DHT or Flooding: A Comparative Study of Name Resolution Approaches in Information Centric Networks

Dan Zhang
WINLAB, Rutgers University
671 US-1 South, North Brunswick, NJ 08902
Email: zhangdan@gmail.com

Hang Liu
InterDigital Communications, LLC
781 Third Ave, King of Prussia, PA 19406
Email: hang.liu@interdigital.com

Abstract—**Name resolution techniques in Information Centric Networks (ICNs) have split into two themes. In one theme as adopted by the Content Centric Network (CCN) architecture, content availability is advertised to all content routers in a network via a flooding protocol in support of name-based routing. In the other theme featuring the deployment of distributed name resolution servers, content location information is inserted to one or more servers, and subsequent requests must be resolved thereby. While remarkable research has taken place in both directions, there still lacks a quantitative model to characterize the bandwidth overhead associated with the two name resolution approaches. From a networking practitioner's point of view, the bandwidth overhead can be decomposed into two parts, one for data delivery and the other for content name resolution. Minimizing the latter entails proper content object naming and name aggregation, as done with IP addresses. This paper proposes models for quantifying the overhead associated with name resolution, without and with name aggregation. Based on these models, this paper also makes baseline comparisons of the two major name resolution approaches in ICNs in terms of their bandwidth overhead. Our comparison reveals essential design tradeoffs and principal design guidelines.**

Keywords-information centric network, DHT, flooding, name resolution

I. Introduction

Future Internet has been envisioned as built around content [1][2][3], instead of host-to-host connections that characterize today's Internet. This new paradigm of networking is motivated by observing that the dominant traffic of Internet is no longer host-to-host communications, but content requests and deliveries for which the primary interest to the end users is the content objects themselves. Users usually have least interest in where these content objects are hosted and how they are handled in the network, as long as they get exactly what they need in a prompt and robust manner. Host oriented networking architecture has limited potential to meet this drastically increasing expectation. A number of clean-slate information-centric network (ICN) architectures [1]-[8] have thus been proposed. While these proposals are diverse in nature, reflecting different utility perceptions of the future Internet by researchers, the core philosophies are essentially the same -- an Internet that serves content in the most versatile and efficient way. Two streams of ICN architecture studies have been dominant in current literature. The flooding based architecture, represented by Content Centric Network (CCN) [1], assigns content objects with distinct hierarchical names that are mapped from human-readable names by some coding rules. The hierarchical names enable flexible user-content interaction. Reuse of OSPF-like routing algorithm (hence a smooth upgrading) and transparent name aggregation are also made possible, much in the same way of longest prefix match for IP addresses. When flooding is used, a content router (CR) advertises the names of content objects which it can serve from its repository to other CRs as an IP router advertises its link states. Any other CRs can then forward the request for a content object to the best content source(s) based on the requested content name. Strictly speaking, a repository may or may not be a standalone detached from the CR and it may even be possible to be co-located with the CR. For the purpose of our paper, we do not make this distinction and assume each CR has its own repository which is nothing more than a massive storage device. We also assume that the CR takes sole responsibility handling the content objects in its repository including, among other things, generating new content objects and removing outdated content objects. Although flooding based routing has been successful for IP networks, flooding faithfully the names of all the content objects can be impractical given the sheer quantity that can easily exceed the number of IP addresses by several magnitudes. Furthermore, the content objects may be dynamically generated and deleted in a network due to caching and cache replacement.

An equally competitive architecture, motivated by research in P2P networks [4][5][6][7], is thus designed to avoid the unbearable flooding traffic, using the so-called Distributed Hash Tables (DHT). DHT provides a distributed lookup service. It uses a hash function to associate an identifier, also called a key, to a content object and maps the identifier onto a node based on the node identifier. Content location resolution can be implemented with DHT by storing the content name-location binding information at the node to which the identifier maps. The node is also referred to as the resolver. One way to generate the content mapping identifier for a content object is to hash the hierarchical name or other types of names of the content object. The content identifier is then mapped to the resolver whose identifier is the closest to and not exceeding the content identifier in the hash space. A CR informs the mapped resolver for a content object it will serve through *insertion/publishing* process. Any participating CR can identify the resolver for a requested content object using its hashed content identifier. It can retrieve the content location information (an IP address or a more general directive for forwarding) from the resolver, and then the content object itself from the source. The hashing process makes the identifiers uniformly mapped to the set of resolvers.

It is possible to map and store the content location information to multiple DHT resolvers for better fault tolerance. Our discussion in this paper is based on the one-to-one mapping, which is simpler yet captures the central idea. Extension to one-to-many mapping is straightforward based on our discussion here. As DHT does not require flooding in content publishing process and a CR only need communicate to the necessary resolvers about the names of the content objects it provides, there seems to be a dramatic decrease in bandwidth overhead compared with flooding. However, DHT requires additional actions to complete name resolution during the content retrieval process – whenever an end user wants a content object, it has to send a name resolution query to the corresponding resolver, before a content request can be sent to the target CR. If we compare flooding and DHT based resolution techniques, we immediately see a major design tradeoff. Flooding of content availability information in a network potentially wastes bandwidth but saves the need of name resolution query. Note that we consider flooding is a name resolution scheme that lets all other CRs know the content location information. It therefore remains unclear which is better under what condition, causing a lasting debate between the two schools of thoughts.

A possible remedy that lowers the wasteful flooding traffic is inspired by IP aggregation. If a set of content names can be justifiably aggregated before being advertised throughout the network, a substantial saving in bandwidth may be achieved. Fortunately, the hierarchical name structure enables natural aggregation opportunity by using prefix aggregation, a technique that has become fundamental for today's longest prefix routing in Internet. One exception is necessary. Due to transient nature and large quantity of networked content objects, pure prefix aggregation that guarantees faultless name resolution is not sufficient. In fact, a certain amount of resolution error is tolerable in exchange for scalability in name resolution overhead. The Bloom filter [11] [12] is one of such lossy aggregation techniques that effectively generates the summarization of a set of names, with controllable error probability. Even with DHT based name resolution, the Bloom filter technique may still prove valuable. The question that is common to both resolution mechanisms is, given a set of names, should we aggregate them or not. If the answer is yes, what would be the best aggregation scheme in terms of the Bloom filter and error probability design. This paper also addresses these issues in the context of both flooding and DHT resolutions.

This paper, to authors' knowledge, is the first work that attempts to perform a comparative study of name resolution and routing approaches in ICN networks and tries to answer a few questions quantitatively: (1) what are the design tradeoffs between flooding and DHT-based name resolution schemes under different network scenarios, and (2) what are the impacts of name aggregation to these name resolution schemes. For completeness, we would like to point out that some quantitative models have been proposed for the throughput of flooding based ICNs, such as CCN [9][10]. But these previous works focused on the cost of data delivery, but our work focuses on name resolution, i.e. control overhead of these networks.

II. Network Models

In this section we describe the respective network models to implement flooding and DHT name resolution mechanisms. The discussion is necessarily simplistic, capturing only the essential properties that are relevant to the problems addressed in this paper. The flooding network model is similar to IP networks, consisting of so called content routers (CRs). These routers not only execute the routing protocol but also take a number of functionalities not found in IP routers. In particular, these CRs are responsible for managing the content objects in the attached network and advertising them. DHT network has routers as well as a special type of elements called resolvers,. Routers are still responsible of managing the content objects generated in its attached networks, but only inform the corresponding resolvers about the existence of the content objects. Note that the resolver function can be distributively co-located within CRs, i.e. a CR is also one of the resolvers in the DHT network. For generality and ease of explanation, we separate these two functions. Specifically, a router informs a particular resolver about the availability of a content object if and only if the content identifier is mapped to the resolver in the hash space. When a router wants to retrieve a content object, it queries the corresponding resolver for the content's location (e.g., the address of the router that manages this piece of content) and forwards the request to the target router. We are mainly concerned with the name resolution overhead in this paper, i.e., we do not quantify the bandwidth overhead incurred in forwarding the request to the content holder and retrieving the content back, partially because the mechanisms for these purposes are applicable to both flooding based and DHT based networks, therefore they are not sufficient to distinguish the capabilities of the two name resolution schemes or their apt use cases. Rather, we derive mathematical models for the overhead associated with flooding advertisement as well as name insertion and name resolution request – basic operations that have been commonly agreed upon that characterize the underlying network architecture. Our analysis shows how the overhead scales with the size of the network or the number of content objects. It also offers a few insights into important design considerations, including content naming, partial aggregation and resolution error rate optimization.

We consider an ICN network with stable network topology i.e. the CRs are fixed and stable, while the locations of content objects change due to dynamic content generation and cache replacements. The nodes that participate in the DHT are stable, and the overhead for maintaining the DHT structure is very small compared to the traffic for content location publishing and query. Note that this is different from a P2P network where peer churns are frequent.

Specifically, our network is modeled as a directed graph $\Gamma = (\mathrm{N},\mathrm{E})$ with $N = |\mathrm{N}|$. Content objects are created on a CR, kept for some time, and then removed. In a steady state, we assume the total number of content objects is M. Thus we can always label the contents in the network by $c_1, c_2, \ldots, c_M$ for the purpose of explaining our modeling. Note that the content label and content name is different. A label c_i always refers to a content object on a particular router, for example, c_1 represents a content object on CR$_1$. However the underlying content object (and its name) labeled by c_i, can be changed. In the steady state, we can nevertheless make the assumption that the request rate for content c_i is λ_i and that the new content object under the label c_i is generated at a rate of α_i. By this assumption, we are essentially saying that when the network has entered the steady state, not only the number of content objects is roughly constant, so are their popularity and generation rates. For content generation, it is understood that when a new content is generated under the label of c_i, the content previously labeled c_i is removed from the network. Therefore, we may assume the generation rate is equal to the replacement rate. This assumption is critical for this paper. Therefore, the authors would like to give further intuitive justification as how specific content objects are identified with their appropriate labels: The content objects on a particular router can be partitioned into groups $C_{j\ell}$ such that group $C_{j\ell}$ contains only content objects that have roughly the same popularity (request rate) λ_j and life span s_ℓ. When the Internet has entered the steady state and given the number of content objects managed by a router is usually huge, we expect the group sizes remain constant. Although the actual content object represented by $C_{j\ell}$ is time variant, at any time, each content object in $C_{j\ell}$ can be assigned a specific label c_i with $\lambda_i = \lambda_j$ and $\alpha_i = 1/s_\ell$. This construction also shows that the definition of generation rate α_i only makes sense from the ensemble view.

Let's consider both name resolution schemes, flooding and DHT, not preclude variable length names. We will analyze both fixed-length naming and variable-length naming schemes. The resolution overhead studied in this paper, without or with aggregation, is measured in terms of bandwidth-hop product, scaled by name resolution error rate. Specifically, give a content object whose name has the effective length of B bits, this name needs to be resolved first at the resolver with the DHT mechanism, which might be h hops away from the node that initiates the request. The bandwidth-hop product is thus defined as Bh. The effective length of a content object name is the cost to transmit a control packet carried the name. It is the actual name size plus certain packetization and transmission overhead. The bandwidth-hop product provides a coarse measure of the bandwidth overhead of the network for name resolution. Similarly, when new content objects are generated, there is an overhead for inserting the names of these content objects into the resolvers. We call it insertion overhead or insertion cost. We recognize that there is additional overhead when the resolver returns the resolved address. Also, when new names are inserted into the resolver, extra information is probably sent along to the resolver. However, we ignore this overhead since they are associated with addresses or information other than content names. How much of this information (e.g., how many bits per address) will be needed remains an open problem and may vary from one design to another. Consequently, we restrict ourselves to consider only content name incurred overhead. It should be straightforward to incorporate the extra information in the calculation by following the line presented in this paper.

III. Bandwidth Overhead of DHT

Suppose c_i has an effective name length of B_i bits. Define

$$p_{ij} = \text{prob}(j \text{ sends the request} | c_i \text{ is requested}) \quad (1)$$

and let h_{ij} be the number of hops from j to the resolver of c_i. The overhead associated with name resolution request in a unit time (call it resolution cost rate) is

$$c_{\text{res}} = \sum_{i=1}^{M} 2\lambda_i B_i \sum_{j=1}^{N} p_{ij} h_{ij}. \quad (2)$$

For ease of explanation, we assume the request and replay incurs the same bandwidth cost. c_{res} can be further written as

$$c_{\text{res}} = 2\sum_{i=1}^{M} B_i \sum_{j=1}^{N} q_{ij} h_{ij} = 2\sum_{i=1}^{M} B_i w_i \quad (3)$$

where we have defined q_{ij} as the rate at which j requests for c_i

$$q_{ij} = \lambda_i p_{ij} \quad (4)$$

and w_i the average total hops incurred by all possible resolutions for c_i from all the nodes

$$w_i = 2\sum_{j=1}^{N} q_{ij} h_{ij}. \quad (5)$$

Similarly we can define the insertion cost rate as

$$c_{\text{ins}} = \sum_{i=1}^{M} \alpha_i B_i \sum_{j=1}^{N} p'_{ij} h_{ij} \quad (6)$$

where

$$p'_{ij} = \text{prob}(j \text{ initiates the insertion } | c_i \text{ inserted}) \quad (7)$$

Define q'_{ij} as the rate at which j inserts c_i

$$q'_{ij} = \alpha_i p'_{ij} \quad (8)$$

and u_i as the average total hops due to inserting c_i

$$u_i = \sum_{j=1}^{N} q'_{ij} h_{ij}, \quad (9)$$

then we may write

$$c_{\text{ins}} = \sum_{i=1}^{M} B_i u_i. \quad (10)$$

The total resolution cost rate is thus given as

$$c_{\text{tot}} = \sum_{i=1}^{M} B_i (w_i + u_i). \quad (11)$$

(11) is the general formula to calculate the total cost rate with DHT, no matter fixed length naming or variable length

naming is used. In the following, we consider both scenarios in more details.

A. DHT Resolution Cost Rate with Fixed Name Length

We look at the special scenario where $B_i = B, \forall i$. According to (11), the total cost rate with fixed length naming is

$$c_{\text{tot,fix}} = B \sum_{i=1}^{M} (w_i + u_i). \tag{12}$$

We impose the natural constraint that these names must be distinct for the M contents that exist in the network so that faultless name resolution is possible. Clearly at least we need $\log_2 M$ bits for each name, in which case the cost is minimized as

$$c^*_{\text{tot,fix}} = \log_2 M \left(\sum_{i=1}^{M} (w_i + u_i) \right). \tag{13}$$

B. DHT Resolution Cost Rate with Variable Name Lengths

Relaxing the fixed length naming assumption, we may still require faultless name resolution. We can at the same time seek to minimize the total cost rate using variable-length names. This problem can be formulated as

$$\begin{aligned} &\text{minimize} && c_{\text{tot,var}} = \sum_{i=1}^{M} B_i (w_i + u_i) \\ &\text{subject to} && \text{distinct names} \\ &\text{variables} && B_i \in \text{Integers}, \quad \forall i. \end{aligned} \tag{14}$$

(14) can be solved by the entropy coding theory [13]. Specifically, define vector

$$\mathbf{v} = \frac{1}{\sum_{i=1}^{M}(w_i + u_i)} (w_1 + u_1, \dots, w_M + u_M) \tag{15}$$

which constitutes the probability mass function (PMF) over the M content objects. Let v_i be the ith component of $\mathbf{v}$ and $H(\mathbf{v})$ be the entropy of the PMF represented by $\mathbf{v}$, then

$$B_i^* = \lceil -\log_2 v_i \rceil \tag{16}$$

solves (14). For the purpose of this paper, the optimal B_i^* can be approximated by

$$B_i^* = -\log_2 v_i. \tag{17}$$

The approximation of $\lceil x \rceil \approx x$ will be repeatedly and implicitly employed in what follows. Consequently

$$c^*_{\text{tot,var}} = \left(\sum_{i=1}^{M} (w_i + u_i) \right) H(\mathbf{v}). \tag{18}$$

IV. Bandwidth Overhead of Flooding

The flooding mechanism is generally implemented in CCN-type network architectures in which content names usually have a hierarchical structure consisting of subnames like a encoded URL. Suppose each name consists of E subnames $x_1/x_2/\dots/x_E$. The subname x_i is considered as an element from a space X_i of m_i values. Thus we may consider each name comes from the space $X = \prod_{i=1}^{E} X_i$. As a result, all the names form a tree with each node in the i-th tier of the tree represents a specific value from X_i. The uniqueness of a name is guaranteed by identifying a unique path from the root node to a leaf node as the name. As flooding is commonly referred to as advertisement, we also call the bandwidth overhead with flooding the advertising cost.

Suppose c_i's subnames have length $b_{i1}, b_{i2}, \dots, b_{iE}$, then the advertising cost can be written as

$$c_{\text{adv}} = \sum_{i=1}^{M} \alpha_i L \sum_{k=1}^{E} b_{ik}, \tag{19}$$

where L is the total number of links in the network. (19) is true because each name must be flooded over the entire network, i.e., over L links.

A. Advertising Cost with Fixed Length Subnames

If we use fixed length subnames, the maximum number of bits we need is $\log_2 m_k$ for x_k, in which case the advertising cost is minimized as

$$c_{\text{adv,fix}} = L \left(\sum_{i=1}^{M} \alpha_i \right) \left(\sum_{k=1}^{E} \log_2 m_k \right) \tag{20}$$

Note it is necessarily true that

$$\prod_{k=1}^{E} m_k \geq M. \tag{21}$$

B. Advertising Cost with Variable-Length Names

If we further allow variable length subnames, we can also derive the best naming scheme with the smallest advertising cost by solving

$$\begin{aligned} &\text{minimize} && c_{\text{adv,var}} = \sum_{i=1}^{M} \alpha_i L \sum_{k=1}^{E} b_{ik} \\ &\text{subject to} && \text{distinct names} \\ &\text{variables} && b_{ik}, \quad \forall i, k. \end{aligned} \tag{22}$$

To derive the solution, rewrite the cost as

$$c_{\text{adv,var}} = L \sum_{k=1}^{E} \sum_{i=1}^{M} \alpha_i b_{ik}. \tag{23}$$

Because the subname x_k takes one of m_k possible values, we can partition $\{c_i\}_{i=1}^{M}$ into m_k groups, $G_{1k}, G_{2k}, \dots, G_{m_k,k}$, each group corresponding to a particular value from X_k with the kth subname lengths of $b'_{1k}, b'_{2k}, \dots, b'_{m_k,k}$ bits, respectively. The definition of G_{gk} is

$$G_{gk} = \{1 \leq i \leq M : k\text{th subname of } c_i \text{ takes the } g\text{th value}\}. \tag{24}$$

Therefore

$$c_{\text{adv,var}} = L \sum_{k=1}^{E} \sum_{g=1}^{m_k} \left(\sum_{i \in G_{gk}} \alpha_i \right) b'_{gk}. \tag{25}$$

Again, define $\mathbf{y}_k$ that represents a PMF over the m_k values

$$\mathbf{y}_k = \frac{1}{\sum_{i=1}^{M} \alpha_i} \left(\left(\sum_{j \in G_{1k}} \alpha_j \right), \left(\sum_{j \in G_{2k}} \alpha_j \right), \dots, \left(\sum_{j \in G_{m_k,k}} \alpha_j \right) \right). \tag{26}$$

Let $H(\mathbf{y}_k)$ be the entropy of $\mathbf{y}_k$ and y_{gk} be the gth component of $\mathbf{y}_k$. The first summation indicates that the objective is decomposable. We are allowed to optimize each subname indexed by k separately. According to the entropy coding theory, $c_{\text{adv,var}}$ is minimized if we set

$$b'^{*}_{gk} = \log_2 y_{gk}. \tag{27}$$

As a result

$$c^{*}_{\text{adv,var}} = L \left(\sum_{i=1}^{M} \alpha_i \right) \left(\sum_{k=1}^{E} H(\mathbf{y}_k) \right). \tag{28}$$

V. Bandwidth Overhead Comparison of DHT and Flooding Based Name Resolution

Fixed length naming can be regarded as a special case of the variable length naming with a rigid constraint. Therefore it usually does not achieve the lowest naming cost as the variable-length naming does. As variable length naming schemes shown in (18) and (28) represent the lowest cost schemes, we may compare DHT and flooding when both are tuned optimally and see under what network scenarios DHT is superior to flooding, or vice versa. The comparison also reveals the best use cases for either scheme. For DHT, we can rewrite (11) in a different form:

$$\begin{aligned} c^{*}_{\text{tot}} &= \left(\sum_{i=1}^{M} (w_i + u_i) \right) H(\mathbf{v}) \\ &= \left(\left(\sum_{i=1}^{M} w_i \right) + \left(\sum_{i=1}^{M} u_i \right) \right) H(\mathbf{v}) \\ &= \left(\left(\sum_{i=1}^{M} \sum_{j=1}^{N} 2\lambda_i p_{ij} h_{ij} \right) + \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \alpha_i p'_{ij} h_{ij} \right) \right) H(\mathbf{v}). \end{aligned} \tag{29}$$

Let λ^j be the rate at which j sends out requests and let

$$p^{ij} = \text{prob}(\text{request for } c_i | j \text{ sends a request}), \tag{30}$$

then

$$\lambda_i p_{ij} = \lambda^j p^{ij} \tag{31}$$

(31) is due to different factorization using Bayesian's law. Similarly we define α^j to be the rate at which j inserts new content and define

$$p'^{ij} = \text{prob}(\text{insert } c_i | j \text{ inserts}), \tag{32}$$

then

$$\alpha_i p'_{ij} = \alpha^j p'^{ij}. \tag{33}$$

Further, let $O_1, O_2, \dots, O_R$ be the node label of the R resolvers. Given two node labels n_1, n_2, let H_{n_1, n_2} be the distance measured in hops between node n_1 and node n_2. Then we have

$$\begin{aligned} \sum_{i=1}^{M} p^{ij} h_{ij} &= E[\text{hops to resolver} | j \text{ req for resolution}] \\ &= \sum_{r=1}^{R} \text{prob}(\text{use resolver } O_r | j \text{ req for resolution})\, h_{j,O_r} \\ &= \frac{1}{R} \sum_{r=1}^{R} h_{j,O_r} \\ &= h^j. \end{aligned} \tag{34}$$

Note the third equal sign follows from the assumption that any node j will send requests to the R resolvers with a uniform distribution. Similarly

$$\begin{aligned} \sum_{i=1}^{M} p'^{ij} h_{ij} &= E[\text{hops to resolver} | j \text{ inserts a name}] \\ &= \sum_{r=1}^{R} \text{prob (use resolver } O_r | j \text{ inserts a name}) h_{j,O_r} \\ &= \frac{1}{R} \sum_{r=1}^{R} h_{j,O_r} \\ &= h^j. \end{aligned} \tag{35}$$

Note the third equal sign follows from the assumption that any node j will insert in the R resolvers with a uniform distribution. Plug (31) and (33) into (29) and make use of (34) and (35), we get

$$\begin{aligned} c^{*}_{\text{tot,var}} &= \left(\left(\sum_{j=1}^{N} 2\lambda^j h^j \right) + \left(\sum_{j=1}^{N} \alpha^j h^j \right) \right) H(\mathbf{v}) \\ &= \left(\sum_{j=1}^{N} (2\lambda^j + \alpha^j) h^j \right) H(\mathbf{v}) \\ &= \left(\sum_{j=1}^{N} (2\lambda^j + \alpha^j) \right) \bar{h}_{\lambda+\alpha} H(\mathbf{v}). \end{aligned} \tag{36}$$

where we have defined

$$\bar{h}_{\lambda+\alpha} = \frac{\sum_{j=1}^{N} (2\lambda^j + \alpha^j) h^j}{\sum_{j=1}^{N} (2\lambda^j + \alpha^j)} \tag{37}$$

as the average distance from a node to a resolver, weighted by the sum of request rate and insertion rate. Let us assume that h^j is independent of the request rates and content generation rates at each node. This assumption would be true if we place a large number of resolvers randomly in the network, which is indeed the acceptable design choice to balance the load of resolution traffic. But under this assumption, we approximate $\bar{h}_{\lambda+\alpha}$ by its statistical mean $E[\bar{h}_{\lambda+\alpha}]$ over random variables $\lambda_j + \alpha_j$ and h^j

$$\begin{aligned}\bar{h}_{\lambda+\alpha} &\approx E[\bar{h}_{\lambda+\alpha}] \\ &= E\left[\frac{\sum_{j=1}^{N}(\lambda^j + \alpha^j)E[h^j|\lambda_j + \alpha_j]}{\sum_{j=1}^{N}(\lambda^j + \alpha^j)}\right] \\ &= E\left[\frac{\sum_{j=1}^{N}(\lambda^j + \alpha^j)E[h^j]}{\sum_{j=1}^{N}(\lambda^j + \alpha^j)}\right] \\ &= E[h^j].\end{aligned} \tag{38}$$

Because the resolvers are located randomly, $E[h^j] = \sum_{r=1}^{R} h_{j,o_r}/R$ can be considered as the average hop distance between a randomly chosen node and a sample of R randomly chosen nodes. Since R is assumed to be large, $E[h^j]$ should be close to the average distance between any two randomly picked nodes on the Internet, denoted as $\bar{h}$. We thus make this approximation and get

$$c^*_{\text{tot,var}} = \left(\sum_{j=1}^{N}(\lambda_j + \alpha_j)\right)\bar{h}H(\mathbf{v}). \tag{39}$$

We can also rewrite the advertising cost of the flooding scheme. Note

$$\sum_{i=1}^{M}\alpha_i = \sum_{j=1}^{N}\alpha^j, \tag{40}$$

which is true because the two sides of the equation are equal to the total new content generation rate of the network, we may write

$$c^*_{\text{adv,var}} = \left(\sum_{j=1}^{N}\alpha^j\right)L\left(\sum_{k=1}^{E}H(\mathbf{y}_k)\right). \tag{41}$$

From (36) and (41), we can see that DHT has a smaller factor of $\bar{h}$ as in $c^*_{\text{tot,var}}$ compared to the factor L in $c^*_{\text{adv,var}}$. Therefore in a richly connected network, the advertising cost may be prohibitive for flooding. On the other hand, DHT has a larger factor of $\sum_{j=1}^{N}(\lambda^j + \alpha^j)$ compared to $\sum_{j=1}^{N}\alpha^j$ of flooding. If a large number of content objects are requested over the network, DHT may be less cost effective. Unfortunately, future Internet features both rich connection and high content utility. Neither DHT nor flooding can be taken as the absolutely preferred approach. However, (36) and (41) enable us to derive the tradeoff between DHT and flooding. In general, we would choose DHT over flooding if

$$\frac{c_{\text{tot,var}}}{c_{\text{adv,var}}} < 1 \tag{42}$$

i.e.,

$$\left(\frac{\sum_{j=1}^{N}\lambda^j}{\sum_{j=1}^{N}\alpha^j} + 1\right)\frac{\bar{h}}{L}\frac{H(\mathbf{v})}{\sum_{k=1}^{E}H(\mathbf{y}_k)} < 1. \tag{43}$$

To make a first order comparison, let $\lambda = \sum_{j=1}^{N}\lambda^j$ denote the total content request rate of the network. Let $\alpha = \sum_{j=1}^{N}\alpha^j$ denote the total content generation rate. For flooding, we further assume that, due to name compression, different subnames take their values independently. Due to the independent subname assumption, when perfect variable length naming is used in flooding and DHT, we would have

$$H(\mathbf{v}) = \sum_{k=1}^{E}H(\mathbf{y}_k). \tag{44}$$

Then we would choose DHT if

$$\left(\frac{\lambda}{\alpha} + 1\right)\frac{H(\mathbf{v})}{\sum_{k=1}^{E}H(\mathbf{y}_k)} < \frac{L}{\bar{h}}. \tag{45}$$

Studies [14][15] of graph theory and Internet topology shows that for scale-free networks exhibiting power law degree distribution, we have

$$\bar{h} \approx \frac{\log N}{\log K} \tag{46}$$

where N is the total number of nodes and K the average degree of each node. For the Internet, $2 < K < 3$, but we do not have an accurate number for N, partly because N is ever growing. On the other hand, let K_j be the degree of node j. By Euler's theorem,

$$L = \frac{1}{2}\sum_{j=1}^{N}K_j = \frac{NK}{2}. \tag{47}$$

Therefore, we choose DHT if

$$\left(\frac{\lambda}{\alpha} + 1\right)\frac{H(\mathbf{v})}{\sum_{k=1}^{E}H(\mathbf{y}_k)} < \frac{NK\log K}{2\log N}. \tag{48}$$

We plot in Fig. 1 the right-hand side of (48) in terms of N, with $K = 2$ and $K = 3$. The lower region delineated by the curve represents the scenario where we would prefer DHT. The decision curve is largely linear in the log-log setting. As network grows in size, we become more in favor of DHT, unless λ/α grows at the same rate. For example, if there are 10^4 nodes in the network, we would choose CCN over DHT if the content request rate is at least 1000 times higher than the generation rate.

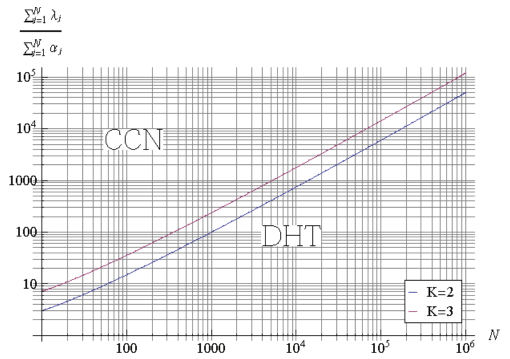

Fig. 1 The critical value of λ/α to choose DHT over CCN

VI. WHEN IS AGGREGATION PRFERRED IN DHT?

In this section, we answer the question in a DHT based network that, given a set of content names, should we aggregate them or not, and if yes, what the optimal aggregation would be. In Section VII, we answer the same question in a flooding based network. Aggregation is publishing technique by a CR transparent to users. Once we fix the naming scheme – fixed length or variable length – the cost due to request would no longer change regardless we use aggregation or not. Therefore in this section, our definition of cost does not include the cost due to request. However, as we use Bloom filter as the primary aggregation technique, a nonzero false positive probability is inevitable, which leads to a portion of the requests erroneously resolved. Before aggregation happens, a CR first identifies a collection of sets of content that it can possibly possess and that would be inserted into the same resolver, called aggregatable sets. Only these sets will be considered for aggregation. Let M' be the size of an aggregatable set and suppose a CR possesses $M < M'$ content files from this set. If the CR decides to aggregate the set, it generates a m-bit Bloom filter summarization of the M content objects and sends it to the resolver. The updating period is T seconds, i.e., every T seconds, a new decision needs to be made. Each content object has an effective name length of B bits. Without aggregation, cost associated with the M content objects would be

$$c_{\text{dht,noagg}} = (MB\bar{h} + \text{request cost for } M \text{ objects})/T \tag{49}$$

where $\bar{h}$ represents the average distance between the CR and the corresponding resolver. Suppose k Hash functions are used to generate the summarization for aggregation, there will be additional cost that is due to the false positive probability,

$$P_{\text{fp}} \approx \left(1 - e^{-\frac{kM}{m}}\right)^k. \tag{50}$$

Consequently, the cost consists of three parts, the insertion cost, request cost for the M contents and request cost due to false positive, which would not exists if it were not for the resolution error. Noting the average hop count between an end user and a resolver is $\bar{h}$, we have

$$\begin{aligned} c_{\text{dht,agg}} &= \frac{1}{T}\left(m\bar{h} + \text{request cost for } M \text{ contents} + B\left(\sum_{i=M+1}^{M'} \lambda_i\right) T \left(1 - e^{-\frac{kM}{m}}\right)^k \bar{h}\right) \\ &= \frac{1}{T}\left(m + \text{request cost for } M \text{ contents} + B\lambda_s T \left(1 - e^{-\frac{kM}{m}}\right)^k \bar{h}\right). \end{aligned} \tag{51}$$

Because (49) and (51) share three common terms – "request cost for M contents", $\bar{h}$ and T. We can drop these terms from both expressions and let

$$c_{\text{dht,nonagg}} = MB, \tag{52}$$

and

$$c_{\text{dht,agg}} = m + B\lambda_s T \left(1 - e^{-\frac{kM}{m}}\right)^k, \tag{53}$$

for the purpose of comparison, where we have defined for convenience

$$\lambda_s = \sum_{i=M+1}^{M'} \lambda_i, \tag{54}$$

which is the rate at which name resolution errors happen. Since the CR computes the Bloom filter summarization and updates it with the resolver, m is the insertion cost. The second term represents the cost of name resolution error due the intrinsic false positive behavior of the Bloom filter.

Both $c_{\text{dht,nonagg}}$ and $c_{\text{dht,agg}}$ are minimized if the smallest B is used. Since B needs to be big enough to distinguish M' contents, it is clear that

$$B_{\min} = \log_2 M'. \tag{55}$$

Now assume $B = B_{\min}$. For the aggregation scheme, we need to properly design the Bloom filter to minimize $c_{\text{dht,agg}}$, i.e., we should determine m, the the Bloom filter length and k, the number of hash functions used. Let $b = B\lambda_s T$. First we note if $b < MB$, then it is trivially true that aggregation is better since in this case the CR can send a 0 bit Bloom filter indicating the possession of all M' contents. This will lead to a false positive probability of 1 that incurs a cost of b. Thus we only consider the interesting case where $b > MB$. In this case, the optimal k that minimizes $c_{\text{dht,agg}}$ is given by

$$k^* = \frac{m}{M}\log 2, \tag{56}$$

which gives

$$c_{\text{dht,agg}} = m + b2^{-\frac{\log 2m}{M}}. \tag{57}$$

We further minimize $c_{\text{dht,agg}}$ by choosing the optimal filter length m:

$$m^* = \frac{M\log\frac{(\log 2b)^2}{M}}{(\log 2)^2}, \tag{58}$$

which leads to

$$c_{\text{dht,agg}} = \frac{M\log\frac{e(\log 2b)^2}{M}}{(\log 2)^2} \approx 2.081M\log\frac{1.306b}{M}. \tag{59}$$

Therefore we would choose not to aggregate if $c_{\text{dht,nonagg}} < c_{\text{dht,agg}}$, i.e.,

$$MB < 2.081M \log \frac{1.306b}{M}. \tag{60}$$

This condition can be rewritten as

$$2^{\frac{B}{2.081}} < \frac{1.306b}{M}. \tag{61}$$

Define

$$\rho = \frac{b}{MB} = \frac{\lambda_s T}{M}. \tag{62}$$

By our assumption, $\rho > 1$. We can rewrite the criterion (61) as

$$2^{\frac{B}{2.081}} < 1.306\rho B. \tag{63}$$

For a given ρ, there exists the smallest B that satisfies (59), meaning a smallest M that can possibly make non-aggregation a better choice.

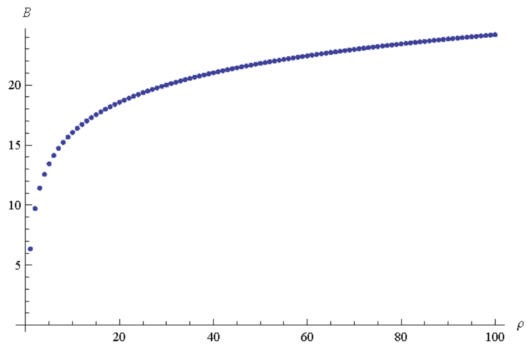

Fig. 2 The maximum name length B ($M' = 2^B$ is the number of content objects) to justify non-aggregation as a function of ρ.

Fig. 2 shows the decision curve. The horizontal axis is ρ. For each $\rho > 1$, the vertical axis gives the largest B that makes (63) hold true, which equivalently puts an upper bound on the number of content objects this network contains, i.e., $M' = 2^B$. As a matter of fact, (63) has an approximate solution

$$B_{\max} = 2.0814 \log_2(e\rho \log_2(e\rho)). \tag{64}$$

When ρ gets larger, it means a larger portion of requests cannot be served by the responsible CR. In other words, ρ measures the repository inefficiency of the CR by the ratio of unservable content requests per content object it holds. However, we expect that in a large content network, the content inefficiency is generally high. In this case, if non-aggregation is a better choice, we require M' not to exceed an upper bound, i.e.,

$$M'_{\max} = 2^{B_{\max}} = (e\rho \log_2(e\rho))^{2.0814} \tag{65}$$

Essentially $M'_{\max}$ increases roughly quadratically with ρ.

While this discussion solves the issue whether or not to aggregate. It does not tell which sets to aggregate. For example, Fig. 3 shows the contents that a repository owns as represented by yellow dots. The entire space of content objects shown in the red square are all the possible requests the resolvers may receive. The repository may choose either faithfully reporting each content name, or aggregating certain sets as delineated in white. Apparently, the small square region A at the lower left corner has a greater opportunity to be aggregated because content objects within that region not held by the repository are few. (63) can be used to make the aggregation decision. However, it is not clear whether it has more benefits to aggregate A and B together from (63).

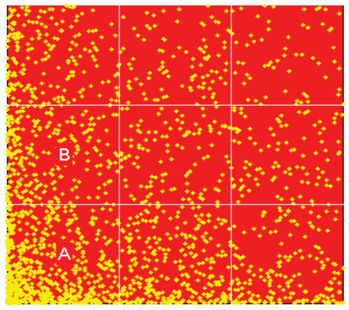

Fig. 3 Representing the contents on a repository as yellow dots. By evaluating (63) one can tell if aggregating contents in region A is better than non-aggregation.

VII. When is Aggregation Preferred in Flooding?

The analysis about when aggregation is preferred over non-aggregation in a flooding based network follows basically the same line with a few twitches. Because the flooding based network features a tree-structured name space, with each name consisting of consecutive subnames, there is a natural and conventional partition of the name space into aggregatable subtrees. Because each subtree is identified by its root, aggregation can be done partially with flooding as opposed to DHT. Specifically, suppose each name consists of E elements and a node is identified by the first J elements, then $E - J$ elements of content names that lie in the same subtree (i.e., sharing the first J elements in their names) can be aggregated. Moreover, a general approach to aggregate the $E - J$ elements would be to generate a Bloom filter summarization to each subname $x_{J+1}, x_{J+2}, \dots, x_E$ of lengths $B_{J+1}, B_{J+2}, \dots, B_E$, using Bloom filters of length $m_{J+1}, m_{J+2}, \dots, m_E$ bits, with $k_{J+1}, k_{J+2}, \dots, k_E$ hash functions. The CR then appends the $E - J$ summarizations after the J subnames to generate the complete entry for flooding. Assume there are M' contents that belong to the subtree and $M < M'$ of them belong to a particular CR, then the advertising cost for non-aggregation is

$$c_{\text{adv,nonagg}} = M \sum_{j=J+1}^{E} B_j L, \tag{66}$$

and for aggregation

$$c_{\text{adv,agg}} = \left(\sum_{j=J+1}^{E} m_j\right) L + \left(\sum_{j=J+1}^{E} B_j\right) \lambda_s T \prod_{j=J+1}^{E} \left(1 - e^{-\frac{k_j M}{m_j}}\right)^{k_j}. \tag{67}$$

We already know that the optimal $k_j, j = J+1, J+2, \ldots, E$ are given by

$$k_j^* = \frac{m_j}{M} \log 2, \tag{68}$$

therefore, the optimal aggregation would imply

$$\begin{aligned} c_{\text{adv,agg}} &= \left(\sum_{j=J+1}^{E} m_j \right) L \\ &+ \left(\sum_{j=J+1}^{E} B_j \right) \lambda_s T 2^{-\frac{\left(\sum_{j=J+1}^{E} m_j\right) \log 2}{M}} \bar{h}. \end{aligned} \tag{69}$$

Let

$$\begin{aligned} m &= \sum_{j=J+1}^{E} m_j , \\ B &= \sum_{j=J+1}^{E} B_j , \\ b &= \frac{B \lambda_s T \bar{h}}{L}, \end{aligned} \tag{70}$$

then we have

$$c_{\text{adv,nonagg}}/L = MB, \tag{71}$$

and

$$c_{\text{adv,agg}}/L = m + b 2^{-\frac{\log 2m}{M}}. \tag{72}$$

Note (71), (72) are exactly the same with (52), (57). Therefore the same conclusion follows -- aggregation is preferred over non-aggregation if and only if

$$B \geq \min\left\{ \frac{b}{M}, 2.801 \log_2(e\rho \log_2(e\rho)) \right\}, \tag{73}$$

where $\rho = b/MB$. Like Section VI, our discussion merely answers the question whether to aggregate a given subtree. It is still an open question that, given multiple subtrees (not necessarily disjoint), which should be aggregated in order to minimize the cost.

VIII. Conclusion

In this paper, we investigate the implications of name resolution overhead in DHT and in flooding based information centric networks, two competitive candidates for the future Internet architectures. The main difference between the two architectures lies in the way of naming content objects on the Internet. While DHT assigns flat names to content objects, flooding uses a hierarchical naming scheme. This difference entails different design choices in routing and routing table population. Our paper specifically discussed some costs associated with these differences. We discussed in both architectures the name resolution overhead with fixed length and variable length names, when aggregation is not used. From this discussion, we made a baseline comparison of the architectures and identified the situations where one architecture is preferred over the other. We then discussed the name aggregation with the assistance of Bloom filters in either architecture separately. We presented the criterion that determines when aggregating a set of content objects incurs lower cost than non-aggregation, and when aggregation is preferred, what the optimal aggregation scheme would be. In all subjects covered in this paper, we strived to make a pronounced comparison of DHT and flooding side by side. We believe our preliminary work can serve as a first order guideline for ICNs. We also hope that this work may inspire interest from network researchers to look deeper into the issue of quantitatively modeling future Internet architectures.

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in Proceedings of 5th Internet Conference on Emerging Networking Experiments and Technologies, ser. CoNEXT’09. ACM, 2009, pp. 1–12.

[2] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, “A data-oriented (and beyond) network architecture,” in Conference on Applications, technologies, architectures, and protocols for computer communications, ser. SIGCOMM’07. ACM, 2007, pp. 181–192.

[3] D. R. Cheriton and M. Gritter, “TRIAD: A scalable deployable NAT-based Internet architecture,” Tech. Rep., 2000. [Online]. Available: http://www-dsg.stanford.edu/triad/

[4] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network,” in Proceedings of ACM SIGCOMM, San Diego, CA, August 2001, pp. 161–172.

[5] A. Rowstron and P. Druschel, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems,” in Proceedings of 18th IFIP/ACM International Conference on Distributed Systems Platforms, ser. Middleware’01, November 2001, pp. 329–350.

[6] J. Saia, A. Fiat, S. Gribble, A. Karlin, and S. Saroiu, “Dynamically faulttolerant content addressable networks,” in Proceedings of 1st International Workshop on Peer-to-Peer Systems, ser. IPTPS’01, March 2002, pp. 270–279.

[7] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in Proceedings of ACM SIGCOMM, San Diego, CA, 2001, pp. 149–160.

[8] S. C. Nelson, G. Bhanage, D. Raychaudhuri, “GSTAR: generalized storage-aware routing for MobilityFirst in the future mobile Internet,” in Proceedings of 6th international workshop on MobiArch, pp. 19-24, Bethesda, MD, 2011.

[9] S. Oueslati, J. Roberts, and N. Sbihi, “Flow-aware traffic control for a content-centric network,” in Proceedings of IEEE Infocom, Orlando, FL, March 2012, pp. 2417–2425.

[10] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino, “Modeling data transfer in content-centric networking,” in Proceedings of 23rd International Teletraffic Congress, ser. ITC’11, 2011, pp. 111–118.

[11] B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Communications of the ACM, vol. 13, no. 7, pp. 422-426, July 1970.

[12] H. Liu, D. Zhang, “A TLV-structured data naming scheme for content-oriented networking,” 5th International Workshop on the Network of the Future with IEEE ICC (FutureNet V), Ottawa, Canada, June 2012.

[13] T. M. Cover, J. A. Thomas, “Elements of information theory,” Wiley-Interscience, 1991.

[14] R. Albert, A.-L. Barabási, “Statistical mechanics of complex networks,” Rev. Mod. Phys., vol. 74, no. 1, pp. 47-97, January, 2002.

[15] A. Fronczak, P. Fronczak, J. A. Hołyst, “Average path length in random networks,” Phys. Rev. E, vol. 70, no. 5, pp. 056110, November, 2004.