

GTPP: General Truncated Pyramid Peer-to-Peer Architecture over Structured DHT Networks

Zhonghong Ou, Erkki Harjula, Timo Koskela and Mika Ylianttila

Abstract— Hierarchical Distributed Hash Table (DHT) architectures have been among the most interesting research topics since the birth of flat DHT architecture. However, most of the previous work has merely focused on the two-tier hierarchy. In this paper, we study and analyze General Truncated Pyramid Peer-to-Peer (GTPP) architecture, the generalized version of Partially Vertical Hierarchical Architecture (PV-HA). The idea is to study whether added tiers of hierarchy can provide added value in performance and functionality. Through mathematical analysis, we demonstrate performance results in comparison to flat architecture, which helps understanding the typical characteristics of hierarchical architectures. Firstly, GTPP has slightly higher expected lookup hop count, although it can be decreased with optimizing the sub-overlay setup. However, GTPP significantly decreases the expected lookup routing latency. Secondly, GTPP has clearer and more reasonable traffic distribution among all the peers from different tiers of sub-overlays, and can work with slightly lower maintenance traffic. Thirdly, our studies indicate that 2-3 tiers are most suitable in most cases for GTPP, considering all the parameters.

Keywords—general truncated pyramid peer-to-peer architecture, hierarchical architecture, partially vertical, fully vertical, horizontal, DHT.

I. INTRODUCTION

Peer-to-Peer (P2P) technologies have attracted extensive attention of the academia, industry, and the media in the past few years due to their decentralized, failure tolerant and scalable characteristics. A few important proposals have been put forward to implement the distributed lookup services utilizing Distributed Hash Table (DHT), including Chord [1], CAN [2], Kademia [3], Pastry [4], Tapestry [5], just to name a few. The key point of DHT-based P2P overlay lookup algorithms is that they firstly define a certain metric for the distance of the peer-peer and peer-key, and then store the key-value pair in the “closest” peer according to the distance metric defined. When looking up the value of the key, they utilize different mechanisms to approach the target peer which stores the key-value pair step by step, and finally, get the value of the key in a logarithmic number of hops. Originally, the aforementioned DHT lookup algorithms are based on flat architecture, which means the peers of the overlay have the same functionalities and load in routing and maintenance, etc.

On the other hand, the hierarchical architecture almost always accompanies the large-scale, complex systems. On the Internet, routers are grouped into various autonomous systems (AS). Different routing protocols are used for the intra-AS routing and inter-AS routing; while the former utilizes Routing Information Protocol (RIP) or Open Shortest Path First (OSPF) protocol, etc., the latter uses Border Gateway Protocol (BGP), etc. When describing the design of a global name system, Butler Lampson [6] stated the hierarchy as a fundamental method for accommodating growth and isolating faults. In the context of DHT design, hierarchy has

This work was supported in part by Tekes, Nokia, Ericsson and Nethawk in the project Decicom.

Zhonghong Ou was with Beijing University of Posts and Telecommunications. He is now with Department of Electrical and Information Engineering, University of Oulu, Finland, and School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing, China. (e-mail: tommy@ee.oulu.fi). Erkki Harjula, Timo Koskela and Mika Ylianttila are with Department of Electrical and Information Engineering, University of Oulu, FIN-90014, Finland. (e-mail: firstname.lastname@ee.oulu.fi).

also some notable advantages proved by the previous work: fault isolation and security, effective caching and bandwidth utilization [7], deduction in lookup hops and latency [8], adaptation to the underlying physical network [7] [9], providing administration control and autonomy [8] [10], more adaptable to mobile environments [11]. In our previous studies [12] [13], we also found the same advantage of hierarchical DHT, e.g. decreased lookup hop count.

In this paper, we continue our exploration in the hierarchical DHT design, or more precisely, the multiple-tier hierarchical DHT which is not limited to two-tier, as was usually done in the previous work. We first put forward a generalized multiple-tier hierarchical DHT, called General Truncated Pyramid Peer-to-Peer (GTPP) architecture. “General” here means the architecture is built on the basic functionalities of the hierarchical DHT, all of the tiers utilize the same basic DHT lookup algorithm (Chord as our instantiation), no optimization is adopted in the architecture, e.g. anycast [10], multicast [14], mesh-connection or single-connection [15] [16] in the lower tier. Thus, we can treat GTPP as a generalized version of multiple-tier hierarchical DHT architecture. With these assumptions, we then analyze the expected lookup hop count, expected lookup routing latency, traffic distribution of each single peer from a different tier of sub-overlay and the total traffic of GTPP compared to flat architecture. Finally, we give the experimental results based on the two-tier hierarchical community management system. The main conclusions in this paper are as follows:

- GTPP, as the generalized version of multiple-tier hierarchical DHT architecture, has slightly higher amount of lookup hop count than flat architecture. However, the lookup latency of GTPP is significantly decreased compared to flat architecture.
- GTPP has clearer and more reasonable traffic distribution among the peers from different tiers of sub-overlays, i.e. the higher tier a peer locates, the more traffic it shares.
- The maintenance traffic of GTPP is slightly less than flat architecture, although the total traffic of GTPP is slightly more than flat architecture in a highly active overlay network where there are a great deal of lookup operations. The total traffic of GTPP is increased as the number of tiers increases.
- As a whole, GTPP shows its advantages compared to flat architecture, our studies indicate that 2-3 tiers are most suitable in most cases for GTPP, considering all the parameters.

The remainder of this paper is organized as follows: Section II gives the related work and describes the taxonomy of hierarchical architecture. Section III introduces the GTPP architecture. Section IV analyzes the performance of GTPP. Section V presents the evaluation results. In Section VI, we elaborate the experimental results. Section VII gives some discussion about the content caching and iterative routing mechanism. Section VIII describes the future work and in Section IX we conclude the article.

II. RELATED WORK AND TAXONOMY

Hierarchical P2P architecture is resulted from the high popularity of file-sharing applications based on unstructured P2P networks, e.g. KaZaA [17], Gnutella [18]. Large population of KaZaA and Gnutella application results in large scale P2P networks around the world, which in turn causes the ever increasing search traffic across different administrative domains. Meanwhile, the search recall ratio, i.e. the ratio of the number of search results and the total number of available copies of the searched object, is

significantly decreased as the network scale becomes larger [16]. Therefore, KaZaA and Gnutella introduce the hierarchy in their architectures. KaZaA [17] designates the more powerful and available peers as *super-nodes*. The new nodes joining the KaZaA network set up connections to the closest *super-nodes* though the shortest Round Trip Time (RTT) metric. The *super-nodes* construct a top-level overlay network by proprietary design. In the later version of Gnutella [18], the similar mechanism is used which it names as *super-peers*. *Non-super-peer* looks up the service provided by the overlay through the gateway *super-peers*. The top level overlay utilizes the same unstructured algorithm as normal Gnutella. Another similar architecture is CAP [19], a two-tier unstructured P2P network focusing on scalability and stability.

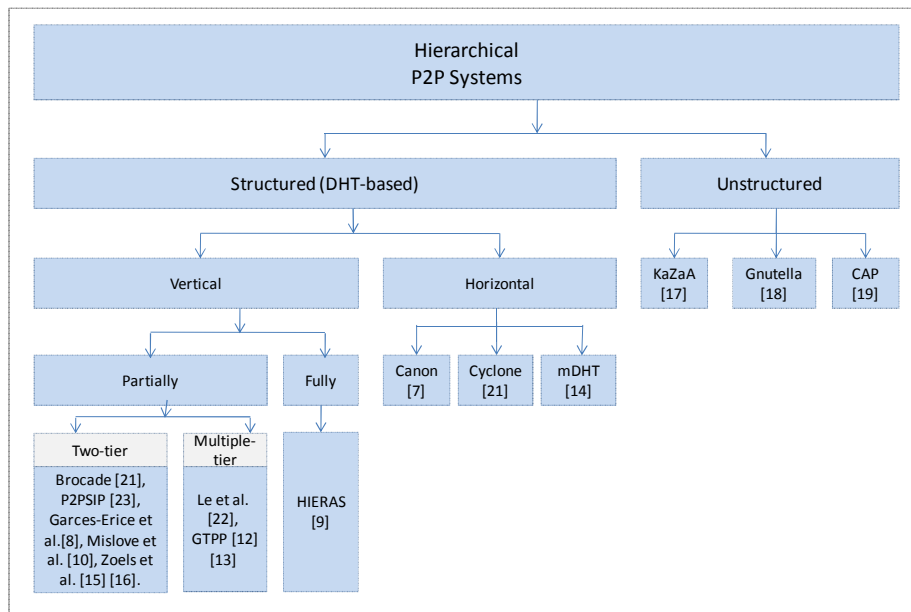


Fig. 1. Taxonomy of hierarchical P2P architecture.

In the structured P2P networks, or more precisely, DHT-based P2P networks, the same trend can be found. Sa´nchez-Artigas et al. [20] distinguish horizontal hierarchical and vertical hierarchical architecture and summarize the differences as follows: “In a horizontal overlay, all the leaf overlay networks are connected using a single DHT that contains the *conceptual hierarchy* and optimizes the routing in the whole network”, while in a vertical hierarchical architecture, “every layer or leaf in the hierarchical tree is a *self-contained* DHT overlay network”. In this paper, we further divide the vertical hierarchical architecture into fully-vertical hierarchical architecture (FV-HA) and partially-vertical hierarchical architecture (PV-HA). In FV-HA, all the peers participate in the top-tier sub-overlay as well as its local sub-overlay; therefore, no gateway peers are needed; while in PV-HA, just partial peers participate in the top-tier overlay, the other peers do not participate in the top-tier because of firewall security, Network Address Translation (NAT), administrative reason, etc. Therefore, dedicated gateway peers are needed in PV-HA to route the query on behalf of the peers from lower-tiers of sub-overlays. An illustrated category of hierarchical P2P architecture is shown in Fig. 1.

In the horizontal hierarchical architecture (H-HA), Canon [7], mDHT [14], and Cyclone [20] are typical examples. Canon [7] and Cyclone [20] adopt special Identifier (ID) design mechanism to make the total number of links per node maintaining remain the same as the flat DHT design. In mDHT [14], the selection of neighboring links is more flexible, it treats a group of host computers

in a subset participating in the DHT overlay as a single *node*, and multiple links can be made between different *nodes*. In H-HA, no gateway peers are needed. The advantages of H-HA include: fault isolation, load balancing, topology awareness, and hierarchical storage of content and access control [7]. To our best knowledge, the most notable advantage of H-HA in comparison to PV-HA and FV-HA is the optimized number of connections each node maintaining. The typical disadvantages of H-HA include: it supposes all the nodes of the overlay network can participate in the top-tier sub-overlay, in cases where many nodes cannot satisfy this requirement because of firewall or NAT problem, etc. the H-HA does not work; furthermore, H-HA does not have a clear distinction of the load different nodes are supposed to take, which makes it non-suitable for highly heterogeneous environments.

In FV-HA, all the peers participate in all the tiers of sub-overlays. HIERAS [9] is a representative example. By grouping topologically adjacent peers into lower level rings, HIERAS provides the overlay network with routing locality. Meanwhile, HIERAS has to create and maintain one finger table for each tier of the architecture. The advantages and disadvantages of FV-HA are obvious. The former includes routing locality, topology awareness, etc. The latter includes the facts that it is non-suitable for environments where lots of peers locate behind firewall or NAT, and it increases overlay maintenance cost, etc.

In PV-HA, the key point is the existence of dedicated gateway peers which participate in more than one tiers of the hierarchy and serve as proxies of lower-tier peers. Besides most of the advantages of H-HA, i.e. fault isolation, administrative autonomy, efficient caching and replica, PV-HA has its own benefits: it has a clear distinction of the load different peers are supposed to take which makes it suitable for highly heterogeneous environments, e.g. wireless networks; it works in environments where a number of peers locate behind NAT or firewall, etc. A lot of proposals have been put forward in this category, including Brocade [21], Le et al. [22], P2PSIP [23], Garces-Erice et al.[8], Mislove et al. [10], Zoels et al. [15] [16]. Our previous studies [12] [13] also belong to this category. However, most proposals are focused on two-tier hierarchy, just [22] and our previous studies [12] [13] explore the multiple-tier, or more precisely, more than two-tier hierarchy. Le et al. [22] presents an architecture where all the peers at the lower-tier form only one loop which makes it lose most of the advantages of PV-HA; therefore, we do not study the special architecture in this paper. In [12], we only analyzed the lookup hop count of the GTPP architecture and did not consider the forwarding hop count (the hop count between different tiers of sub-overlays); while in [13], we studied a special three-tier hierarchical architecture without any emphasis on the general architecture (GTPP). In this paper, we extend the analysis from our previous work by 1) considering more performance metrics, including expected lookup hop count, expected lookup latency, and maintenance traffic of multiple-tier PV-HA, and 2) substantiating the claims regarding the lookup hop count and maintenance traffic by simulations. The idea is to study whether more than two tiers of hierarchy can provide added value in performance and functionality. To our best knowledge, this paper is the first attempt to provide mathematical analysis to these metrics in a multiple-tier PV-HA.

III. GENERAL TRUNCATED PYRAMID P2P ARCHITECTURE

In this section, the GTPP architecture and the associated service lookup procedure are introduced. We do not discuss the

mechanisms to classify N heterogeneous nodes into k sub-overlays. There is a great deal of research work on this already; one example is [24]. There are also a great number of DHT algorithms that can be used to constitute the overlay. In this paper, we utilize Chord [1] as one instantiation to form the overlay network. However, our results can be easily extended to other DHT algorithms, e.g. Kademlia [3], Pastry [4].

A. GTPP Architecture

GTPP is a multiple-tier PV-HA. To make it clear, Fig. 2 just illustrates a three-tier hierarchical architecture. In each tier of sub-overlay, all the peers are grouped into several disjointed sub-sub-overlays (denoted as SSOs, also the notation of $S_{x,y}$ in Fig. 2). There is only one SSO at the highest tier. Because of the autonomy characteristic of the hierarchical architecture, different SSO can utilize different DHT algorithms, or the topologies of different SSO can be different, e.g. fully-mesh, single-connection (each normal peer just sets up one connection to its super-peer), or DHT-based. Without loss of generality, this paper assumes all the SSOs adopt the same DHT topology and the same DHT algorithm, i.e. Chord as our instantiation. Peers from the same SSO can look up the keys stored on each other according to the associated DHT algorithm. Peers from different SSO of the same tier cannot look up the keys stored on each other directly; instead, they utilize the nodes from the upper tier of sub-overlay as proxies. Each SSO, except the topmost one, has one super-peer, which we name as *gateway super-peer* to differentiate it from the other normal peers. Besides participating in the local SSO, the *gateway super-peer* also participates in the higher tier of SSO. From the viewpoint of robustness, practical overlay usually has more than one *gateway super-peers* in each SSO; however, this does not affect our analysis much. Therefore, for the simplicity of our analysis, we just assume one *gateway super-peer* in each SSO in the rest of this paper. Accordingly, the peers from the first tier of sub-overlay participate in one SSO, while the peers from the other tiers of sub-overlays participate in two SSOs at the same time.

As one example in Fig. 2, the first tier of sub-overlay consists of six SSOs, denoted as $S1_1$, $S1_2$, etc.; the second tier of sub-overlay consists of three SSOs, denoted as $S2_1$, $S2_2$ and $S2_3$, while the third tier of sub-overlay consists of only one SSO, denoted as $S3_1$. $N1_{51}$ and $N1_{42}$ participate in only one SSO, i.e. $S1_6$. $N2_{48}$ participates in two SSOs at the same time, i.e. $S1_6$ and $S2_3$. $N3_{39}$ participates in two SSOs as well, i.e. $S2_3$ and $S3_1$. From the top down, from the vertical perspective, all the peers from different sub-overlays constitute multiple trees rooted by the peers at the topmost sub-overlay, which makes it look like a pyramid being truncated. And this inspires the name of Truncated Pyramid (TP) in our GTPP architecture.

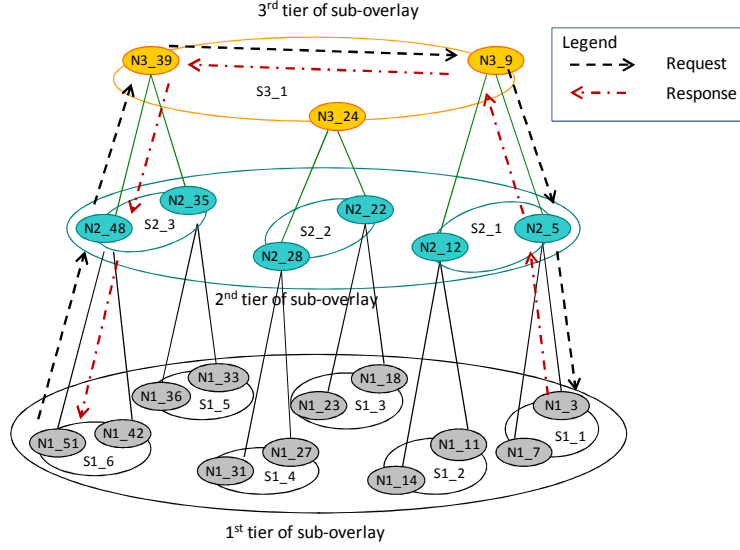


Fig. 2. General Truncated Pyramid P2P (GTPP) architecture.

B. Service lookup procedure

When a peer wants to look up a service or resource provided within the GTPP architecture, it follows the following steps:

1. The initiating peer sends the lookup request to its *gateway super-peer*, and then the *gateway super-peer* forwards the request to its own *gateway super-peer* at the higher tier, and so on. This operation is executed recursively until the request reaches the top-most tier of SSO.
2. The super-peer locating at the top-most tier of SSO resolves the request and looks up the service or resource in its local SSO utilizing the associated DHT algorithm. If no successful response is got from its local SSO, the request is forwarded to the immediately close lower tier of SSO which is closest to the service or resource and the lookup procedure jumps to Step 3. If one successful response is got from the top-most tier of SSO, the lookup procedure is halted immediately and the response is returned back to the initiating peer.
3. The same procedure is done recursively as in Step 2. If no successful response is received, the lookup procedure continues until the request reaches the lowest tier of SSO.
4. At the lowest tier of SSO, a response, either successful or failed, is returned to the initiating peer. The response follows the reverse path of the request (the so-called recursive routing) to the initiating peer.

In Step 1, we assume that all the lookup requests are resolved at the top-most tier of SSO firstly, which makes sense as the peers at the top-most tier of SSO have the most knowledge of the whole overlay network. In Section VII.A, we will see the advantage of this “fountain-like” lookup mechanism to the efficiency of content caching in PV-HA.

We also assume recursive routing in the aforementioned lookup procedure. We will analyze the iterative routing mechanism in Section VII.B. In Step 4, if the response is returned back to the initiating peer directly instead of following the reverse path of the request, the routing mechanism is so-called *semi-recursive routing*. The advantages and disadvantages of *semi-recursive routing* are still under debate. The most explicit disadvantages of semi-recursive routing include asymmetric routing and weak capability of

traversing NAT. We do not consider this routing mechanism in this paper.

One example scenario is illustrated in Fig.2, where N1_51 in S1_6 wants to find the key (ID=2) stored at N1_3 in S1_1. The requests and responses are shown as the black dashed lines and the red dash-dotted lines respectively. The lookup procedure is as follows:

1. N1_51 sends the lookup request to its *gateway super-peer*, i.e. N2_48;
2. N2_48 forwards the request to its own *gateway super-peer*, i.e. N3_39;
3. N3_39 resolves the request in its local SSO, i.e. S3_1, by utilizing the Chord algorithm, and then the request is routed to N3_9 which is the closest peer to the requested key in S3_1;
4. N3_9 looks up the requested key in its lower tier of SSO, i.e. S2_1, by utilizing Chord, and then the request is routed to N2_5 which is closest to the target key in S2_1;
5. Finally, N2_5 resolves the request in S1_1 and finds the key stored at N1_3. The successful response with the key-value pair is returned back to N1_51 along the reverse path of the request.

In the lookup procedure mentioned above, it is noteworthy that Step 1 and Step 2 only cost one lookup hop, while Step 3-Step5 each costs multiple lookup hops according to the adopted DHT algorithm. Except the forwarding hops to the *gateway super-peers*, we can see that through one SSO (S3_1), the keys stored at the top-most tier of SSO (S3_1) can be found; through two SSOs, the keys stored at the 2nd tier of sub-overlay can be found; while through three SSOs, the keys stored at the 1st tier of sub-overlay can be found. Therefore, in a k -tier GTPP architecture, at most k SSOs are needed to cover the whole overlay architecture.

IV. PERFORMANCE ANALYSIS

Le et al. [22] adopted exponential distribution (ED) to divide all the peers of the whole P2P overlay into multiple tiers of sub-overlays. The basic idea is that, according to the maximum information entropy principle [25], the exponential distribution with a mean $1/\lambda$ is the maximum entropy distribution among all the continuous distributions supported in $[0, \infty)$ that have a mean of $1/\lambda$. Physically, this also makes sense. As in real world, the majority of peers have relatively weak capability, while the much powerful peers are in the minority. By grouping the relatively weak peers into lower tiers, and the powerful peers into the upper tiers, the limited physical capability of the whole overlay network can be optimized. Therefore, Le et al. [22] utilized the ED classifier to group the peers of the overlay network into different tiers.

In our previous work [12], we found the similarity between ED and Geometric Distribution (GD), i.e. the GD has almost the same curve shape as ED except GD is in discrete domain and the ED is in continuous domain, thus, GD can roughly be treated as the discrete counterpart of ED. Therefore, we utilized GD to distribute all the peers of the overlay into different tiers approximately. In this paper, we continue to utilize the GD to group all the peers into different tiers. It means from the bottom up, the number of peers at each tier of sub-overlay follows GD. We argue here that the adopted distribution is not so important, as the main goal of this paper focuses on the qualitative analysis instead of quantitative analysis. Different distribution may have an influence on the quantitative result; however, its influence on qualitative analysis is limited. The notations used are shown in Table 1.

TABLE I SYMBOLS AND THEIR MEANING

Symbol	Description	Symbol	Description
N	The total number of peers in the whole overlay, natural number.	$D(n,k)$	The average lookup latency for the whole GTPP overlay network.
p	The probability of one peer being located at the first tier of sub-overlay, $0 < p < 1$.	$Z^{(i)}$	The matrix of the lookup traffic distribution for a single lookup which is initiated by the i^{th} tier of sub-overlay, $i=1,2,\dots,k$.
k	The total number of tiers of GTPP, $k=1,2,3,\dots$, natural number.	$z_j^{(i)}$	The j^{th} column vector of $Z^{(i)}$, $j=1,2,3,\dots,k$. The lookup traffic distribution among each tier of sub-overlay for the lookup halted through j SSOs and initiated by the i^{th} tier of sub-overlay, $i=1,2,\dots,k$.
N_i	The number of peers at the i^{th} tier of sub-overlay, $i=1,2,3,\dots,k$.	\mathbb{F}	Interim matrix, with the size of $k*k$.
ζ_i	The probability of one key being stored at the i^{th} tier of sub-overlay, $i=1,2,3,\dots,k$.	f_i	The i^{th} column vector of \mathbb{F} .
P_i	The probability of one peer being located at the i^{th} tier of sub-overlay, $i=1,2,\dots,k$.	α_i	The lookup traffic distribution of the peer at the i^{th} tier of sub-overlay.
$H_{i,j}$	The hop count of one successful lookup through j SSOs, $j=1,2,\dots,k$, for the request initiated at the i^{th} tier of sub-overlay, $i=1,2,3,\dots,k$.	β_i	The maintenance traffic distribution of the peer at the i^{th} tier of sub-overlay.
$L_{i,j}$	The latency of one successful lookup through j SSOs, $j=1,2,\dots,k$, for the request initiated at the i^{th} tier of sub-overlay, $i=1,2,3,\dots,k$.	γ_i	The republish traffic distribution of the peer at the i^{th} tier of sub-overlay.
ω_i	The average round trip time (RTT) of one lookup hop at the i^{th} tier of sub-overlay, $i=1,2,3,\dots,k$.	ϑ	The average number of items shared by each peer.
Γ_i	The expected hop count of one successful lookup for peers at the i^{th} tier of sub-overlay, $i=1,2,\dots,k$.	m_i	The total traffic distribution of the peer at the i^{th} tier of sub-overlay.
Ω_i	The expected latency of one successful lookup for peers at the i^{th} tier of sub-overlay, $i=1,2,\dots,k$.	T	The time interval.
$E(n,k)$	The average lookup hop count for the whole GTPP overlay network.		

To make the analysis clearer, we make the following assumptions:

1. The overlay is in stable state, which means all the N peers are located at the proper sub-overlay tiers according to their physical capability. (The effect of churn to the performance of PV-HA is our future work)
2. The IDs of peers and keys are generated and distributed uniformly, which means all the peers from different tiers have the same probability to store a key. (This assumption can be easily satisfied utilizing the existing load balancing mechanisms)
3. Each peer from a different tier of sub-overlay has equal probability to look up a random key.

A. Lookup hop count

According to the assumptions and notations, we can see that the number of peers at each tier of sub-overlay has the following equation:

$$\left. \begin{array}{l} N_1 = p \cdot N \\ N_2 = p \cdot (1-p) \cdot N \\ \dots \\ N_{k-1} = p \cdot (1-p)^{(k-2)} \cdot N \\ N_k = (1-p)^{(k-1)} \cdot N \end{array} \right\}. \quad (1)$$

Here the topmost tier of sub-overlay has the difference between the total number of peers and the accumulated number of peers

from the other tiers of sub-overlays. That is to say, $N_k = N - \sum_{i=1}^{k-1} N_i = (1-p)^{(k-1)} \cdot N$. Therefore,

$$\frac{N_{k-1}}{N_k} = \frac{p}{1-p} = \frac{1}{1-p} - 1. \quad (2)$$

Eq. (2) stands for the ratio of the number of peers at the $(k-1)^{\text{th}}$ tier of sub-overlay and the number of peers at the k^{th} tier of sub-overlay. From Eq. (2), we can see that as long as $p > 0.5$, then $N_{k-1} > N_k$. It means the $(k-1)^{\text{th}}$ tier of sub-overlay has more peers than the k^{th} tier of sub-overlay. It is intuitively reasonable in the sense that the peers at the lower tiers of sub-overlays have less physical capability and account for the majority of the whole overlay network. It also conforms to the maximum information entropy principle aforementioned.

From Eq. (1), we can also see that, if $i \neq k$, then $N_{i-1} / N_i = 1 / (1-p)$. As $0 < p < 1$, the value of $1 / (1-p)$ is always larger than 1. It means that the number of peers at the $(i-1)^{\text{th}}$ tier of sub-overlay is $1 / (1-p)$ times of the number of peers at the i^{th} tier of sub-overlay. It also means that the number of peers in each SSO (except the topmost one) is $1 / (1-p)$ (see Fig. 2). More precisely, the number of peers in each SSO (except the topmost one) is $1 + 1 / (1-p)$, as the *gateway super-peer* participates in two SSOs at the same time. To make it clear, we denote $1 / (1-p)$ as n , then Eq. (1) can be expressed as follows:

$$\left. \begin{array}{l} N_1 = (1 - \frac{1}{n}) \cdot N \\ N_2 = (1 - \frac{1}{n}) \cdot \frac{1}{n} \cdot N \\ \dots \\ N_{k-1} = (1 - \frac{1}{n}) \cdot (\frac{1}{n})^{(k-2)} \cdot N \\ N_k = (\frac{1}{n})^{(k-1)} \cdot N \end{array} \right\}. \quad (3)$$

We can see that the SSO from the topmost tier of sub-overlay has N_k peers, while the other SSOs from the lower tiers of sub-overlays have the same $(n+1)$ peers.

As an example in Fig. 2, the SSO from the topmost tier of sub-overlay, i.e. S3_1, has three peers (N3_9, N3_24, and N3_39); while the other SSOs from the first and second tiers of sub-overlays have the same three peers. S1_6 includes peers N1_42, N1_51, and N2_48. S2_3 includes peers N2_35, N2_48, and N3_39. *Gateway super-peers*, e.g. N2_48 and N3_39, participate in two SSOs

at the same time. Actually, as we assume one *gateway super-peer* in each SSO, each peer from the upper tiers of sub-overlays is a *gateway super-peer*, therefore, participates in two SSOs at the same time. In this way, by adjusting the value of n , we can get different overlay topologies with variable peers at each tier of sub-overlay.

From Eq. (3), we can see that the probability of one peer being located at the i^{th} tier of sub-overlay, $i=1, 2 \dots k$, can be calculated by the following equations:

$$\left\{ \begin{array}{l} P_1 = \frac{N_1}{N} = 1 - \frac{1}{n} \\ P_2 = \frac{N_2}{N} = (1 - \frac{1}{n}) \cdot \frac{1}{n} \\ \dots \\ P_{k-1} = \frac{N_{k-1}}{N} = (1 - \frac{1}{n}) \cdot (\frac{1}{n})^{(k-2)} \\ P_k = \frac{N_k}{N} = (\frac{1}{n})^{(k-1)} \end{array} \right\} . \quad (4)$$

With the Assumption 2, the IDs of keys and peers are distributed uniformly. Therefore, the probability of one key being stored at the i^{th} tier of sub-overlay, $i=1, 2 \dots k$, equals to the probability of one peer being located at the i^{th} tier of sub-overlay, i.e. $\zeta_i = P_i$, therefore, the following results exist:

$$\left\{ \begin{array}{l} \zeta_1 = \frac{N_1}{N} = 1 - \frac{1}{n} \\ \zeta_2 = \frac{N_2}{N} = (1 - \frac{1}{n}) \cdot \frac{1}{n} \\ \dots \\ \zeta_{k-1} = \frac{N_{k-1}}{N} = (1 - \frac{1}{n}) \cdot (\frac{1}{n})^{(k-2)} \\ \zeta_k = \frac{N_k}{N} = (\frac{1}{n})^{(k-1)} \end{array} \right\} . \quad (5)$$

Now, let us first consider the situation where the peer from the topmost tier of sub-overlay wants to look up the keys stored by the other peers in GTPP. In this situation, the following results exist:

$$\left\{ \begin{array}{l} H_{k,1} = \frac{1}{2} \cdot \log(N_k) = \frac{1}{2} \cdot \log(N) - \frac{1}{2} \cdot (k-1) \cdot \log(n) \\ H_{k,2} = \frac{1}{2} \cdot \log(N_k) + \frac{1}{2} \cdot \log(n+1) \\ \dots \\ H_{k,k-1} = \frac{1}{2} \cdot \log(N_k) + \frac{1}{2} \cdot (k-2) \cdot \log(n+1) \\ H_{k,k} = \frac{1}{2} \cdot \log(N_k) + \frac{1}{2} \cdot (k-1) \cdot \log(n+1) \end{array} \right\} . \quad (6)$$

In Eq. (6) above, we assume the overlay is fully populated. According to [1], on average, $\frac{1}{2} \log N$ hops are needed to resolve the responsible peer in a fully populated DHT overlay with N peers. Take $H_{k,2}$ as an example, it means the lookup request is initiated from the k^{th} tier of sub-overlay, and needs two SSOs to complete the lookup procedure. The first part of $\frac{1}{2} \cdot \log(N_k)$ means the lookup

hop count in the topmost, i.e. k^{th} , tier of sub-overlay (the first SSO has N_k nodes), while the second part of $\frac{1}{2} \cdot \log(n+1)$ means the lookup hop count in the $(k-1)^{\text{th}}$ tier of sub-overlay (the second SSO has $n+1$ nodes). Thus, the expected hop count for one successful lookup initiated at the topmost tier of sub-overlay has the following result:

$$\left\{ \begin{array}{l} \Gamma_k = \sum_{j=1}^k H_{k,j} \cdot \zeta_{k+1-j} \\ = \frac{1}{2} \log(N) - \frac{1}{2} (k-1) \cdot \log(n) + \frac{1}{2} \left(1 - \frac{1}{n}\right) \cdot \log(n+1) \cdot \sum_{j=1}^{k-1} j \cdot \left(\frac{1}{n}\right)^{(k-1-j)} \end{array} \right\} . \quad (7)$$

From the lookup procedure mentioned in Section III.B, we know that the lookup procedure from the lower tiers of sub-overlays just adds a small number of additional forwarding hops to reach the topmost tier of sub-overlay. Thereafter, the lookup procedure is the same as the lookup initiated from the topmost tier of sub-overlay. Thus, compared to the lookup initiated from the k^{th} tier of sub-overlay, the lookup initiated from the $(k-1)^{\text{th}}$ tier of sub-overlay adds one more lookup hop, while the lookup initiated from the $(k-2)^{\text{th}}$ tier of sub-overlay adds two more lookup hops, and so on. Based on Eq. (7), we can get the following equation:

$$\left\{ \begin{array}{l} \Gamma_1 = \Gamma_k + k - 1 \\ \Gamma_2 = \Gamma_k + k - 2 \\ \dots \\ \Gamma_{k-1} = \Gamma_k + 1 \\ \Gamma_k = \frac{1}{2} \log(N) - \frac{1}{2} (k-1) \cdot \log(n) + \frac{1}{2} \left(1 - \frac{1}{n}\right) \cdot \log(n+1) \cdot \sum_{j=1}^{k-1} j \cdot \left(\frac{1}{n}\right)^{(k-1-j)} \end{array} \right\} . \quad (8)$$

After we get the expected hop count for the lookup initiated from each tier of sub-overlay and the probability of one peer located at each tier of sub-overlay, the expected hop count for one successful lookup in the whole GTPP overlay can be calculated as follows:

$$E(n, k) = \sum_{i=1}^k P_i \cdot \Gamma_i . \quad (9)$$

B. Lookup latency

Following the similar analysis procedure, based on Eq. (6), we can get the values of $L_{k,j}$ as follows by multiplying the expected lookup hop count by the associated lookup latency for each hop:

$$\left\{ \begin{array}{l} L_{k,1} = \frac{1}{2} \cdot \log(N_k) \cdot \omega_k = \left[\frac{1}{2} \cdot \log(N) - \frac{1}{2} \cdot (k-1) \cdot \log(n) \right] \cdot \omega_k \\ L_{k,2} = \frac{1}{2} \cdot \log(N_k) \cdot \omega_k + \frac{1}{2} \cdot \log(n+1) \cdot \omega_{k-1} \\ \dots \\ L_{k,k-1} = \frac{1}{2} \cdot \log(N_k) \cdot \omega_k + \frac{1}{2} \cdot \log(n+1) \cdot \sum_{j=2}^{k-1} \omega_j \\ L_{k,k} = \frac{1}{2} \cdot \log(N_k) \cdot \omega_k + \frac{1}{2} \cdot \log(n+1) \cdot \sum_{j=1}^{k-1} \omega_j \end{array} \right\} . \quad (10)$$

The expected latency of one successful lookup for the peers at the top-most tier of sub-overlay is shown as: $\Omega_k = \sum_{j=1}^k L_{k,j} \cdot \zeta_{k+1-j}$.

The other values of Ω_i can be calculated as follows:

$$\left\{ \begin{array}{l} \Omega_1 = \Omega_k + \sum_{j=1}^{k-1} \omega_j \\ \Omega_2 = \Omega_k + \sum_{j=2}^{k-1} \omega_j \\ \dots \\ \Omega_{k-1} = \Omega_k + \omega_{k-1} \\ \Omega_k = \sum_{j=1}^k L_{k,j} \cdot \zeta_{k+1-j} \end{array} \right\} . \quad (11)$$

Take Ω_{k-1} as an example, it means the lookup request is initiated from the $(k-1)^{\text{th}}$ tier of sub-overlay. Compared to the lookup request which is initiated from the k^{th} tier of sub-overlay, one more forwarding hop taking place at the $(k-1)^{\text{th}}$ tier of sub-overlay is needed, and thus, the associated RTT latency is ω_{k-1} .

After we get the expected latency for the lookup initiated from each tier of sub-overlay, and the probability of one peer located at each tier, the expected latency for one successful lookup in the whole GTPP architecture has the following equation:

$$D(n, k) = \sum_{i=1}^k P_i \cdot \Omega_i . \quad (12)$$

C. Traffic distribution

In this section, we analyze the traffic distribution of each single peer from different tier of sub-overlay. We begin our analysis with the lookup traffic distribution. Then, the maintenance traffic is analyzed. Finally, the traffic generated by republishing the resources is analyzed.

Lookup traffic distribution

We first analyze the lookup request initiated from the topmost (k^{th}) tier of sub-overlay. From the analysis procedure in Section IV.A, we know that, if the lookup procedure is halted at the topmost tier of sub-overlay, one SSO is enough, i.e. $1/2 \cdot \log(N_k)$ lookup hops; if the lookup procedure is halted at the $(k-1)^{\text{th}}$ tier of sub-overlay, two SSOs are needed, one from the k^{th} tier of sub-overlay, and the other from the $(k-1)^{\text{th}}$ tier of sub-overlay, $1/2 \cdot \log(N_k)$ and $1/2 \cdot \log(n+1)$ lookup hops, respectively. For the number of messages sent and received, as one hop needs two messages sent and two messages received, the number of messages sent and received is twice of the lookup hop count, respectively. Meanwhile, it is noteworthy that the messages sent and received by the *gateway super-peer* should be grouped into the upper tier of sub-overlay. Therefore, the following equation exists (just shows the number of sent messages):

$$\left\{ \begin{array}{l} \mathcal{Z}_1^{(k)} = [0, 0, \dots, \log(N_k)]^T \\ \mathcal{Z}_2^{(k)} = [0, 0, \dots, \log(n+1) - 1, \log(N_k) + 1]^T \\ \dots \\ \mathcal{Z}_{k-1}^{(k)} = [0, \log(n+1) - 1, \dots, \log(n+1), \log(N_k) + 1]^T \\ \mathcal{Z}_k^{(k)} = [\log(n+1) - 1, \log(n+1), \dots, \log(n+1), \log(N_k) + 1]^T \end{array} \right\} . \quad (13)$$

The probability of one successful lookup through i SSOs, $i=1, 2, 3 \dots k$, is shown in Eq. (5). Thus, we can get the value of f_k as follows:

$$\mathbf{f}_k = \begin{bmatrix} 0 & 0 & \dots & 0 & \log(n+1)-1 \\ 0 & 0 & \dots & \log(n+1)-1 & \log(n+1) \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \log(n+1)-1 & \dots & \log(n+1) & \log(n+1) \\ \log(N_k) & \log(N_k)+1 & \dots & \log(N_k)+1 & \log(N_k)+1 \end{bmatrix} \cdot \begin{bmatrix} \left(\frac{1}{n}\right)^{(k-1)} \\ \left(1-\frac{1}{n}\right) \cdot \left(\frac{1}{n}\right)^{(k-2)} \\ \dots \\ \left(1-\frac{1}{n}\right) \cdot \frac{1}{n} \\ 1-\frac{1}{n} \end{bmatrix}. \quad (14)$$

If the lookup request is initiated from the $(k-1)^{\text{th}}$ tier of sub-overlay, most of the lookup procedure is the same as the request initiated from the k^{th} tier, just one more lookup hop (forwarding the request to the top-most tier of sub-overlay) is needed at the beginning of the lookup procedure. Therefore, the value of \mathbf{f}_{k-1} is as follows:

$$\mathbf{f}_{k-1} = \begin{bmatrix} 0 & 0 & \dots & 0 & \log(n+1)-1 \\ 0 & 0 & \dots & \log(n+1)-1 & \log(n+1)+1 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \log(n+1) & \dots & \log(n+1)+1 & \log(n+1)+1 \\ \log(N_k)+1 & \log(N_k)+2 & \dots & \log(N_k)+2 & \log(N_k)+2 \end{bmatrix} \cdot \begin{bmatrix} \left(\frac{1}{n}\right)^{(k-1)} \\ \left(1-\frac{1}{n}\right) \cdot \left(\frac{1}{n}\right)^{(k-2)} \\ \dots \\ \left(1-\frac{1}{n}\right) \cdot \frac{1}{n} \\ 1-\frac{1}{n} \end{bmatrix}. \quad (15)$$

With the same processes, we can get the other column vectors of \mathbb{F} . The probability of one peer being located at the i^{th} tier of sub-overlay is shown in Eq. (4). Thus, we can get the lookup traffic distribution of one single peer from different tier of sub-overlay as follows:

$$[\alpha_1, \alpha_2, \dots, \alpha_k]^T = [f_1, f_2, \dots, f_k][P_1, P_2, \dots, P_k]^T, \quad (16)$$

$$\alpha = \sum_{i=1}^k \alpha_i, \quad (17)$$

wherein α is the expected number of messages sent (or received) for one lookup of the whole GTPP architecture. As one lookup hop needs two messages sent, we can also get the relationship between α and $E(n, k)$ as:

$$\alpha = 2 \cdot E(n, k). \quad (18)$$

Maintenance traffic distribution

In addition to the traffic generated by the service lookup, each peer at each SSO also has to keep some state information to make the overlay network function correctly. In Chord [1], each peer has to run the STABILIZE and FIXFINGER protocols periodically to detect the failed peers and ensure the consistency of the finger tables. Each STABILIZE procedure includes three messages: the initiator sends the REQUESTPREDECESSOR message to its successor, the successor responds with RESPONSEPREDECESSOR message which includes its new (probably) predecessor, and finally, the initiator sends the NOTIFY message to this new (probably) predecessor of the

successor. As the *gateway super-peers* participate in two SSOs at the same time, so they have to run the STABILIZE algorithm in each of the two participating SSOs periodically. Thus, the number of messages generated by the STABILIZE procedure for each peer from the i^{th} tier of sub-overlay is as follows:

$$\left\{ \begin{array}{l} \beta_{1,STAB} = 3 \\ \beta_{2,STAB} = 3 + 3 = 6 \\ \dots \\ \beta_{k-1,STAB} = 3 + 3 = 6 \\ \beta_{k,STAB} = 3 + 3 = 6 \end{array} \right\} . \quad (19)$$

Each peer from the i^{th} tier of sub-overlay also has to run the FIXFINGER algorithm periodically to update its finger table. Originally, each FIXFINGER operation is a lookup operation for the ID of the required finger [1]. However, we do not consider the churn effect in this paper and assume all the peers are in steady state; therefore, each peer just has to send KEEPALIVE message to its fingers periodically. Similar to STABILIZE procedure, the *gateway super-peers* also have to run the FIXFINGER algorithm in each of the two participating SSOs periodically. In a SSO with n peers, the size of the finger table is $\log(n)$, and each FIXFINGER operation needs two messages, one from the initiator, the other from the responder. Therefore, we can get the following results:

$$\left\{ \begin{array}{l} \beta_{1,FIX} = 2 \cdot \log(n+1) \\ \beta_{2,FIX} = 2 \cdot \log(n+1) + 2 \cdot \log(n+1) \\ \dots \\ \beta_{k-1,FIX} = 2 \cdot \log(n+1) + 2 \cdot \log(n+1) \\ \beta_{k,FIX} = 2 \cdot \log(n+1) + 2 \cdot \log(N_k) \end{array} \right\} . \quad (20)$$

Republish traffic distribution

In order to make the stored service or key-value pair up-to-date, peers also have to operate the republish procedure periodically for each of their shared items. The republish procedure is almost the same as the lookup operation. The only difference is the republish time interval. Therefore, the relationship between γ_i and α_i is: $\gamma_i = \vartheta \cdot \alpha_i$.

Summary

In recursive routing, as the number of messages sent and received are the same, therefore, the total traffic distribution of one single peer from each tier of sub-overlay has the following result:

$$m_i = 2 \cdot \left(\frac{\alpha_i}{T_{LKP}} + \frac{\beta_{i,STAB}}{T_{STAB}} + \frac{\beta_{i,FIX}}{T_{FIX}} + \frac{\gamma_i}{T_{REP}} \right) . \quad (21)$$

The total number of messages sent and received by the whole overlay network is made up of two parts, the first part is the lookup traffic and republish traffic, the second part is the maintenance traffic, i.e. the STABILIZE traffic and the FIXFINGER traffic. Therefore, we can get the following results for the total traffic of the whole overlay network of GTPP and Chord:

$$\begin{aligned} M(n, k) &= 2 \cdot N \cdot \left[\left(\frac{\alpha}{T_{LKP}} + \frac{\vartheta \cdot \alpha}{T_{REP}} \right) + \sum_{i=1}^k \left(\frac{\beta_{i,STAB}}{T_{STAB}} + \frac{\beta_{i,FIX}}{T_{FIX}} \right) \cdot P_i \right] \\ M_{Chord} &= 2 \cdot N \cdot \left[\left(\frac{\log(N)}{T_{LKP}} + \frac{\vartheta \cdot \log(N)}{T_{REP}} \right) + \left(\frac{3}{T_{STAB}} + \frac{2 \cdot \log(N)}{T_{FIX}} \right) \right] . \end{aligned} \quad (22)$$

The maintenance traffic for GTPP and Chord per time unit is as follows:

$$\begin{aligned}
\text{Maintenance}(n,k) &= 2 \cdot N \cdot \sum_{i=1}^k \left(\frac{\beta_{i,STAB}}{T_{STAB}} + \frac{\beta_{i,FIX}}{T_{FIX}} \right) \cdot P_i \\
\text{Maintenance}(\text{Chord}) &= 2 \cdot N \cdot \left(\frac{3}{T_{STAB}} + \frac{2 \cdot \log(N)}{T_{FIX}} \right)
\end{aligned} \tag{23}$$

V. RESULTS ANALYSIS

In this section, we analyze the expected lookup hop count, expected lookup routing latency, traffic distribution of one single peer from a different tier of sub-overlay and the total traffic of GTPP architecture in comparison to flat architecture, based on the equations in Section IV. In the analysis procedure, we use three different network scales, 10^4 , 10^6 and 10^8 . However, we observe the similar curves with different network scales. Therefore, the subsequent sections just show the results with network scale 10^6 .

In the following figures, we look through all the integers between the lower and upper limit of n . The lower limit of n is two, which means there are at least two peers in each SSO from the lower tiers of sub-overlays to make the SSO function as a distributed system. The upper limit of n can be roughly determined by the following formula: $N_k = \left(\frac{1}{n}\right)^{(k-1)} \cdot N \geq 2$.

Take $k=2$ as an example, the upper limit of n is roughly 500,000, actually 499,999 is the exact number. In this case, the upper tier of SSO has 2 peers, while each SSO from the lower tier of sub-overlay has equally 499,999 peers.

Therefore, once the total number of the whole overlay N is fixed, a different depth of GTPP (k) will result in a diverse upper limit of n . The larger the value of k is, the smaller the upper limit of n is. Thus, the span of x -axis is different in the following figures.

A. Expected lookup hop count

In this part, the effect of different value of n and k (the depth of GTPP) to the expected lookup hop count is studied. We analyze four different overlay depths (k equals to 2, 3, 4, and 5, respectively), and the results are shown in the Fig. 3(a) - 3(d). The figures are based on Eq. (9) in Section IV. From the Fig. 3(a) – 3(d), on the one hand, we can see that the value of n does not affect the expected lookup hop count much. On the other hand, as the value of k becomes larger, the expected lookup hop count become slightly higher as well. The variation is proportional to the depth of GTPP, i.e. two tiers add around one more lookup hop compared to flat architecture (Chord as one instantiation), three tiers add two more hops, and so on. The primary reason for this is the additional forwarding hops sent to the top-most *gateway super-peer* by the peers locating at the lowest tier of sub-overlay. As in the example in Fig. 2, two more hops are needed for the request from N1_51 to reach the topmost (3rd) tier of sub-overlay. Because the peers from the lowest tier of sub-overlay account for the majority of GTPP, they have the primary impact on the expected lookup hop count of the whole overlay network.

The expected lookup hop count can be further decreased by caching the top-most *gateway super-peer* information (node ID, IP address, port) in each peer at the lower tiers of sub-overlays. In that case, the lookup request is sent to the topmost *gateway super-peer* directly from the lower tiers. Only one more forwarding hop is needed for all the peers from the lower tiers of sub-overlays. The resulting figures are always similar to Fig. 3(a) and are independent of the overlay depth (k).

Therefore, from the figures and analysis, we can conclude that GTPP has a slightly higher amount of lookup hop count compared

to flat architecture. The expected lookup hop count can be further decreased by caching the topmost *gateway super-peer* information at the lower tiers of sub-overlays.

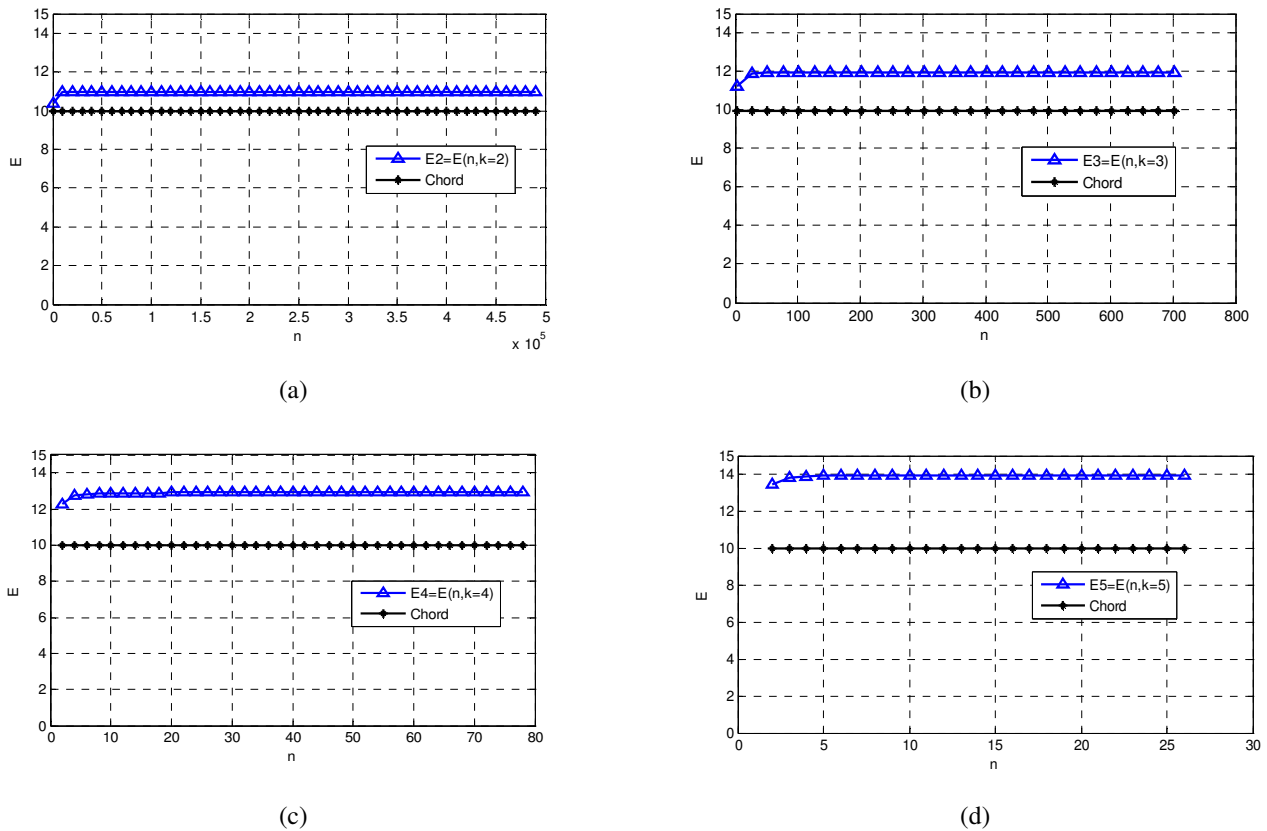


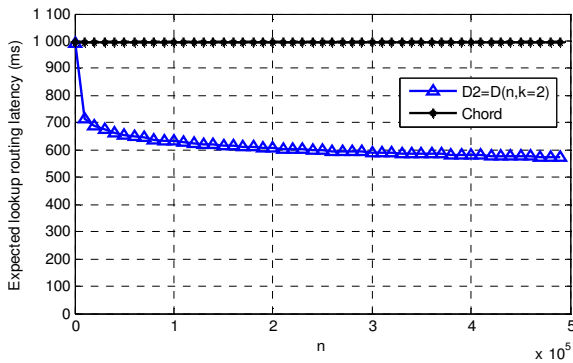
Fig. 3 (a). Expected lookup hop count ($k=2$, $N=10^6$); (b). Expected lookup hop count ($k=3$, $N=10^6$); (c). Expected lookup hop count ($k=4$, $N=10^6$); (d). Expected lookup hop count ($k=5$, $N=10^6$).

B. Expected lookup routing latency

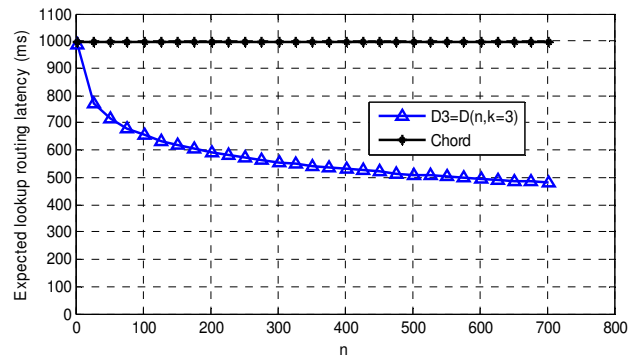
This section analyzes the impacts of different value of n and k to the expected lookup routing latency. We first fix the values of the RTT latency of one single hop at different tiers of sub-overlays. In HIERAS [9], Xu et al. found that in a two-tier hierarchical overlay architecture, with simple proximity approach, the average link delay in the lower ring was only 35.23% of the link delay of the higher layer. In real life, however, the link latency difference between local area network (LAN) and wide area network (WAN) can be even larger. Without loss of generality, we assume that the average RTT latency for one lookup hop in flat Chord architecture is 100 ms. In GTPP, we assume the same 100 ms RTT latency for one hop at the top-most (k^{th}) tier of sub-overlay, while the RTT latency for one hop at the $(k-1)^{\text{th}}$ tier of SSO is half of that, i.e. 50 ms, and so on. Therefore, the average RTT latency for one hop at the lowest (1^{st}) tier of sub-overlay is $(100/2^{k-1})ms$.

There is also some proximity approach that can be used in flat architecture to decrease the lookup routing latency. As we focus on the generalized version of PV-HA without optimizations in this paper, we just compare GTPP with generalized flat architecture without optimization either. The expected lookup latency of GTPP is shown in Fig. 4(a) - 4(d). We can see that both the values of n

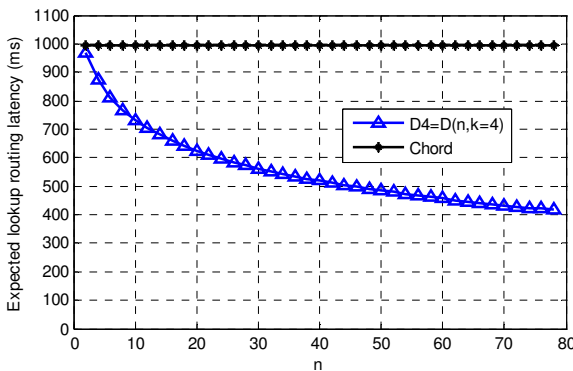
and k have effects on the expected lookup routing latency.



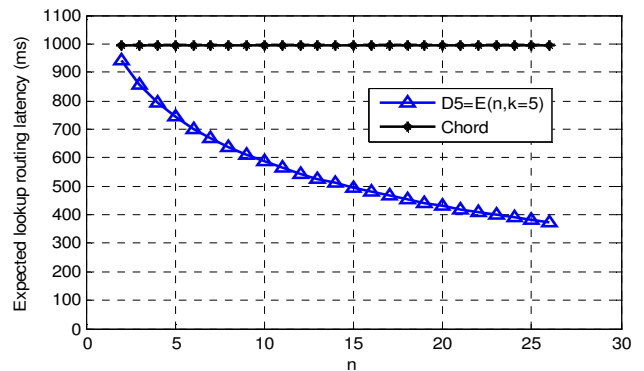
(a)



(b)



(c)



(d)

Fig. 4(a). Expected lookup routing latency ($k=2$, $N=10^6$, time unit=ms). (b). Expected lookup routing latency ($k=3$, $N=10^6$, time unit=ms). (c). Expected lookup routing latency ($k=4$, $N=10^6$, time unit=ms). (d). Expected lookup routing latency ($k=5$, $N=10^6$, time unit=ms).

From the perspective of n , the trends in each of Fig. 4(a)-4(d) are the same. As long as the value of n becomes larger, the expected lookup routing latency of GTPP becomes shorter. From the analysis procedure in Section IV.A, we know that n (or more precisely $n+1$) stands for the number of peers in each SSO at the lower tiers of sub-overlays (except the topmost SSO). As the value of n increases, more peers are located in the lowest tier of sub-overlay, which has the shortest average RTT latency in GTPP, therefore, the expected lookup routing latency of the GTPP architecture decreases as the value of n increases.

From the perspective of k , the trends are also clear. As the value of k increases, the optimal expected lookup routing latency decreases. It means the more tiers GTPP has, the shorter expected lookup routing latency it can achieve. The results are reasonable as the more tiers the GTPP architecture has, the shorter RTT routing latency it has at the lowest tier of sub-overlay. From Fig. 4(a) - 4(d), we can also see that, compared to flat architecture, two-tier GTPP architecture can achieve around 30% shorter routing latency compared to flat architecture for most of the values of n (see Fig. 4(a), from the point where the value of y-axis is 700 ms to the upper limit of n). When the value of k varies from 2 to 3, 4, the optimal value of routing latency varies from around 600ms to 500ms, and 400 ms, respectively. However, when we continue to increase the value of k from 4 to 5, the optimal value of routing latency varies slightly.

Therefore, from the viewpoint of routing latency, we can conclude that GTPP can achieve better performance compared to flat architecture. When the depth of GTPP increases from 2 to 4, the expected lookup routing latency becomes even shorter. However, after that, we cannot achieve even better performance just by increasing the overlay depth. Thus, 2-4 tiers are optimal for GTPP.

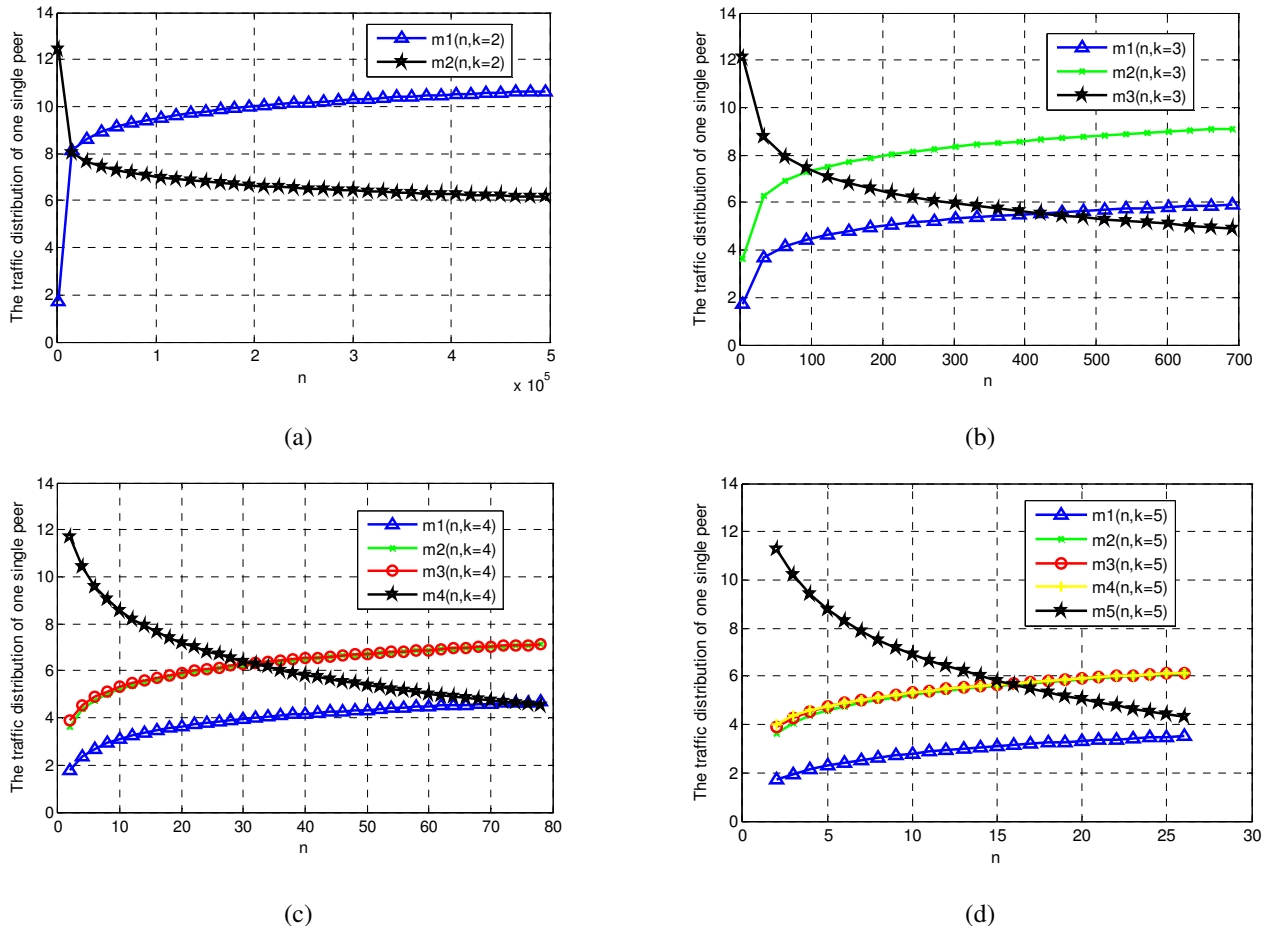


Fig. 5(a). Traffic distribution of a single peer (number of messages sent and received by each peer per second) from different tier of sub-overlay ($k=2$, $N=10^6$). (b). Traffic distribution of a single peer (number of messages sent and received by each peer per second) from different tier of sub-overlay ($k=3$, $N=10^6$). (c). Traffic distribution of a single peer (number of messages sent and received by each peer per second) from different tier of sub-overlay ($k=4$, $N=10^6$). (d). Traffic distribution of a single peer (number of messages sent and received by each peer per second) from different tier of sub-overlay ($k=5$, $N=10^6$).

C. Traffic distribution of each single peer from different tier of sub-overlay

In this section, we analyze the traffic distribution of each single peer from different tier of sub-overlay. We use the following experimental parameter values which are the same as in [16], unless otherwise mentioned. $T_{LKP} = 60s$, $T_{STAB} = 5s$, $T_{FIX} = 30s$, $T_{REP} = 300s$, $\vartheta = 50$. The results are based on the Eq. (21) in Section IV.C. In Fig. 5(a)-5(d), the y-axis, i.e. $m_i(n, k)$, stands for the average number of messages sent and received by each single peer per second from the i^{th} tier of sub-overlay. From Fig. 5(a)-5(d), we can see that both the values of n and k have influences on the results.

From the perspective of n , the trends of all the curves are the same, i.e. as long as the value of n increases, the traffic of each single peer from the lower tiers of sub-overlays increases, while the traffic of one single peer from the topmost tier of sub-overlay decreases.

At some points, they intersect. We can also see that the curves of the intermediate tiers of sub-overlays intersect with the curve of the topmost tier before the curve of the lowest tier does. It means the traffic of the intermediate tiers of sub-overlays increases promptly with the SSO size of the lower tiers increasing. Generally, in hierarchical architectures, the peers from upper tiers of sub-overlays are assumed to have more physical capabilities. Therefore, the crossover points in the figures above can be used to decide the overlay size of the lower tiers of sub-overlays to avoid overloading the peers from intermediate tiers of sub-overlays. Furthermore, in order to avoid overloading the peers from the top-most tier of sub-overlay, some other parameters should also be defined to get certain optimal operating point for the whole overlay network. The setting of these parameters will be included in our future work.

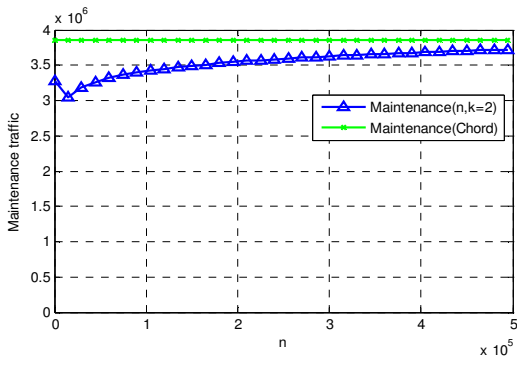
From the perspective of k , hierarchical architectures also show their benefits compared to flat architecture or H-HA. In reasonable functional domains (the overlay size less than the value of the crossover points), the higher tier one peer locates, the more traffic it shares. This makes PV-HA distribute traffic among the peers from different tiers of sub-overlays in a clearer and more reasonable way. From Fig. 5(a) and 5(b), it is clear that, when the depth of GTPP increases from 2 to 3, the traffic shared by the lowest tier of sub-overlay decreases significantly. When we continue increasing the depth of GTPP from 3 to 4, and 5, the alleviation of the traffic load from the lowest tier of sub-overlay is not so distinct. Although in Fig. 5(c) and 5(d), the peers from the middle tiers of sub-overlays almost share the same traffic; this is mainly because we assume each SSO from the lower tiers of sub-overlays have the same number of peers for the simplicity of analysis. This effect also proves, from another perspective, that hierarchical architectures should not have too many tiers to make a clear distinction of traffic distribution.

From the analysis above, we can conclude that, from the perspective of traffic distribution among different tiers of sub-overlays, GTPP has a clearer and more reasonable way than flat architecture and 2-3 tiers are optimal to keep this benefit.

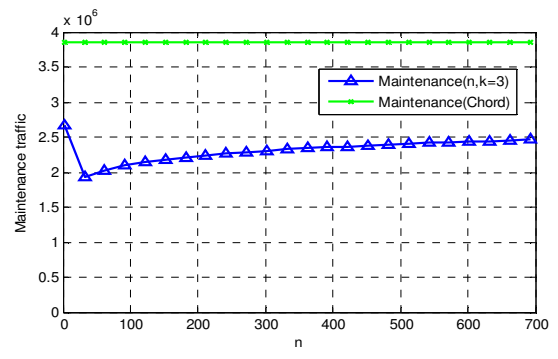
D. Total traffic of the whole overlay network

In the section, we analyze the total traffic and maintenance traffic of GTPP in comparison to Chord. The results are based on Eq. (22) and Eq. (23) in Section IV.C. We use two different lookup rates $T_{LKP} = 60s$ and $T_{LKP} = 1s$ for the total traffic analysis. The other parameters are the same as in Section V.C., i.e. $T_{STAB} = 5s$, $T_{FIX} = 30s$, $T_{REP} = 300s$, $\vartheta = 50$. We firstly show the results of the maintenance traffic per second (number of messages sent and received per second for maintenance) in Fig. 6(a)-6(d). From Fig. 6(a)-6(d), it is clear that the maintenance traffic of GTPP is smaller than Chord. As long as the overlay depth k of GTPP increases, the overall maintenance traffic decreases as well. This trend is evident when we vary the value of k from 2 to 3, and 4. The primary reason for this is that, when we increase the overlay depth of GTPP, the peers are partitioned into smaller sizes of SSOs, which have fewer neighbors to maintain compared to larger SSOs. Therefore, the maintenance decreases remarkably.

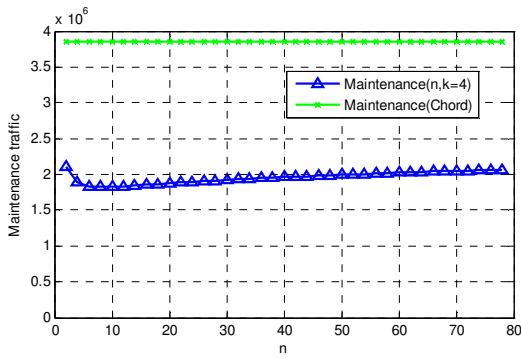
Now, let us consider the total traffic (number of messages sent and received per second for all the peers) of GTPP compared to Chord. The results with $T_{LKP} = 60s$ are shown in Fig. 7(a)-7(d), while the results with $T_{LKP} = 1s$ are shown in Fig. 8(a)-8(d).



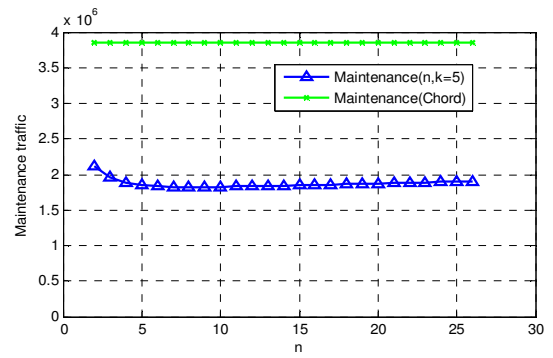
(a)



(b)

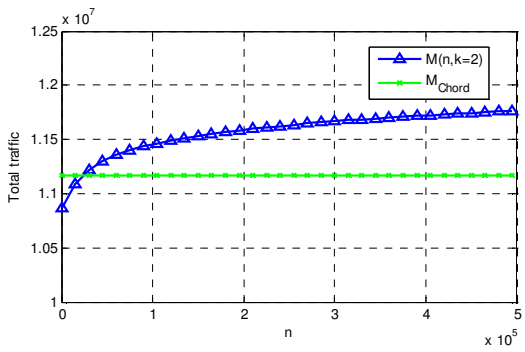


(c)

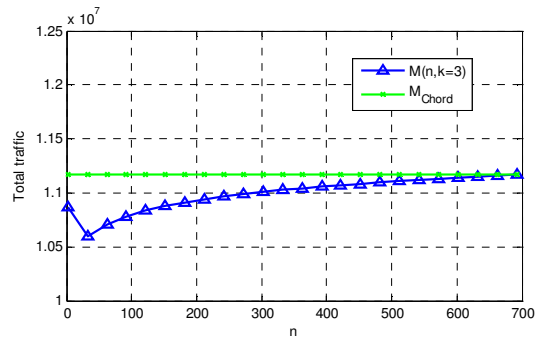


(d)

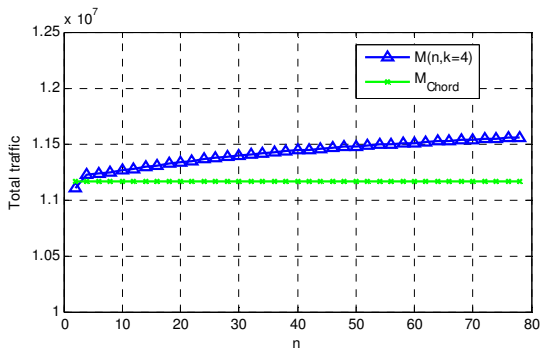
Fig.6 (a).Maintenance traffic GTPP vs. Chord ($k=2, N=10^6$). (b). Maintenance traffic GTPP vs. Chord ($k=3, N=10^6$). (c). Maintenance traffic GTPP vs. Chord ($k=4, N=10^6$). (d). Maintenance traffic GTPP vs. Chord ($k=5, N=10^6$).



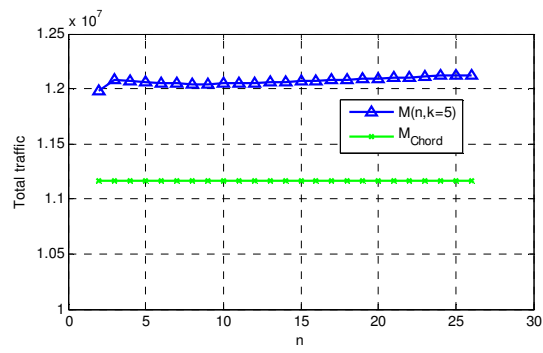
(a)



(b)

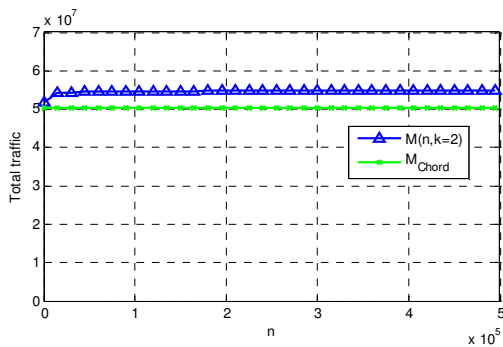


(c)

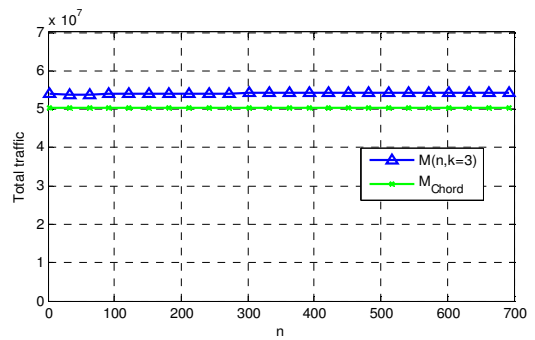


(d)

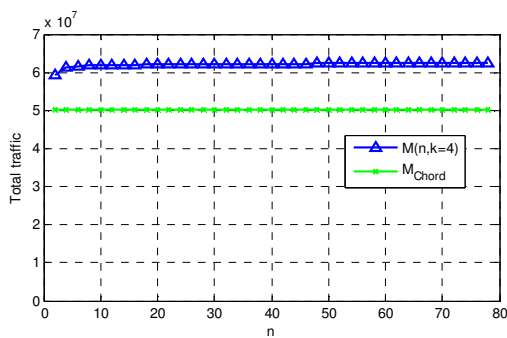
Fig. 7(a). Total traffic per second GTPP vs. Chord ($k=2$, $N=10^6$, $T_{LKP}=60s$). (b). Total traffic per second GTPP vs. Chord ($k=3$, $N=10^6$, $T_{LKP}=60s$). (c). Total traffic per second GTPP vs. Chord ($k=4$, $N=10^6$, $T_{LKP}=60s$). (d). Total traffic per second GTPP vs. Chord ($k=5$, $N=10^6$, $T_{LKP}=60s$).



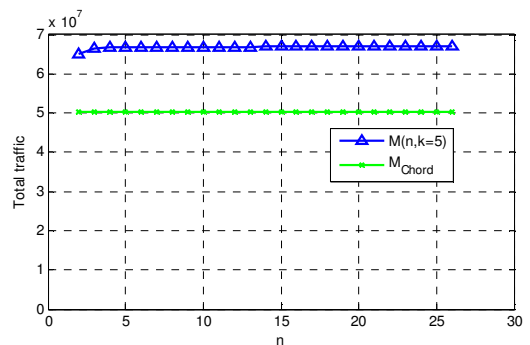
(a)



(b)



(c)



(d)

Fig.8(a). Total traffic per second GTPP vs. Chord ($k=2$, $N=10^6$, $T_{LKP}=1s$). (b). Total traffic per second GTPP vs. Chord ($k=3$, $N=10^6$, $T_{LKP}=1s$). (c). Total traffic per second GTPP vs. Chord ($k=4$, $N=10^6$, $T_{LKP}=1s$). (d). Total traffic per second GTPP vs. Chord ($k=5$, $N=10^6$, $T_{LKP}=1s$).

From Fig. 7(a)-7(d), we can see that the total traffic of GTPP slightly decreases when the depth increases from 2 to 3, and then significantly increases when the depth varies from 3 to 4, 5. When the depth is 3, the total traffic of GTPP is optimal and is slightly smaller than Chord. From Fig. 8(a)-8(d), it is clear that the total traffic of GTPP is greater than Chord. However, the depth of 2 or 3 still show its benefits as the difference of total traffic between GTPP and Chord is not so big. The primary reason behind these phenomena is that the total traffic is made up of two parts, the lookup and republish traffic, and the maintenance traffic. The lookup

and republish traffic is larger in GTPP than in Chord (see Fig. 3(a)-3(d)), while the maintenance traffic of GTPP is smaller than Chord (see Fig. 6(a)-6(d)). In Fig. 7(a)-7(d), where $T_{LKP} = 60s$, the lookup traffic is not the predominant part of the total traffic; while in Fig.8(a)-8(d), where $T_{LKP} = 1s$, the lookup traffic is the dominant part and accounts for the majority of the total traffic. Therefore, much more traffic is caused in GTPP than in Chord.

E. Summary

From the figures and analysis above, we can conclude that GTPP, the generalized version of PV-HA, has significantly decreased lookup latency though the expected lookup hop count is slightly larger than flat architecture. GTPP also shows its advantage in heterogeneous environments as it has a clear and distinct traffic distribution among the peers from different tier of sub-overlay. As for the maintenance traffic, GTPP still has its advantage compared to flat architecture, i.e. smaller maintenance traffic. From the perspective of total traffic of the whole overlay network, GTPP has slightly larger traffic than flat architecture. However, when the depth of GTPP is 2 or 3, the added total traffic is still tolerable. Considering all the parameters analyzed in this paper, 2-3 hierarchical tiers are most suitable for GTPP in most cases.

VI. SIMULATION RESULTS

In order to substantiate the claims made in the previous sections, we present in this section the simulation results of the GTPP and a tentative community management system which utilizes two tiers of sub-overlays.

In Fig. 9, we present the simulated expected lookup hop count with different GTPP sub-overlay setups. Due to the limitations of the server capabilities, only the network size of 500 nodes is simulated.

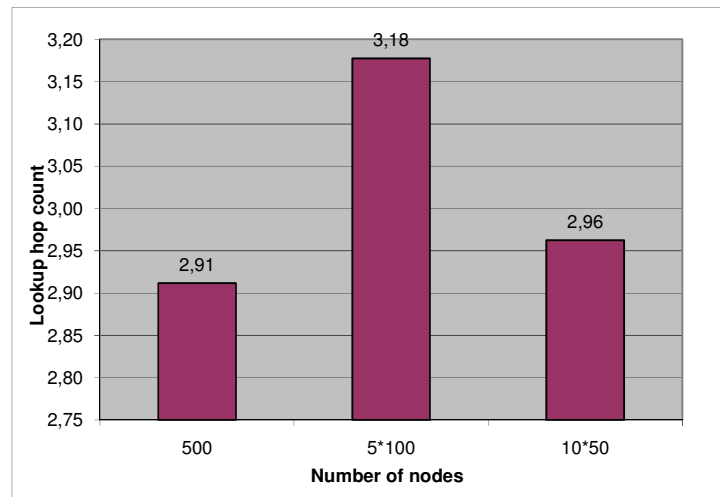


Fig. 9. Lookup hop count of GTPP vs. flat architecture.

In Fig. 9, the first column, labeled “500”, stands for the flat architecture where all the 500 nodes are located in one single overlay; while the second column, denoted by “5*100”, stands for the case where the upper tier of sub-overlay has 5 nodes, and each of the 5 lower tier of sub-overlays has equally 100 nodes; the last column, labeled “10*50”, in turn, stands for the case where the upper tier of sub-overlay has 10 nodes, and each of the 10 lower tier of sub-overlays has equally 50 nodes. Fig. 9 indicates that using the GTPP

(illustrated by the second and the third column) leads to a slightly higher lookup hop count compared to the flat architecture (the first column). This clearly confirms the claim we made in Section V.A., although with optimizing the sub-overlay setup, the expected lookup hop count can be decreased closer to that of flat architectures (illustrated by the third column).

To evaluate the maintenance traffic of hierarchical architecture, a tentative community management system is implemented. The core idea of such a system is to establish a separate DHT overlay for each community resulting in multiple small overlays that are subsets of the main overlay [26]. The main overlay is primarily used for discovering community and membership information. In this case, it is more like a FV-HA than a PV-HA, however, the simulation results are still convincing to prove the fact that the maintenance traffic of a large overlay can be decreased by grouping the peers into small sub-overlays. We have conducted measurements with a mobile implementation of the P2PSIP peer protocol candidate, called Peer-to-Peer Protocol [27] (P2PP, has been merged into another draft names RELOAD [28]) to evaluate the feasibility of the community management system. As a unit of measurement, the average number of messages processed by each node per second was used. Due to the server facilities, the maximum network size was limited to 500 nodes. The results are shown in Fig. 10, where the network overlay comprised of 500 nodes and 50 communities, each containing 10 nodes. In the first case, the communities were implemented within a single overlay, and in the second and the third case, the communities were implemented using separate community overlays.

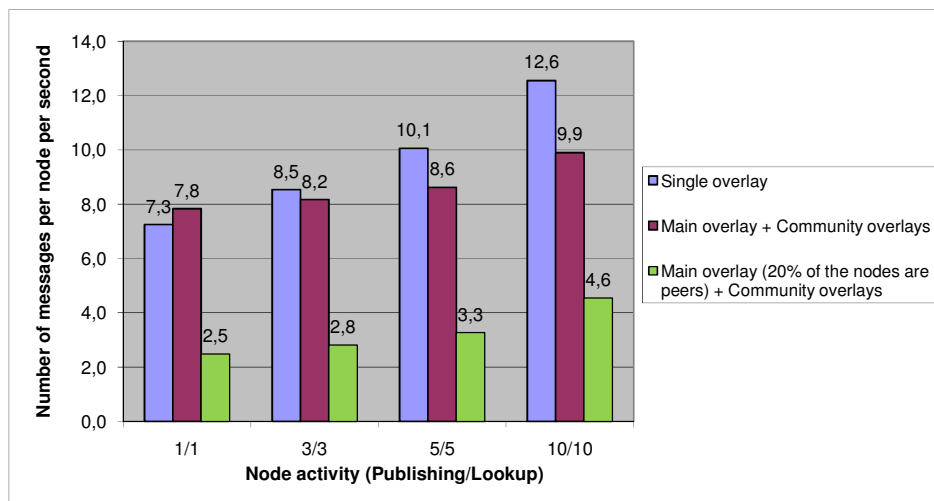


Fig.10. Number of messages processed by each node per second

Fig. 10 indicates that the usage of separate community overlays is beneficial when the level of community-related node activity is high (i.e. more lookup and publish operations), and especially when the amount of unnecessary maintenance traffic is reduced in the main overlay. The former was due to the fact that the routing is more efficient in smaller community overlays than in a single, large main overlay. The latter was achieved by changing peer nodes into clients (nodes that do not take part in the overlay maintenance, just send keep-alive messages to their connected peers). Since the testing was conducted only with small-scale networks, it has to be noted that the benefits achieved with more efficient routing would have been greater in large-scale networks. Therefore, from Fig. 10, we can conclude that the usage of hierarchical architecture results in less maintenance traffic and more efficient intra-domain (local SSO) routing, when compared to flat architecture. This proves our conclusion concerning the

maintenance traffic in Section V.D.

VII. DISCUSSION

In this section, we discuss the effects of content caching, iterative routing mechanism and churn to the performance of GTPP.

A. Content caching

To make full use of the advantage of content caching mechanism in PV-HA, we make a minor modification to the lookup procedure defined in Section III.B. When a request is initiated from a lower tier of sub-overlay, it is looked up in its local SSO firstly. If a successful response is returned, the lookup procedure is halted immediately. If no successful response is returned in the local SSO, the request is forwarded to its *gateway super-peer*. The remaining lookup procedure follows the same steps as in Section III.B. After the successful response is returned from some other SSO, the key-value pair or the file (in file sharing application) is cached at the peer in the local SSO whose ID has the closest distance with the key (until it is replaced by a newer copy). In this case, the next time when the other peers from the same local SSO initiate requests targeting at the same service, the lookup procedure can be halted within the local SSO. This mechanism results in a high hit rate in an environment where the vast majority of the lookup operations are aimed at the popular service, e.g. file sharing application. Meanwhile, as the RTT of one single hop in the lower tiers of SSOs is much less than the flat architecture, the hierarchical architecture with a content caching mechanism can further decrease the lookup routing latency.

B. Iterative routing

We assumed the recursive routing mechanism in Section III.B and utilized this mechanism for the whole analysis procedure in the previous sections. In this section, we will evaluate the effect of iterative routing mechanism to the results we achieved in the previous sections.

We summarize the main differences between iterative routing and recursive routing as follows: intermediate peers in the iterative routing do not forward the request to the next hop peer, instead, they return the next hop peer information to the initiating peer; the initiating peer then sets up connection to the next hop peer by itself and the lookup procedure is executed iteratively. While in the recursive routing, intermediate peers forward the request to the next hop peer, after getting response from the next hop peer, they then return the response to the previous hop peer; the lookup procedure is executed recursively.

In GTPP, or in hierarchical architecture, inter-domain (or inter-SSO) iterative routing is not suitable because of the following reasons: Firstly, inter-domain routing mechanism results in longer RTT latency which weakens the advantage of shorter intra-domain routing latency in hierarchical architectures; Secondly, some peers in hierarchical architectures are located behind security firewall or NAT, which makes it impractical to utilize the inter-domain iterative routing. Therefore, in this section, we just analyze the intra-domain iterative routing.

In intra-domain iterative routing, half of the messages are sent and received by the initiating peer, while the other half is shared by the other peers included in the lookup procedure of the SSO. However, from the viewpoint of the whole overlay network, the total

messages sent and received are the same. For each hop, two messages are sent and received respectively which is the same as the recursive routing. For each lookup operation, the total hop count is also the same, regardless of whether iterative routing or recursive routing is utilized. Meanwhile, the intra-domain RTT is the same as in recursive routing. Therefore, iterative routing does not have an effect on the results we got in Section V.A, V.B, and V.D. It just has an effect on the results of Section V.C, i.e. traffic distribution of each single peer from different tier of sub-overlay. Even in this case, the effect of intra-domain iterative routing mechanism is minor, because it just influences the top-most tier of sub-overlay and the lowest tier of sub-overlay. The top-most tier of SSO shares half the traffic which is originally shared by the lowest tier of SSO in recursive routing. The number of messages sent and received by the intermediate SSOs is the same. The result is that more traffic is shared by the peers from the topmost tier of sub-overlay, and this, in turn, decreases the traffic load of peers at the lowest tier of sub-overlay. Therefore, intra-domain iterative routing mechanism strengthens our conclusion drawn in Section V.C, i.e. hierarchical architecture has clearer and more reasonable traffic distribution among the peers from different tiers of sub-overlays than flat architecture.

C. Churn

We did not consider the churn effect in this paper. The performance of GTPP in the presence of churn is included in our future work. However, as hierarchical architecture groups the more powerful and reliable peers into the upper tiers of sub-overlays, the churn effect can be largely isolated to the lower tiers of SSOs. As a matter of fact, Garcés-Erice et al. [8] proved that, in the presence of churn, the expected lookup hop count in a two-tier hierarchical architecture was significantly decreased in comparison to flat architecture. Therefore, we argue here that the conclusions we drew in this paper can also be applied to the hierarchical architectures in the presence of churn. We will verify this argument in our future work.

VIII. FUTURE WORK

This paper provides a theoretical mathematical analysis of the proposed GTPP architecture, the generalized version of PV-HA. As a result of the analysis, this paper presents useful results about the GTPP's performance, especially regarding the expected lookup hop count, lookup routing latency, traffic distribution of each single peer from different tiers of sub-overlays, and the total traffic of GTPP vs. flat architecture. In the future, we will continue to analyze the performance of GTPP in the presence of churn, and the parameter set for defining the optimal operating point of the whole hierarchical overlay network. Meanwhile, we will change some of the assumptions in section IV of this paper, to make the analysis and evaluation more realistic. Network simulations and real-world prototypes will be implemented to deepen the evaluation of these aspects. Furthermore, H-HA will also be analyzed to provide a comparative analysis for the PV-HA.

IX. CONCLUSIONS

In this paper, we continued the development of our GTPP architecture, the generalized version of PV-HA. The idea was to study whether added tiers of hierarchy could provide added value in performance and functionality. We adopted the maximum information entropy principle to distribute all the peers into multiple sub-overlays. Based on the similarity of ED and GD, we

utilized the GD to distribute peers into different tiers approximately. Through mathematical analysis, we demonstrated performance results in comparison to flat architecture, which helped understanding some of the most typical characteristics of hierarchical architectures as compared to flat architectures:

- GTPP has a slightly higher expected lookup hop count, although with optimizing the sub-overlay setup, the expected lookup hop count can be decreased closer to that of flat architectures;
- GTPP can significantly decrease the expected lookup routing latency, due to the differences in the RTT latency between the nodes at different tiers of sub-overlays;
- GTPP has a clearer and more reasonable traffic distribution among all the peers of different tiers of sub-overlays;
- GTPP slightly decreases the maintenance traffic.

In conclusion, 2-3 tiers of GTPP were considered as the most suitable configuration in most cases, when all the parameters were included.

X. ACKNOWLEDGEMENT

The authors would like to thank TEKES (the Finnish Funding Agency for Technology and Innovation), Ericsson, Nokia and Nethawk for their financial support in Decicom project. The authors would also like to thank Dr. Jie Chen for his valuable advice in improving the formulas of the paper and the anonymous reviewers for their helpful comments.

REFERENCES

- [1] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F., Balakrishnan, H.. Chord: a scalable peer-to-peer lookup protocol for Internet applications. *IEEE/ACM Transactions on Networking*, Volume 11, Issue 1, Feb. 2003 Page(s):17 - 32.
- [2] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of SIGCOMM 2001*, Aug. 2001.
- [3] P. Maymounkov and D. Mazires. Kademia: A peer-to-peer information system based on the xor metric. In *Peer-to-Peer Systems: First International Workshop, IPTPS 2002 Cambridge, MA, USA, March 7-8, 2002*. Page(s): 53 - 65.
- [4] Rowstron A., Druschel P.. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, Nov. 2001, Page(s):329 - 350.
- [5] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," Tech. Rep. UCB/CSD-01-1141, Computer Science Division, University of California, Berkeley, Apr 2001.
- [6] B. Lampson, "Designing a global name service," in *Proc. 4th ACM Symposium on Principles of Distributed Computing*, Minaki, Ontario, 1986, pp. 1–10.
- [7] P. Ganesan, K. Gummadi, H. Garcia-Molina, Canon in G major: designing DHTs with hierarchical structure, in: *IEEE International Conference on Distributed Computing Systems (ICDCS 2004)*, Tokyo, Japan, 2004, pp. 263–272.
- [8] L. Garcés-Erice, E. Biersack, K. Ross, P. Felber, G. Urvoy-Keller, Hierarchical P2P systems, in: *ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par)*, Klagenfurt, Austria, 2003.
- [9] Z. Xu, R. Min, and Y. Hu. HIERAS: A DHT-Based Hierarchical Peer-to-Peer Routing Algorithm. *Proceedings of 2003 international conference on parallel processing (ICPP 2003)*, 2003. Page(s):187-194.
- [10] A. Mislove, P. Druschel, Providing administrative control and autonomy in Peer-to-Peer overlays, in: *International Workshop on Peer-to-Peer Systems (IPTPS'04)*, San Diego, CA, 2004.
- [11] S. Zoels, S. Schubert, W. Kellerer, Z. Despotovic, Content availability and signaling overhead in DHT systems for mobile environments, in: *ACM Symposium on Principles of Distributed Computing (PODC 2006)*, Denver, CO, 2006.
- [12] Z. Ou, E. Harjula, M. Ylianttila. GTPP: General truncated pyramid architecture over P2PSIP networks. *The International Conference on Mobile Technology, Applications & Systems 2008 (Mobility Conference)*, 10-12 September, 2008, Ilan, Taiwan.

- [13] Z. Ou, J. Zhou, E. Harjula, M. Ylianttila. Truncated Pyramid Peer-to-Peer Architecture with Vertical Tunneling Model. The Fourth International Workshop on Dependable and Sustainable Peer-to-Peer Systems (DAS-P2P 2009) on CCNC 2009. 10–13, January 2009 in Las Vegas, Nevada. (to appear).
- [14] J. W. Lee, H. Schulzrinne, W. Kellerer and Z. Despotovic. mDHT: Multicast-augmented DHT Architecture for High Availability and Immunity to Churn. CCNC 2009. 10–13, January 2009 in Las Vegas, Nevada. (to appear).
- [15] S. Zoels, Z. Despotovic, W. Kellerer, Cost-based analysis of hierarchical DHT design, in: IEEE International Conference on Peer-to- Peer Computing (P2P 2006), Cambridge, UK, 2006.
- [16] S. Zoels, Z. Despotovic, W. Kellerer, On Hierarchical DHT Systems- An Analytical Approach for Optimal Designs. Computer Communications 31 (2008) 576-590.
- [17] Seungwon Shin, Jaeyeon Jung, Hari Balakrishnan. Malware prevalence in the KaZaA file-sharing network. IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement. October 25–27, 2006, Rio de Janeiro, Brazil. Page(s):333-336.
- [18] Chao Xie; Yi Pan; ISE02-5: Analysis of Large-Scale Hybrid Peer-to-Peer Network Topology. IEEE Global Telecommunications Conference, 2006. GLOBECOM '06. Nov. 27 2006-Dec. 1 2006. Page(s):1 - 5.
- [19] B. Krishnamurthy, J. Wang, and Y. Xie, “Early measurements of a cluster-based architecture for p2p systems,” in ACM SIGCOMM Internet Measurement Workshop, (San Francisco, CA), November 2001.
- [20] M. Sa´nchez-Artigas, P. Garc´ıa, J. Pujol, A.F. Go´mez Skarmeta, Cyclone: a novel design schema for hierarchical DHTs, in: IEEE International Conference on Peer-to-Peer Computing, Konstanz, Germany, 2005.
- [21] B. Y. Zhao, Y. Duan, L. Huang, A. D. Joseph, and J. D. Kubiawicz, “Brocade: Landmark routing on overlay networks,” in In Proceedings of IPTPS'02, (Cambridge, MA), March 2002.
- [22] Le, L.; Kuo, G.-S.. Hierarchical and Breathing Peer-to-Peer SIP System. IEEE International Conference on Communications, 2007. ICC '07. 24-28 June 2007 Page(s):1887 – 1892.
- [23] Bryan D., Matthews P., Shim E., Willis D. (2007) Concepts and Terminology for Peer to Peer SIP. Internet Draft, draft-ietf-p2psip-concepts-01
- [24] V. Lo, Zhou D.-Y, Liu Y.-H, Dickey C. -G and Li J. Scalable Supernode Selection in Peer-to-peer Overlay Networks. Hot topic in Peer-to-Peer System, 2005.
- [25] E. T. Jaynes. Information Theory and Statistical Mechanics. Physical Review, vol. 106, no. 4, pp.620-630, May 15, 1957.
- [26] Koskela, T., Kassinen, O., Korhonen, J., Ou, Z., & Ylianttila, M. (2008). Peer-to-Peer Community Management using Structured Overlay Networks. In Proceedings of the International Conference on Mobile Technology, Applications and Systems (MOBILITY), Yilan, Taiwan.
- [27] Baset, S., Schulzrinne H., and Matuszewski, M. 2007. Peer-to-Peer Protocol (P2PP), IETF Internet Draft (19 November 2007). URL: <http://tools.ietf.org/html/draft-baset-p2psipp2pp-01>. (work-in-progress)
- [28] C. Jennings et al. REsource LOcation And Discovery (RELOAD). IETF Internet Draft (June 10, 2008). URL: <http://tools.ietf.org/html/draft-bryan-p2psip-reload-04>. (work-in-progress)