Proc. of the International Conference on Electrical, Computer and Energy Technologies (ICECET)
9-10 December 2021, Cape Town-South Africa

LADA: Locality Aware Distributed Addressing for Edge/Fog Computing Infrastructures

Mohammed B. Alshawki
Eotvos Lorand University
Budapest, Hungary
Furtwangen University
Furtwangen, Germany

Peter Ligeti
Department of Computeralgebra
Eotvos Lorand University
Budapest, Hungary
ligetipeter@inf.elte.hu

Christoph Reich
Faculty of Informatics
Furtwangen University
Furtwangen, Germany
christoph.reich@hs-furtwangen.de

2021 International Conference on Electrical, Computer and Energy Technologies (ICECET) | 978-1-6654-4231-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICECET52533.2021.9698791

***Abstract*—In edge/fog computing infrastructures, the resources and services are offloaded to the edge and computations are distributed among different nodes instead of transmitting them to a centralized entity. Distributed Hash Table (DHT) systems provide a solution to organizing and distributing the computations and storage without involving a trusted third party. However, the physical locations of nodes are not considered during the creation of the overlay which causes some efficiency issues. In this paper, Locality aware Distributed Addressing (LADA) model is proposed that can be adopted in distributed infrastructures to create an overlay that considers the physical locations of participating nodes. LADA aims to address the efficiency issues during the store and lookup processes in DHT overlay. Additionally, it addresses the privacy issue in similar proposals and removes any possible set of fixed entities. Our studies showed that the proposed model is efficient, robust and is able to protect the privacy of the locations of the participating nodes.**

***Index Terms*—DHT, Distributed scheme, Location aware scheme**

I. Introduction

The nodes in the storage systems with centralized managing entity store the data in a trusted third party such as a cloud server that usually has a high computation and storage power. The fully centralized scheme affects negatively the efficiency as the system grows. Edge/fog computing scheme, comparing to the cloud computing tries to distribute the computations among many edge nodes. One of the challenges in the edge/fog computing is that how to manage the addressing and allocation of the participating nodes and their services and resources, as a centralized managing entity might turns into the bottleneck in the system. This issue can be addressed by using distributed techniques such as Distributed Hash Tables (DHT), as it allows the participating nodes to locally generate their own identifiers and be positioned in the network based on their generated identifiers.

DHT creates a logical overlay on top of the physical underlay network and position the nodes in the logical overlay based on their identifiers. The identifiers are generated as a result of hashing an information using a collision resistant hash function and using its output as the identifiers of the nodes. Although the randomness of the hash function is one of the required features in DHT, it may cause some efficiency issues if used without considering the physical location of nodes.

The location mismatch between the underlay network and DHT overlay in terms of locations of nodes leads to some inefficiency in accessing the nodes. The reason is the difference in the closeness values between underlay and overlay. This difference happens because the physical path of two underlying nodes and their closeness value is different from the logical path of those nodes in the DHT overlay and their closeness. Some researchers proposed solutions that remove this mismatch by creating a connection between the underlay and the overlay to address this issue. These proposals utilize different techniques such as the Autonomous System Numbers (ASNs) of the participating nodes [1], distance to the known landmarks in the system [2] or clustering [3]. Each of the proposed solutions has its advantages that help to overcome the DHT locality issue. On the other side, some privacy and efficiency issues have to be addressed in the edge/fog computing infrastructure. Locality Aware Distributed Addressing (LADA) proposes a locality aware overlay that aims to overcome the privacy and efficiency issues in the DHT overlay. We presented a preliminary and abstract version of this paper at MaCS2020. In this paper, we describe and study the processes and effectiveness of Region-based DHT [4]. The main contributions of our paper are as following:

- A locality aware overlay that can be applied in edge/fog computing infrastructure.
- Privacy preserving addressing approach for private participants.
- Removing any centralized organizing entity or predefined members during overlay creation.

This research has been partially supported by Application Domain Specific Highly Reliable IT Solutions project which has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme) funding scheme, by the European Union, co-financed by the European Social Fund EFOP-3.6.3-VEKOP-16-2017-00001.

The rest of this paper is organized as follows. In the next section the DHT is briefly explained and some DHT based proposals are reviewed. In section three, LADA and its different components are explained. Section four analyses the proposed model and discusses the results. Finally, in section five our conclusions are listed.

II. Literature Review

Using DHT as a distributed storage technology, a logical overlay of nodes can be created on top of the underlay network. The resulted overlay and its features are used in different fields such as decentralised resource discovery models [5], [6] as an alternative to the trusted third party schemes. There are a number of implementations such as Pastry [7], Chord [8], Kademila [9] and Tapestry [10] that can be used to create a DHT overlay in a given environment. There is no centralized organizing entity that controls the joining and leaving processes of the nodes. Any node in DHT can be join freely the created DHT overlay by using a collision-resistant one-way hash function that is used in DHT for both node allocation and data responsibility definition. Upon joining the DHT overlay, a new node uses the hash function to create its own seemingly unique identifier. This is done be feeding the hash function with some unique information such as the IP address of the node, and using the output of the function as the identifier of the node. The generated identifier indicates the logical position of the newly joined node in the overlay. Since the physical location of nodes is not considered during the identifier generation, there is no match in term of closeness between the physical underlay network and the logical DHT overlay.

As in hash tables [11], the DHT is used to store and then retrieve the data by using a hash function. DHT is designed to work in a distributed scheme. The stored data in the DHT consists of a pair of key/value parameters. The key parameter is generated by feeding an attribute of the stored data such as its name or type to the hash function. The value parameter includes the necessary information to find the actual location of the required data. The same hash function can be used for both key generation in the key/value pair and the identifier generation of the participating nodes. This makes the mapping between the stored pair and the nodes works by matching the keys with the identifiers of the nodes in the overlay. If no exact identifier during matching process has been found, a closeness function will be used to determine the identifier of a close node in the DHT overlay that is responsible for storing the pair of key/value. Later, a similar process will be used to retrieve the stored value of a given key. It is noteworthy to mention that for n participating nodes in the overlay, it takes at most $O(log(n))$ steps to find a node that is responsible for storing a specific data based on the key parameter of the stored pairs. This feature makes the DHT scalable and efficient when it implements in large scale networks. However, ignoring the physical location of nodes during overlay creation [7]–[10] adds a significant delay to the applications that run over it [5], [6]. To address this issue some researchers proposed solutions that removes this mismatch by creating a connection between the underlay and the overlay.

Wu et al. [1] proposed a model called locality aware DHT that uses ASNs of the participating nodes to create the first part of the identifiers. This guarantees that the nodes with the same autonomous system number will have close positions as their identifiers will have similar prefixes. Its dependence on the IP addresses of the participating nodes lead to different identifier when the IP address of the node modifies. Toda et al. [3] proposed a model of identifier generation for SkipGraph, a DHT based implementation for data storage, that generates the identifiers of the generated overlay based on a number of clusters. The clusters are created based on a predefined list of hotspots. The newly joined node gets a randomized identifier that might be modified based on the locality information. By joining a new node and assigning an initial random identifier, likely the identifiers of $Olog(n)$ nodes may be changed. This procedure requires a significant traffic by joining any new node to the system, that specifically affects the systems with churn.

Authors in [2] proposed LANS, a locality aware identifier assignment algorithm that is based on known landmarks. The landmarks are defined prior to the start of the system at the initialization phase. The identifiers of nodes consists of prefix and body parts. At system setup the landmarks, that are considered churn free, are assigned with a dynamic prefixes. The newly joined node contacts the landmarks and computes its coordinates based on the latency of the communication to compute the prefix part of the identifier based on the latency between the newly joined node and each of the landmarks. The known points are considered to be fixed and can not be updated during the system operation. In addition, in the models that include a predefined set of known points, the known points might turn into single point of failure and attack. The main feature that LADA adds to distributed addressing is locality aware creation of the overlay. It does that considering the privacy of the participating nodes, without requiring a predefined and fixed list of nodes, a centralized organizing node or a need to update of any subset of nodes' identifiers.

III. Locality Aware Distributed Addressing

LADA uses a collision-resistance hash function that generates a d-bit digest output to generate the identifiers of the participating nodes. The identifier of a node in LADA consists of three parts of representative region, sub region and local data. In the following sections, the sub-regions and regions sets, the different types of lookup, the approaches of identifier generations and the parallelism and replications are discussed.

A. Sub-regions:

LADA consists of local regions (*sub-regions*) that are grouped into a number of *regions sets*. Each region set includes a number of sub-regions, among which, one *representative sub-region*. The model uses a hash function that generates a fixed d bits digest out of any given input. LADA consists of maximum $2^{d/2}$ region sets, each with a maximum of $2^{d/2} - 1$ sub-regions and one set representative sub-region. A region

set does not have to reach the maximum number of sub-regions, and can be created even with one sub-region, namely the set representative sub-region. The region sets are created based on any group of geographical regions. In addition to the region sets, the special region sets include two regions, *private* and *public* regions. Although it is optional for a node to be part of its geographically close sub-region, any node in LADA has to be part of these two special regions. The private region is used to store any private data, to be accessable only by a specific group of nodes in the system. These data are protected by symmetric key cryptography. The public region on the other hand is used to store any data regardless of its actual physical location. Storing the data in the public region protects the privacy of the nodes by storing the data in LADA without revealing the geographical location of the data owner. Original DHT implementations can be considered special cases of LADA that includes only one public region. Fig. 1 illustrates the structure of LADA.

B. Lookup process

The lookup process in LADA can be done in the private region, public region or in any give sub-region based on the physical location of the stored data. Based on that, there are three main lookup types: *local lookup*, *intra lookup* and *regional lookup*. The local lookup is done when the requesting node and the target node both belong to the same sub-region. This means that they lookup initiator and destination nodes are in the same geographical region. In this case the lookup process is initiated directly from the initiator node to the destination node. In the intra lookup, the sub-regions of both the initiator of the lookup process and the destination of the lookup process are different, but belong to the same region set. The lookup process first initiated to get the access data of a node in the destination sub-region from the set representative sub-region. In the second phase, the specific node in the destination sub-region will be reached. In the regional lookup, the process starts by targeting the set representative sub-region of the initiator node, then the set representative sub-region of the destination node, and finally the specific node in the destination sub-region.

C. Generation of Identifiers

A collision-resistance hash function is used in LADA during the process of generating the identifiers of the nodes. Each identifier in LADA consists of $2d$ bits, considering the use of hash functions that generates d bits message digest. A node in the proposed model might have either two or three distinct identifiers. The generation of identifiers for each newly joined process passes through two phases to join both private and public regions using two different generated identifiers, while the third phase of identifier generation for the local sub-region is optional. Fig. 2 illustrates these phases.

A new node in LADA uses the collision-resistant hash function to generate the d-bits digest based on the given input. A unique local information of the newly joined node is fed to the hash function as its input, and the output represents its local node id. Based on the output, it generates two identifiers that will be used to identify the node in the two special regions. The $2d$-bits identifiers in private region starts with d zeros, followed by the output of the used collision-resistant hash functions. The $2d$-bits identifier in public region starts with d ones followed by the output of the used collision-resistant hash functions. This means that the two generated identifiers of the same node in both private and public regions share the same last d bits. The newly joined node can join its local sub-region as well.

In order to join its local sub-region, the newly joined node in LADA has to generate the region id and then concatenate it with the local node id. As mentioned earlier, the regions in LADA are divided into multiple sets. Each set of sub-regions has a set representative sub-region and a number of other sub-regions. To generate the region id part of an identifier of a newly joined node, the node first get the location information of the set representative sub-region and fed that into the collision-resistant hash function. The most left $d/2$ bits of the output will be used, and the rest will be dropped. Then, the newly joined node takes the location information of the local sub-region it belongs to, and fed that into the collision-resistant hash function. The most right $d/2$ bits of the output will be used, and the rest will be dropped. The location information of the regions in the system can be constructed by any form such as a prefix of their geographical latitude/longitude coordinates, or human-readable names. The last $d/2$ bits in the region id part of the identifiers of nodes in the set representative sub-region is set to $d/2$-bits of zeros. Considering the Avalanche effect property [12] in the hash function algorithms, any given subset of generated digests in hash functions should be affected equally as any other given subset of the generated digests. Based on that, the generated region id part in the identifier of nodes that is based on the left $d/2$ bits of the digest of the location information of the set representative sub-region and the right $d/2$ bits of the digest of the location information of the local sub-region should not affect the randomness of the resulted final region id part in the generated identifiers.

After generating the region id part of the final identifier, the local node id that is resulted from hashing A unique local information of the newly joined node is concatenated to form the identifier of the node in a local sub-region. Fig. 3 shows the process of generating an identifier of a node in a local sub-region in LADA. A new node joins LADA through an *introducer node*. As any node in LADA can be an introducer node, we assume that the introducer node is already joined in LADA and is know by the newly joined node. The new node can join LADA by initiate a lookup request for its own identifiers in the respected sub-regions, namely private, public and local sub-region. By this process, the identifiers of the newly joined node and its address data will be registered in a number of nodes in LADA.

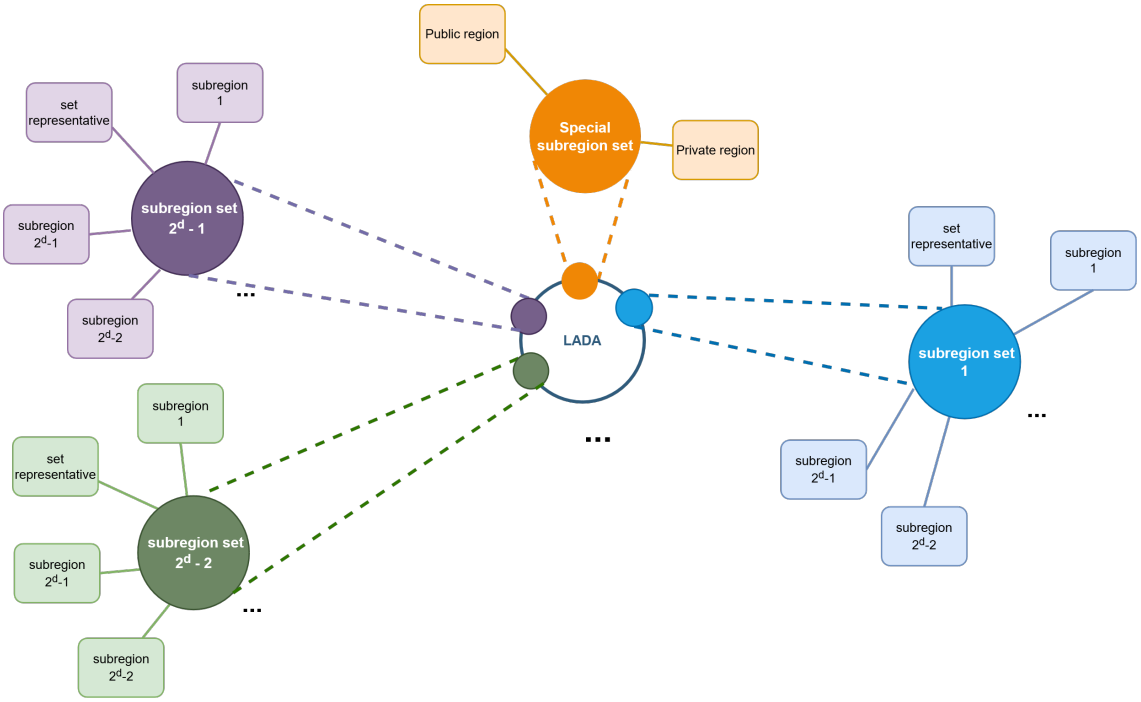

Fig. 1. Structure of LADA

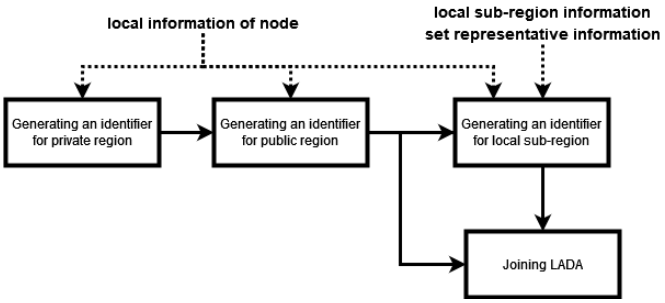

Fig. 2. Identifier generation phases

D. Parallelism and Replications

Two of the most important parameters in DHT based systems such as Kademlia [9] are the parallelism and the global replications that are denoted by α and k, respectively. A node in LADA has d lists, each of them with k-buckets [9]. A list l in the d lists includes the addresses and their corresponding identifiers to a maximum of k nodes in the same sub-region in LADA that shares the same $d+l$ bits of identifiers. k replicas of any given data in the LADA overlay is stored in k nodes that their identifier is close to the hash digest of the data, i.e. its key. The mapping M

$$M : \mathcal{D} \rightarrow \mathcal{N}$$

is used to find the subset of nodes in LADA as in 1 that are responsible to store a replica of the stored data in the overlay, in which $\mathcal{D}$ is the set of all possible keys in LADA and $\mathcal{N}_{rg}$ is the set of all nodes in the sub-region rg in LADA.

$$M(d) = \{n \in \mathcal{N}_{rg} : H(d) \approx id_n, \\ \nexists n' \in \mathcal{N}_{rg} \mid dst(id_{n'}, H(d)) < dst(id_n, H(d))\} \quad (1)$$

The dst function is used to compute the logical distance between any two nodes in LADA. A node in a sub-region in LADA also keeps a $d/2$ lists that include the access data to a maximum of k nodes in the different representative sets per list that share the same l bits of identifiers in a given list l. Additionally, the nodes in a set representative sub-region keep $d/2$ lists that includes the access data to some of the nodes in various sub-regions of the same region set. This ensures the ability of any node in LADA to access any representative set sub-region, and then through that, to access any node in a specific sub-region in that region set. There are number of researches such as in [13] [14] that focus on analysing parallelism, replications and other factors to improve the efficiency and latency in DHT based systems. These results can be applied in LADA to improve the its efficiency.

IV. Analysis and Discussions

One of the mains goals in LADA is to generate the identifiers in the system such that the close nodes in the underlay network are positioned in close locations in the overlay. This property prevent the high latency during the store and lookup processes among various edge/fog nodes int the system. To study the performance of the proposed model in the large scale, PeerSim [15] has been used. The Kademlia implementation [1] of DHT overlay in PeerSim has been slightly modified to fit

[1] http://peersim.sourceforge.net/

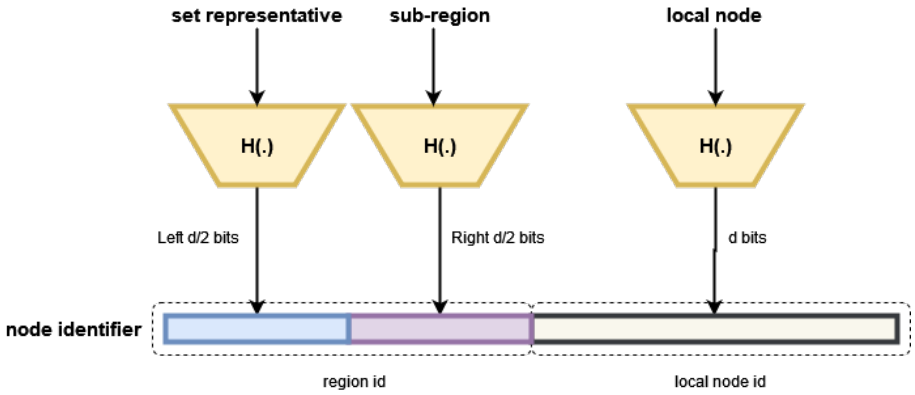

Fig. 3. Identifier generation in LADA

different parts of LADA. During studying the performance of the proposed model, the randomized latency parameters in Table I have been assumed that are based on the online available data [2] of possible communication delay.

TABLE I: Network parameters

type	parameter
local connection latency	2 ms
sub-regional latency (local region)	3 - 8 ms
intra-regional latency (region set)	10 - 30 ms
long distance latency	80 - 120 ms

The parallelism and replication factors have been set based on the used parameters in uTorrent [3], the popular implementation of Kademlia. They are set to four and eight, respectively. The simulated LADA overlay is constructed as 200 region sets. Each region set consists of one set representative sub-region and 199 other sub-regions. Each sub-region consists of 10 thousand up to 50 thousand participating nodes. The local lookup in a sub-region in LADA has been studied and the result is showed in Fig. 4 that states the lookup latency with different number of participating nodes in the sub-region.

The affect of churn on LADA has been studied to evaluate its availability and robustness. To study its affect, we introduced different ratio of churn in the system. The experiment has been done in a region with 10 thousand nodes and lasts for 120 thousand milliseconds, with a new lookup request every ten milliseconds. During the test and over different intervals from 100 milliseconds and up to 2000 milliseconds, an existing node has been randomly removed from LADA or a new node joined. Fig. 5 shows that the lookup delay is 11 milliseconds higher in a network with 100 millisecond churn rate (i.e. every 100 ms either a node leaves LADA or a new node joins LADA) comparing to a network that has no churn. The lookup delay drops below one millisecond in case that the churn rate is higher than 1600 milliseconds.

The affect of using a local cache in each system has been studied. In this experiment, various probability of cache hit from 5% up to 25% have been used. If the lookup is done for

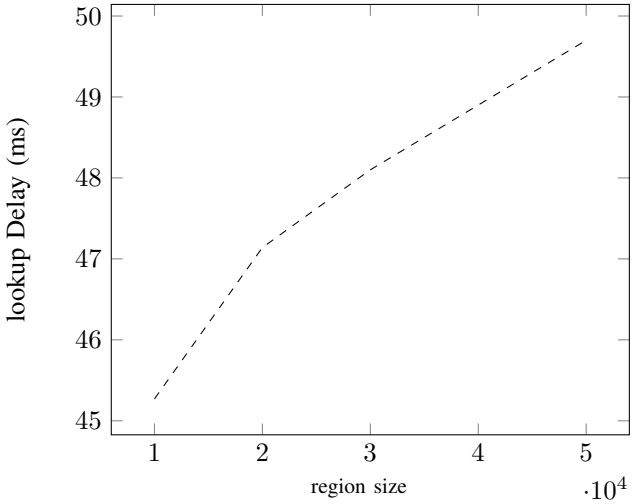

Fig. 4. Sub-region lookup delay

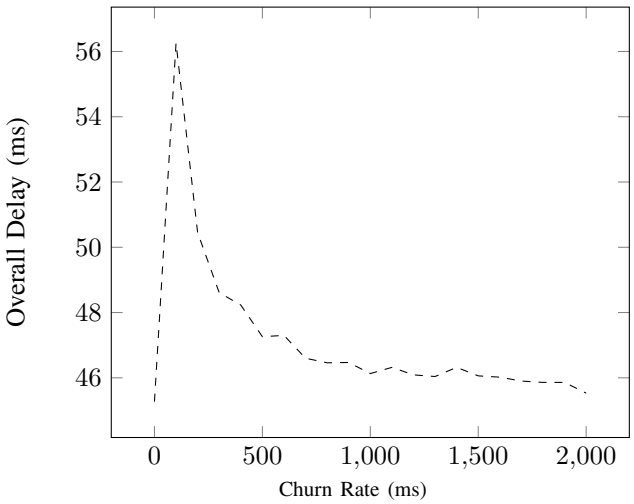

Fig. 5. Churn affect on LADA

[2]https://wondernetwork.com/pings
[3]https://www.utorrent.com/

a frequently lookup data, the probability of the cache hit is higher than other less frequently lookup data. The experiment has been done in a region with 10 thousand nodes, in which a new lookup request every one milliseconds is issued during the test of our model. Fig. 6 shows that the lookup delay in LADA has been improved linearly based on the used cache and type of the lookup data.

Unlike the models [2], [3] that have a set of known landmarks, LADA has no single or set of known and preset landmarks. This property removes any possible single point of attack and failure or points that might turn into performance bottleneck in the system. Additionally, new sub-regions and region sets can be created during the system life cycle and there is no need for the regions to be predefined prior to the system initialization. Compared to models that are based on ASNs of the participating nodes [1], distance to the known landmarks [2], or clustering [3], the participating nodes in LADA can join only the private and public regions and store the data in those two regions. Therefore, by joining only the public regions, their location data can be kept hidden. It is noteworthy to mention that this comes in the cost of lookup delay, since the lookup of the stored data in the public region is done based on the regional lookup process.

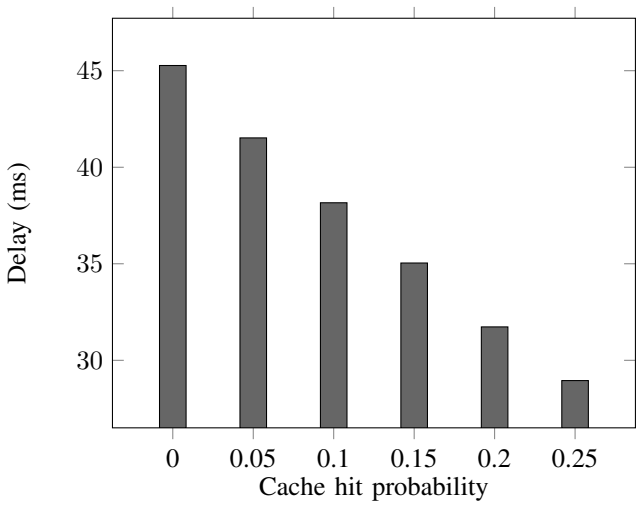

Fig. 6. Cache affect on LADA

V. Conclusions

This paper proposes a locality aware model for overlay generation and addressing of nodes (LADA) in a distributed scheme. The model focuses on edge/fog computing infrastructure where the computation is distributed among many nodes in the system. The model considers the physical locations of the participating nodes during the generation of identifiers that will be used to position the nodes in the created overlay. The model increases the efficiency of the lookup in a distributed scheme. It defines several local regions divided into a number of region sets. The nodes join the local sub-regions based on their physical locations, and this guarantees the data will be stored in a physically close node in the overlay. Additionally, the nodes join private and public regions. The private region stores the private data that can be later accessed based on a pre-shared key. The data can be stored in the public region regardless of the physical location of the node. Therefore their physical locations can be kept hidden and not revealed. The future directions can include improvements in creating different region sets, other identifier generation approaches, and positioning the newly joined nodes.

References

[1] W. Wu, Y. Chen, X. Zhang, X. Shi, L. Cong, B. Deng, and X. Li, "Ldht: Locality-aware distributed hash tables," in *2008 International Conference on Information Networking*, pp. 1–5, IEEE, 2008.

[2] Y. Hassanzadeh-Nazarabadi, A. Küpçü, and Ö. Özkasap, "Decentralized and locality aware replication method for dht-based p2p storage systems," *Future Generation Computer Systems*, vol. 84, pp. 32–46, 2018.

[3] T. Toda, Y. Tanigawa, and H. Tode, "Autonomous and distributed construction of locality aware skip graph," in *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 33–36, IEEE, 2017.

[4] M. B. Alshawki, P. Ligeti, and C. Reich, "Region-based distributed hash table for fog computing infrastructure," in *13th Joint Conference onMathematics and Informatics*, pp. 82–83, 2020.

[5] M. B. Alshawki, B. Crispo, and P. Ligeti, "A decentralized and scalable model for resource discovery in iot network," in *2019 International Conference on Wireless and Mobile Computing, Networking and Com-munications (WiMob)*, pp. 1–4, IEEE, 2019.

[6] M. B. Alshawki, Y. Yan, P. Ligeti, and C. Reich, "A decentralized resource discovery using attribute based encryption for internet of things," in *2020 4th Cyber Security in Networking Conference (CSNet)*, pp. 1–3, IEEE, 2020.

[7] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, pp. 329–350, Springer, 2001.

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001.

[9] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *International Workshop on Peer-to-Peer Systems*, pp. 53–65, Springer, 2002.

[10] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE Journal on selected areas in communications*, vol. 22, no. 1, pp. 41–53, 2004.

[11] W. D. Maurer and T. G. Lewis, "Hash table methods," *ACM Computing Surveys (CSUR)*, vol. 7, no. 1, pp. 5–19, 1975.

[12] Y. Yang, X. Zhang, J. Yu, P. Zhang, *et al.*, "Research on the hash function structures and its application," *Wireless Personal Communications*, vol. 94, no. 4, pp. 2969–2985, 2017.

[13] R. Jimenez, F. Osmani, and B. Knutsson, "Sub-second lookups on a large-scale kademlia-based overlay," in *2011 IEEE International Conference on Peer-to-Peer Computing*, pp. 82–91, IEEE.

[14] S. Roos, H. Salah, and T. Strufe, "On the routing of kademlia-type systems," Advances in Computer Communications and Networks, 2017.

[15] A. Montresor and M. Jelasity, "PeerSim: A scalable P2P simulator," in *Proc. of the 9th Int. Conference on Peer-to-Peer (P2P'09)*, (Seattle, WA), pp. 99–100, Sept. 2009.