the selected popular objects
// U_i : a set of objects that will cache in AS_i and k= $|u_i|$ is the number of objects
// OB: Object file
***Step 1**: for global cache assignment in this step we assigned a set of object for each AS*
1- Randomly initialize S solution as a population of GA
2- Evaluate the S solutions using equation 2
3- Select new S solutions by applying tournament selection
4- For i=1 to S/2
Apply crossover operation randomly
5- For i = 1 to S
Apply mutation operation randomly
6- Go to step 2 till termination condition
***Step 2**: for local cache assignment in this step each object is assigned for specific router for each AS*
7- For i = 1 to k
If OB_i is not cached in neighbors AS_s
Then cache it in BG router
Else cache it in the router that have largest number of request (popular assignment)

Steps of proposed GA

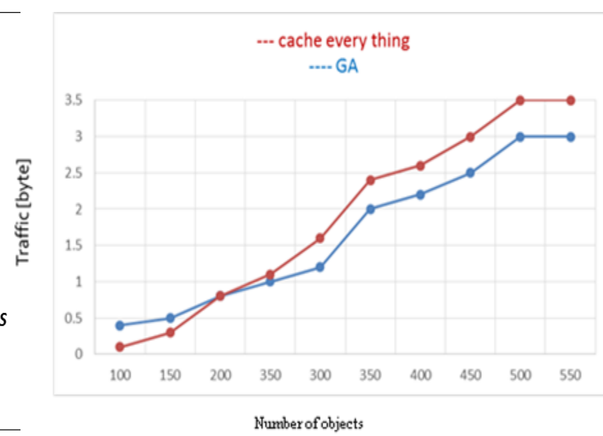

Figure 6 simulation result

4. Conclusions

This paper introduce a new central cache management algorithm for ICN based on GA algorithm in order to minimize the total overall network overhead, this algorithm in contrast to some traditional locally caching systems will support both local and global searching for best locations to cache contents in available cache storage. The result show that our GA can find good solution.

References

1. http://www.named-data.net/
2. B. Ohlman et al, “First NetInf architecture description,” Apr. 2009.
3. Jacobson, V.; Smetters, D. K.; Thornton, J. D.; Plass, M. F.; Briggs, N.; Braynard, R. Networking named content. *Proceedings of the 5th ACM International Conference (CoNEXT 2009)*; 2009 December 1-4; Rome, Italy. NY: ACM; 2009; 1-12.
4. Wei Koong Chai, Diliang He, Ioannis Psaras, George Pavlou, Cache “Less for More” in Information-Centric Networks , *NETWORKING 2012* , Volume 7289, 2012, pp 27-40.
5. César Bernardini, Thomas Silverston, Olivier Festor “Cache Management Strategy for CCN Based on Content Popularity”, *AIMS - 7th International Conference on Autonomous Infrastructure*, Management and Security 7946 (2013) 92-95.
6. http://www.netinf.org/