αRoute: A Name Based Routing Scheme for Information Centric Networks

Reaz Ahmed, Md. Faizul Bari, Shihabur Rahman Chowdhury, Md. Golam Rabbani, Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo
`[r5ahmed | mfbari | sr2chowdhury | m6rabban | rboutaba]@uwaterloo.ca`

Bertrand Mathieu
Orange Labs, Lannion, France
`bertrand2.mathieu@orange-ftgroup.com`

(Technical Report: CS-2012-16)

Abstract

One of the crucial building blocks for Information Centric Networking (ICN) is a name based routing scheme that can route directly on content names instead of IP addresses. However, moving the address space from IP addresses to content names brings scalability issues to a whole new level, due to two reasons. First, name aggregation is not as trivial a task as the IP address aggregation in BGP routing. Second, the number of addressable contents in the Internet is several orders of magnitude higher than the number of IP addresses. With the current size of the Internet, name based, anycast routing is very challenging specially when routing efficiency is of prime importance. We propose a novel name-based routing scheme (αRoute) for ICN that offers efficient bandwidth usage, guaranteed content lookup and scalable routing table size. αRoute consists of two components: an alphanumeric Distributed Hash Table (DHT) and an overlay to underlay (Internet topology) mapping algorithm. Simulation results show that αRoute performs significantly better than Content Centric Network (CCN) [1] in terms of network bandwidth usage, lookup latency and load balancing.

1 Introduction

Information Centric Networking (ICN) has recently received significant attention in the research community. ICN philosophy prioritizes a content ("what") over its location ("where"). To realize this separation of a content from its location, a name based routing mechanism is essential. However, a number of crucial issues and challenges related to name based routing are yet to be addressed in order to successfully realize a content oriented networking model for the future Internet.

Today's Internet exists as an interconnection of thousands of Autonomous Systems (ASs) from around the globe. The biggest Internet routing table contains around 4×10^5 Border Gateway Protocol (BGP) [2] routes for covering about 3.8×10^9 IPv4 addresses and 6×10^8 hosts . This 10^4 scaling factor between IPv4 addresses and BGP routes is achieved by prefix based routing and route aggregation. However, the number of addressable ICN contents is expected to be several orders of magnitude higher. Number of "data pieces" including sensor data, etc., is approximately on the order of 10^{16} [3], while Google has indexed approximately 10^{12} URLs [4], which would impose 7 orders of magnitude scalability requirement on a routing scheme similar to BGP.

The routing scalability issue in ICN is related to how contents are named and how inter-AS and intra-AS routing protocols process these names. Even if a BGP like inter-AS ICN routing protocol covers only the top-level domains as prefixes, it will need to carry approximately 2×10^8 unique prefix routes [5], as no aggregation is possible at this level. So, the crux of the problem lies in the fact that ICN requires Internet routers to maintain a Brobdingnagian amount of routing state, which does not seem to be possible with existing technology. However, in reality the scalability requirement will be much higher for the following reasons: (i) content names are not as aggregatable as IP addresses, (ii) names with same prefixes may not be advertised from nearby network locations, (iii) routing cannot depend on topological prefix binding as content retrieval should be location independent, (iv) restricting the content name to some form of specialized format limits the usability of the system, and finally, (v) supporting content replication, caching, and mobility reduces the degree of route aggregation that can be applied, as multiple routes for the same content need to be maintained in the routing table.

In this paper we address the routing scalability issue for ICN. This paper has three major contributions. First, we identify the key requirements for designing an ICN routing mechanism (Section 2). Second, we propose a name-based overlay routing scheme named αRoute, which is scalable and offers content lookup guarantee (Section 3). Both the routing table size and the number of hops for content lookup in αRoute are logarithmically bounded by network size. Third, for Internet inter-domain routing using αRoute, we propose a distributed overlay-to-underlay mapping scheme that enables near shortest path routing in underlay (AS-network) by preserving the adjacency relations in the overlay graph (Section 4).

We also provide mathematical bounds on the routing scalability of αRoute and the operating range of any mapping scheme (Section 5). We provide a brief review of the existing approaches for ICN routing in Section 6. Finally, we conclude and present future research directions in Section 7.

2 Design Considerations

We identify the following key requirements for an inter-domain ICN routing mechanism: 1) *name-based, any-cast routing* : directly forward a query to a valid copy of the requested content, unlike the contemporary, two-step process of URL-to-IP lookup followed by IP-to-host routing; 2) *bandwidth efficiency* : routing should follow the shortest possible path adhering to AS level policy agreements and should be able to accommodate with large number of ASs; 3) *lookup guarantee* : ensure discovery of an existing content regardless of its popularity and conserve bandwidth spent in non-existing content lookups; 4) *index placement freedom* : ASs should have the freedom of choosing the content they will be indexing; 5) *in-network caching* : content should be cached in routers and served from a copy, closest to a requester; 6) *routing scalability* : routing table size should grow sub-linearly with network size.

The design implications to satisfy these requirements may conflict with each other. For example, index placement freedom can be achieved by allowing any content to be placed anywhere, which mandates the routing table to grow linearly with content population in order to provide content lookup guarantee and bandwidth efficiency. Thus, it is difficult to develop a routing scheme that simultaneously satisfies all of these requirements. Instead, our aim is to develop a routing scheme that satisfies each of these performance requirements as closely as possible.

In a DHT-based structured overlay network, we can achieve all of the requirements for ICN routing except index placement freedom and one-step (or direct) name-based content lookup. Any DHT algorithm assigns unique identifier to each node (network location) and places a content's index at a designated indexing node. This guarantees content lookup within a limited number of overlay hops and a small routing table size, both logarithmically bounded by the network size. As a result a DHT cannot offer index placement freedom. In addition, a DHT requires two-step routing for content access: first, route to an indexing node to obtain content's index and then, route to the content's location. By adopting proper caching strategies we can achieve index placement freedom and one-step content routing to some extent. We explain these strategies in Section 4.4.

The most difficult part of using a DHT for inter-domain routing is to map the overlay graph to the AS-topology while preserving adjacency relations. This mapping problem is know to be NP-Complete [6, 7]. In the following two sections, we first provide a name-based DHT mechanism and then present our mapping scheme.

3 αRoute: A Name-based DHT

A DHT essentially maps a key to a value in a distributed manner. A DHT design involves two steps: a) *partitioning* : segregating the entire key-space into partitions and assign each partition to a node and b) *routing*: a mechanism

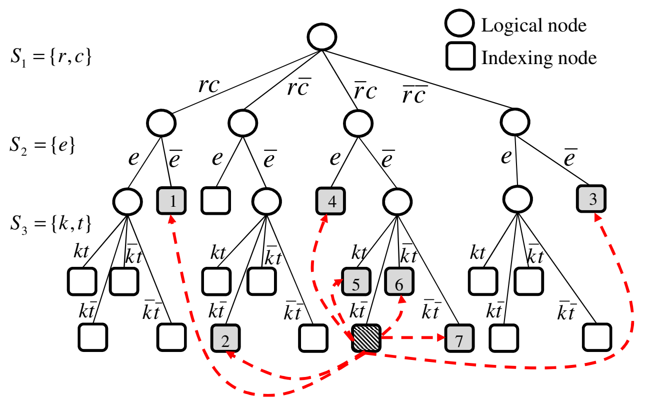

Figure 1: An example partitioning tree

for locating any key in a bounded number of hops.

3.1 Partitioning

Our partitioning policy has three desirable properties: first, it places similar names in same partition, second, it provides control over the number of partitions and third, it creates non-overlapping partitions.

We explain the partitioning process with the example presented in Fig. 1. First, we create some partitioning sets (S_is) using the 36 alphanumeric English characters (lower case alphabet and digits). For example, in Fig. 1, we have three partitioning sets: $S_1 = \{r, c\}$, $S_2 = \{e\}$ and $S_3 = \{k, t\}$, respectively. We can classify any string to one of the $2^{|S_i|}$ partitions based on the presence or absence of the characters in S_i. Here, $|S_i|$ represents the cardinality of S_i. Continuing the example in Fig. 1, we can create four partitions ($2^{|S_1|} = 4$) based on S_1, namely, rc, $r\bar{c}$, $\bar{r}c$ and $\bar{r}\bar{c}$. All strings having both r and c are classified to the rc partition. Strings that contain r but do not contain c are classified to partition $r\bar{c}$, and so on. Now we construct a logical partitioning tree where each node corresponds to a partition. The root node classifies all strings in a single partition. Nodes at level 1 correspond to the $2^{|S_1|}$ partitions based on S_1. In general, a node at level i corresponds to the strings containing (or not containing) the characters along the path from that node to the root. For example, the shaded node in Fig. 1 classifies the strings corresponding to $\bar{r}c - \bar{e} - k\bar{t}$, i.e., all the strings containing characters c and k, and not containing characters r, e and t.

Now we consider the logic of classifying a string, say "www.rocket.com", using the partitioning tree in Fig. 1. First, we ignore the character order and multiple occurrences to get an unordered character set $\{w, r, o, c, k, e, t, c, m\}$. Since, this set contains characters r, c, e, k and t, the string will be classified to the left-most leaf node $rc - e - kt$ in Fig.1.

We generate S_is in a three step process: (i) a large corpus of content names are parsed to compute character frequencies, (ii) all possible combinations of characters upto a fixed length (4 in our experiments) are generated, and (iii) the combination that produces the most balanced branching is selected. For example, let us assume that the tree is grown upto height 2 with the characters r, c, and e. Now, when we want to extend the tree height, we first compute the character frequency of the content names that are partitioned under the 8 nodes at height 2. Then we generate all possible character combinations (leaving out r, c, and e) of length at most 4 and select the combination that produces the most balanced branching at height 3, which is $\{k, t\}$ in Fig. 1.

S_is influence the distribution of published names over the partitions and the indexing load across the network, since we assign each leaf node (and selective internal nodes) to ASs. We ensure proper load balancing while constructing S_is. If we use a fairly large corpus, then the character frequency distribution and the resulting S_is are highly unlikely to change over time. Alternatively, we can control the load at a leaf node by adjusting its height, since the proportion of strings classified by a leaf node holds inverse relation with its height.

Non-English characters in a string may be handled in two alternative ways. First, if we limit ourselves to the 36 English characters, then non-English characters can be mapped to English characters using some predefined rules. Alternatively, we can incorporate non-English characters in the partitioning process, which may increase routing overhead. It is worth noting that we can accommodate around 64 billion nodes using a partitioning tree of 36 alpha-numeric characters.

3.2 Routing

Our routing mechanism has two components: routing table and message forwarding mechanism. Ideally the routing table should be logarithmic on network size, while the forwarding mechanism should ensure shortest path routing using local information only. In this section we present an overlay routing architecture which achieves both of these goals. In the next section we will present a mapping algorithm to achieve these goals in an underlay network as closely as possible.

Routing table: Each partition in the aforementioned partitioning tree can be identified by a *pattern* $s_1 s_2 \ldots s_h$ where, s_i is a character presence combination over the characters of S_i and h is the height of the partitioning tree. For example if $S_i = \{r, c\}$ then s_i can be any of rc, $r\bar{c}$, $\bar{r}c$ or $\bar{r}\bar{c}$. We define a *prefix* of a *pattern* as a leftmost sub-pattern of any length, e.g., $s_1 s_2 \ldots s_p$, where $p \leq h$.

Now we describe the routing table entries for the node responsible for partition $s_1 s_2 \ldots s_i \ldots s_p$. For some level i, a node's routing table will have $2^{|S_i|} - 1$ routing links corresponding to the partitions $s_1 s_2 \ldots t_i \ldots s_p$, where t_i is a character presence combination over the characters of S_i and $t_i \neq s_i$. In general, the routing table at a node will have $\sum_{i=1}^{p}(2^{|S_i|} - 1)$ entries.

We can better describe the routing table entries with an example. Consider the shaded node in Fig. 1 with prefix $\bar{r}c - \bar{e} - k\bar{t}$. This node will have a total of $7 (= (2^2 - 1) + (2^1 - 1) + (2^2 - 1))$ routing links to the nodes marked with numbers 1 to 7 in Fig. 1. The first three routing links are computed by taking the character presence combination over the characters in $S_1 = \{r, c\}$, which gives us $rc - \bar{e} - k\bar{t}$, $r\bar{c} - \bar{e} - k\bar{t}$ and $\bar{r}\bar{c} - \bar{e} - k\bar{t}$. Note that for the first and the third links, the tree has not been fully expanded to level 3, so the links will be pointing to nodes $rc - \bar{e}$ and $\bar{r}\bar{c} - \bar{e}$, respectively. For computing link 4, prefix characters corresponding to S_1 and S_3 will remain unchanged, while the character(s) in S_2 will be complemented and so on. Slightly different situation can arise if a node has a shorter prefix than other nodes, *e.g.*, the node with prefix $rc - \bar{e}$; its first routing entry would be $r\bar{c} - \bar{e}$, which is an internal node. Here any node with prefix $r\bar{c} - \bar{e}$ can be considered as the first link for $rc - \bar{e}$.

Message forwarding : We can define a simple message forwarding mechanism based on the above described routing table. A lookup string is converted to a set of characters corresponding to the partitioning set, S_i. The lookup request will be forwarded to the node responsible for the queried characters in a multi-hop path. This path is obtained by gradually transforming the prefix of the current node to the lookup pattern. Following the previous example in Fig 1, suppose the dark shaded node is looking for the node responsible for string “www.rectangle.ca”, which is mapped to the node with prefix $rc - e - \bar{k}t$. At the first step, node $\bar{r}c - \bar{e} - k\bar{t}$ will forward the query to node $rc - \bar{e}$ using routing link 1. The 2^{nd} routing link of $rc - \bar{e}$ will have the prefix of any node, say $rc - e - kt$, under node $rc - e$. Thus the query will be forwarded to $rc - e - kt$, which will finally forward the query to $rc - e - \bar{k}t$.

It is worth noting that the partition tree, as in the example of Fig. 1, does not exist in terms of network links because only the leaf nodes and a few selective internal node (explained in the next section) are assigned to ASs. Rather, the tree exists logically at each indexing ASs as prefix strings to the root. The overlay network is composed of the routing links (dashed lines in Fig. 1) between the indexing nodes.

3.3 Join protocol

To join the network a new node (i.e., AS), say X, has to know an existing AS in the system, say M. X will query M for the neighbor, say M_1, with shortest prefix. Next, X will query M_1 for the neighbor with shortest prefix. In this way X will crawl the network and find a local minima, i.e., a node with shorter prefix than all of its neighbors. In case of a tie, X will choose the node storing higher number of index records. Once the local minima, say Y is found, X will request Y to increase its prefix by one step. If prefix of Y is $s_1 s_2 \ldots s_p$ and Y has $2^{|S_p|}$ siblings in level p then Y will increase its prefix to $s_1 s_2 \ldots s_{p+1}$ and X will become a new sibling of Y, otherwise X will become a sibling of Y at level p. Accordingly, X has to populate its routing table using the routing information at Y. It can be trivially proven that all the neighbors of X will be within 2 hops from Y.

4 From Overlay to Underlay

To perform inter-domain routing, we have to map the nodes in αRoute overlay graph to the AS topology. In this section we first explain the impact of Internet topology on the mapping process (Section 4.1), then we present the mapping algorithm (Section 4.2) followed by the lookup (Section 4.3) and caching (Section 4.4) mechanisms.

4.1 Topology Considerations for Mapping

The inter-domain AS network is based on BGP, while each AS controls its intra-domain routing protocol independently. Hence, it will be inappropriate to use αRoute for both inter- and intra-domain routing in the future information centric Internet. Instead, following the current tradition, we assume that ASs will collaborate using αRoute for inter-domain routing, while for intra-domain routing an AS may extend its own αRoute prefix or it may use a separate routing protocol.

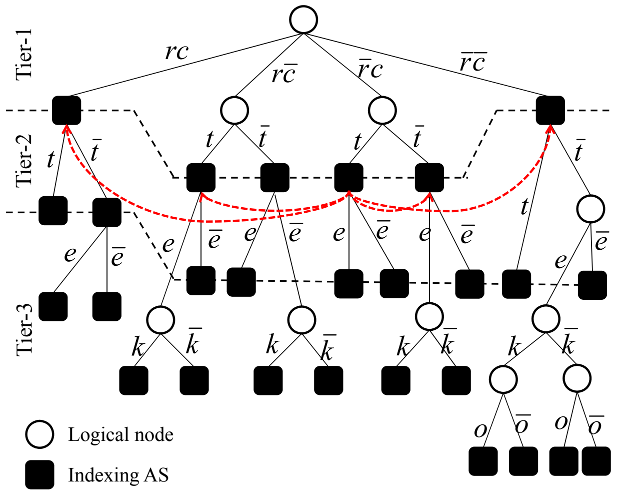

Figure 2: Mapping a partitioning tree to AS-topology

Node degree distribution at the overlay (αRoute) and underlay (AS-topology) graphs has profound impact on the mapping process. According to Fig. 1, each indexing node of a uniformly grown partitioning tree should have similar number of routing links. In other words, the overlay graph is a nearly regular graph. On the contrary, it has been reported in [8] that the node degree distribution in the AS-topology exhibits a power law relationship with the number of ASs. We exploit this dissimilarity in node degree distribution during the mapping process.

Algorithm 1 PERFORMNEIGHBORMAPPING(ξ, ρ)

```
Require: neighbor set $\xi$, and own prefix $\rho$.
Ensure: neighbors in $\xi$ are mapped to an extension of $\rho$
1: $S_{patterns} \leftarrow$ set of all unmapped patterns starting with $\rho$
2: while There are unmapped neighbors do
3:     $nbr \leftarrow$ neighbor with the highest number of mapped
           neighbors from $\xi$
4:     $nbr.pattern \leftarrow$ select a pattern from $S_{patterns}$ that
           minimizes the Hamming distance between $nbr$
           and its neighbors
5:     Remove $nbr.pattern$ from $S_{patterns}$
6:     Mark $nbr$ as mapped
7: end while
```

Recent studies [9] on Internet topology revealed that a small number (around 12 to 16) of high-degree ASs form an almost completely connected core. The rest of the ASs have multiple physical links to the core, which results into many triangles in the AS graph. It is also reported [10] that the inter-connect graph between the non-core ASs is sparse. For the mapping process, we treat the core ASs as Tier-1 AS, while the ASs directly connected to at least one Tier-1 AS are treated as Tier-2 ASs and so on. Hence, a Tier-2 AS, directly connected to multiple Tier-1 ASs, can route a lookup request to a core AS with an appropriate prefix, or it may use its peering links with other Tier-2 or lower tier ASs. This process recurs for the lower tier ASs as well.

Fig. 2 depicts a conceptual overview of αRoute prefix distribution over the ASs. To exploit the heterogeneous inter-AS connectivity, we assign short prefixes to the highly connected top tier ASs. A lower tier AS, on the other hand, extends a prefix of an upper tier AS. In contrast to the partitioning tree introduced in Fig. 1, selected logical nodes (partitions) at different levels are assigned to highly connected upper tier ASs. In addition to having the regular αRoute links (as presented by dashed arrows in Fig. 2), an upper tier AS will have physical links to the lower tier ASs that extend its prefix.

4.2 Mapping Algorithm

The mapping procedure is initiated by a centralized entity referred to as the Name Assignment Authority (NAA). The NAA chooses a set of prefixes and assigns them to the Tier-1 ASs. The prefixes are selected in such a way that the expected name resolution related processing load on each Tier-1 AS is distributed proportionally to its capacity. In the next step, each Tier-1 AS executes Algorithm 1 to assign prefixes to Tier-2 ASs. Each Tier-1 AS extends its own prefix to generate a set of patterns $S_{patterns}$ that are not yet mapped and starts with the same pattern as its own, *e.g.*, if a Tier-1 AS is assigned prefix $r\bar{c}$ then its $S_{patterns}$ set contains all unmapped patterns starting with $r\bar{c}$. Next, the AS finds a neighbor (*nbr*) that has the highest number of mapped neighbors. *nbr* is then assigned a prefix in such a way that its Hamming distance from all its neighbors is minimized and the process goes on until all neighbors are mapped. After the Tier-1 ASs have executed this mapping process, the already mapped Tier-2 ASs map their neighbors using the same Algorithm. The process goes on in a recursive manner until each AS is mapped.

In terms of mapping an edge (or logical link) in overlay graph to the underlay network, this mapping strategy can produce three scenarios : *equal*, *compression* and *expansion*. In most of the cases, a logical link will be mapped to a physical link, resulting into an *equal* or one-to-one mapping. Recall that, if two logical nodes in the overlay space have more than one mismatch (Hamming distance) between their prefixes, it results in a multi hop path between them in the overlay routing space. Therefore, when two physical neighbors have more than one mismatch in their assigned prefixes, a logical path between them in the overlay space is mapped with the physical link between them. In this case, traversing the physical link is equivalent to a jump in the overlay space while routing. This will essentially results into a *compression* of an overlay path into a physical link. Finally for a few cases, adjacent overlay nodes will be more than one hop away in the underlay, which will degrade the mapping performance due to the *expansion* of a logical link to a physical path in underlay AS-topology. In the experimental results section, we will provide quantitative measures for these three cases.

4.3 Content lookup

To lookup a content, we first create a pattern depending on the presence or absence of the letters in the given name (or keywords) matching the partitioning strings (S_is). Then we can use αRoute to route the lookup request to the AS indexing the names matching this pattern. At the indexing AS, we will find one or more index records of the form $< N_l, P_l >$, which indicates that the content with name N_l is stored at AS with pattern P_l. Now we use αRoute to reach the AS responsible for pattern P_l.

Each AS has to maintain a routing table (as explained in Section 4.1) for routing messages to an AS responsible for any given pattern. For each logical routing link L_k (corresponding to the dashed lines in Fig 1 and Fig 2), the routing table will contain an entry like $< L_k, I_k, h_k >$. Here, I_k is the inter-AS link that should be used for routing to the AS responsible for pattern L_k and h_k is the number of ASs to be traversed for reaching L_k. With a good mapping algorithm, h_k will be 1 for most of the cases. In addition to the logical links, an AS will keep separate routing entries like $< P_k, I_k, 1 >$ for each physical neighbor. Here, P_k is the pattern of the neighbor AS reachable through the inter-AS link I_k.

Similar to BGP, αRoute supports policy-based routing. αRoute can be augmented with different policies during the route selection process. In the current implementation we adhere to the following policy. If a lookup request can be resolved using a peering link (usually free of cost) we route using that link. Otherwise, the request has to be forwarded to a provider AS, which usually incurs cost to the requesting AS.

4.4 Indexing and Caching

Strict index placement restriction is a major disadvantage for any DHT approach. To enable efficient content lookup we have to place a content's index at a specific network location. It also results into two step routing: first, route to an index and then route to the content. We can mitigate both of these problems (i.e., index placement freedom and two-step lookup) by intelligently caching indexes and contents. Moreover, such caching policies will reduce expensive inter-AS traffic.

Index caching : ASs may not agree to store any content's index for several reasons, including legal implications and high query traffic for a popular content. If the content is illegal or access restricted then this behaviour of an AS is appropriate. But, for a popular content, such behaviour can decrease the content's reachability. To minimize the volume of lookup traffic for a popular content, each AS can cache the indexes returned by outgoing lookup requests for resolving future lookup requests. This index caching strategy will effectively reduce the popular content lookup traffic at the rendezvous indexing AS.

Content caching: As previously reported in [11] and [12], content popularity in the Internet and hence lookup rate follows power law distribution. We can use this property to improve response time by caching popular contents at the AS storing the content's index. This will allow us to access a content in one DHT lookup. However, we may face two barriers while deploying this strategy. First, a content owner may not allow the indexing AS to cache and serve its content due to financial, privacy or legal reasons. Second, an AS may be overloaded if the distribution of popular contents over the ASs is not uniform. The second obstacle can be reduced if ASs cache a popular, permitted content and update content's index by adding a link to the cached copy. In the later strategy, a user can lookup the indexing AS to find a list of ASs caching the desired content and access the content from a nearby AS. This approach can be more effective while accessing large contents.

5 Analysis

5.1 Bounds on αRoute

Routing table size (RT_{size}) and the number of hops required to reach a target node in the overlay from any given node are the most important performance measures for αRoute. Here we define loose upper bounds for both of these quantities. Consider a network of N routing nodes. Assume that there are ℓ partitioning sets in the overlay graph, and the minimum and maximum sizes for the partitioning sets are b_1 and b_2, respectively. Evidently, $b_1^{\ell-1} \leq N$ and $\ell \leq \frac{\log N}{\log b_1} + 1$. Hence, we can compute an upper-bound on RT_{size} as follows:

$$RT_{size} \leq \sum_{i=1}^{\ell} (2^{|S_i|} - 1) \leq \ell(2^{b_2} - 1) \leq \Big(\frac{\log N}{\log b_1} + 1\Big)(2^{b_2} - 1)$$

Now we find the bound for hop-distance between any pair of nodes in the overlay. We can observe that at each routing hop the search space is reduced by $\frac{1}{|S_i|}$ and the route converges when there is only one node in the search-space. Hence, we can bound the routing hops, say g, as follows:

$$\frac{N}{\prod_{i=1}^{g} 2^{|S_i|}} = 1$$

$$\Rightarrow \frac{1}{b_2} \log_2 N \leq g \leq \frac{1}{b_1} \log_2 N$$

From the above inequalities we claim that both RT_{size} and routing hops (g) in αRoute are logarithmically bounded by network size N. In addition, we can trade off RT_{size} for routing hops (g); increasing the cardinality of partitioning sets ($|S_i|$) will result in a shorter, fatter tree with shorter overlay paths but increased RT_{size}.

5.2 Bounds on Mapping

The effectiveness of our mapping strategy is inversely proportional to the number of path expansions, i.e., the cases where an overlay edge is mapped to a multi-hop underlay path. We use *expansion ratio* (η) to measure this quantity. Suppose a mapping algorithm $\mathscr{M}$ maps an overlay path P_{ov} to an underlay path $\mathscr{M}(P_{ov})$, where $\mathscr{M}(P_{ov})$ is the concatenation of the underlay paths corresponding to the edges along P_{ov}. Let $|P|$ denote the length of path P. Hence, we can define $\eta_{\mathscr{M}}$ as follows:

$$\eta_{\mathscr{M}} = E\Big[\frac{|\mathscr{M}(P_{ov})|}{|P_{ov}|}\Big] \tag{1}$$

Lemma 1 *The expansion ratio ($\eta_{\mathscr{M}}$) of a mapping strategy, say $\mathscr{M}$, that tries to preserve the adjacency relations in overlay graph will not be greater than the expansion ratio ($\eta_{\mathscr{R}}$) of a mapping strategy, say $\mathscr{R}$, that maps overlay nodes to randomly chosen underlay nodes without considering the adjacency relations in overlay graph.*

Proof 1 *Suppose, $\mathscr{R}$ maps the overlay nodes x, y and z to underlay nodes x', y' and z', respectively, such that the followings hold: (a) x and y are neighbours, (b) x' and z' are neighbours, and (c) none of the adjacency relations are preserved for y and z. If $\mathscr{M}$ takes the adjacency relation into consideration, and remaps y to z' and z to y', then path expansion will decrease for one edge (as $x \rightarrow y$ mapped to $x' \rightarrow z'$) and $\eta_{\mathscr{M}}$ will become smaller than $\eta_{\mathscr{R}}$. By repeating this process, mapping $\mathscr{M}$ can reduce $\eta_{\mathscr{M}}$ for every instance of x, y and z. If no such x, y and z exist then $\eta_{\mathscr{M}}$ will remain equal to $\eta_{\mathscr{R}}$.*

From Lemma 1, we can say that $\eta_{\mathscr{R}}$ will serve as an upper bound for the expansion ratio ($\eta_{\mathscr{M}}$) of our mapping algorithm, since our mapping is based on overlay adjacency relations. Evidently, for $\mathscr{R}$, the mapped path length $|\mathscr{R}(P_{ov})|$ does not depend on overlay path length $|P_{ov}|$. Hence, we can write:

$$\eta_{\mathscr{R}} = \frac{E\Big[|\mathscr{R}(P_{ov})|\Big]}{E\Big[|P_{ov}|\Big]} \tag{2}$$

Now, we compute $E\Big[|\mathscr{R}(P_{ov})|\Big]$, for the random mapping strategy $\mathscr{R}$. Since, $\mathscr{R}$ does not try to preserve the adjacency relations in overlay, we can say that two adjacent nodes in the overlay will be mapped to two arbitrary nodes in the underlay and their distance will be the expected underlay path length, $E\Big[|P_{un}|\Big]$. Thus, we can compute the expected underlay path length for any overlay path as follows:

$$E\Big[|\mathscr{R}(P_{ov})|\Big] \quad = \quad E\Big[|P_{ov}|\Big] \times E\Big[|P_{un}|\Big] \tag{3}$$

Hence, we can compute $\eta_{\mathscr{R}}$ for mapping $\mathscr{R}$ by combining Equations 2 and 3 as: $\eta_{\mathscr{R}} = E\Big[|P_{un}|\Big]$.

Now, to compute $E\Big[|P_{un}|\Big]$, we use the fact that the underlay, i.e., the Internet AS-topology, is a power law graph. In a power law graph, the number of vertices of degree d is proportional to $1/d^{\beta}$ for some exponent β. It has been reported in [13] that the range of β is $2 < \beta < 3$ for the Internet AS-topology, and for this range of β the average distance between two nodes in a power law graph, and hence $\eta_{\mathscr{R}}$ (an upper bound for $\eta_{\mathscr{M}}$), can be derived as follows [13]:

$$\eta_{\mathscr{R}} = E\Big[|P_{un}|\Big] = (2+\epsilon)\frac{\log\log N_{un}}{\log(1/(\beta-2))} \tag{4}$$

where $\epsilon << 1$.

We can also define a loose lower bound for the expansion ratio. Let us consider some mapping strategy $\mathscr{A}$ and a routing scheme $\mathscr{T}$ with the following property. If $\mathscr{A}$ maps overlay nodes x and y to some nodes x' and y', respectively in the underlay, then $\mathscr{T}$ can route messages between x' and y' in shortest path in the underlay network, regardless of the path length and the intermediate nodes between x and y in overlay. Thus we can write

$$\eta_{\mathscr{A}} = E\Big[\frac{|\mathscr{A}(P_{ov})|}{|P_{ov}|}\Big] = \frac{E\Big[|\mathscr{A}(P_{ov})|\Big]}{E\Big[|P_{ov}|\Big]} + Cov\Big(|\mathscr{A}(P_{ov})|, \frac{1}{|P_{ov}|}\Big) \tag{5}$$

If $\mathscr{A}$ is an adjacency preserving mapping then the covariance between $|\mathscr{A}(P_{ov})|$ and $|P_{ov}|^{-1}$ will be positive, otherwise for a random mapping it will be zero. In addition, the expected underlay shortest path ($E\Big[|\mathscr{A}(P_{ov})|\Big]$) between the mapped endpoints of P_{ov} is basically the expected underlay distance $E\Big[|P_{un}|\Big]$, since the distance between any pair of nodes in the underlay is calculated along a shortest path. Thus, we get from Equation 5 the following:

$$\eta_{\mathscr{A}} \leq \frac{E\Big[|P_{un}|\Big]}{E\Big[|P_{ov}|\Big]} \tag{6}$$

We can get $E\Big[|P_{un}|\Big]$ from Equation 4. For computing $E\Big[|P_{ov}|\Big]$ we exploit the fact that the overlay is a nearly regular graph. We can calculate the expected distance between any two overlay nodes as follows:

$$E\Big[|P_{ov}|\Big] = \sum_{|P_{ov}|=1}^{K} Pr(|P_{ov}|) \times |P_{ov}| \tag{7}$$

Here, $Pr(|P_{ov}|)$ denotes the probability that P_{ov} has length $|P_{ov}|$ and K is the overlay graph diameter. According to the Moore bound [14, 15] (p. 180), the diameter K of a d-regular graph ($d > 2$) with N_{ov} nodes can be derived as follows:

$$K = \frac{\log \frac{N_{ov}(d-2)+2}{d}}{\log(d-1)} \tag{8}$$

Since, the overlay graph is a regular graph, every node has d paths of length 1 and there are N_{ov} such nodes. So, the number of paths of length 1 is $N_{ov}d/2$. Again, from every node, we can reach d nodes, and from those d nodes, we can reach another $(d-1)$ nodes without considering the overlaps. So, the number of paths of length 2 is $N_{ov}d(d-1)/2$. Similarly, the number of paths of length $|P_{ov}|$ is $N_{ov}d(d-1)^{(|P_{ov}|-1)}/2$, for $1 \leq |P_{ov}| \leq K$. So,

$$\begin{aligned} Pr(|P_{ov}|) &= \frac{N_{ov}d(d-1)^{(|P_{ov}|-1)}/2}{\sum\limits_{|P_{ov}|=1}^{K} N_{ov}d(d-1)^{(|P_{ov}|-1)}/2} \\ &= \frac{(d-2)(d-1)^{(|P_{ov}|-1)}}{(d-1)^{K}-1} \end{aligned} \tag{9}$$

From Equation 7 and 9 we can derive that

$$E\Big[|P_{ov}|\Big] \quad = \quad \frac{(d-1)^K(K(d-2)-1)+1}{(d-2)((d-1)^K-1)} \tag{10}$$

In Section **??**, we compare the expansion ratio for our mapping strategy against the above computed bounds.

6 Related Work

The last few years have witnessed a significant number of research efforts in the field of ICN. Several of these research works address content naming and routing as key research challenges in ICN and have proposed different solutions for these problems. There is a clear contrast between the different routing mechanisms proposed for ICN. DONA [16] provides a hierarchical name resolution infrastructure including new network entities named Resolution Handlers (RH), which stores routing information of the domain it is attached with. The storage and computational load on RHs increase as we go up the RH hierarchy. The root RH needs to maintain routing information for all content in the network, which severely confines the scalability of this mechanism.

NetInf [17] adopts a hierarchical DHT [18] based approach for routing. The topmost level in the DHT hierarchy in NetInf, called REX, needs to store index for all the contents in the network, which results in a performance bottleneck and a scalability issue. LANES [19, 20] proposes a multi-layered routing architecture for ICN. The topmost layer, named the *rendezvous layer* is responsible for inter-domain routing, which is maintained by a hierarchical DHT.

CCN [1], CURLING [21] and TRIAD [22] use gossip based routing protocols, which incur significant management and control overheads. CCN also suffers from a scalability issue since the routing table size grows with the number of contents, which is several orders of magnitude higher than the current BGP routing tables. CURLING proposes to remove the DNS and assigns a new entity, "Content Resolution Servers (CRS)", with each ISP to maintain routing information. Content publish and request resolutions are performed using a gossip based hop by hop protocol maintaining the business relationships between the ISPs. CURLING also provides a mechanism for publishing and routing content requests in a non-global scope. Routing in CBCB [23] is based on controlled flooding of attribute-value pairs. Interest for a particular content is issued by creating logical disjunctions of conjunctions of elementary constraints over the values of possible attributes. CBCB includes: a broadcast protocol, which ensures loop-free paths; and a content based routing protocol, which avoids sending content to uninterested receivers by pruning branches of the broadcast tree. The size of routing table in a CBCB router is expected to be exponential in the number of attributes, and network wide flooding is required to forward a request with any unknown attributes, which incurs significant network overhead.

DMap [24] proposes a global name resolution service for mobility and efficient content delivery in the Internet. The flat ID (GUID) assigned to a content is hashed to obtain a list of IP addresses and the content's original location is indexed at these IP addresses. These indices are updated when a content is moved to a different location. DMap requires a collaboration of the ASs to store the content to location index of other ASs, and every network entity needs to have a global knowledge of the IP prefix advertisements from all the ASs. Moreover, the routers in the Internet have to store extra index along with the BGP routing table that they are already maintaining, which imposes a significant storage overhead. The CONET [25] project proposes a name based routing technique similar to CCN. The routers have a fixed number of rows in the name based routing table. When a router misses routing info for a name based routing request, it looks it up in a DNS like name-system and updates its name based routing table using some cache replacement policy.

7 Conclusion and Future Work

In this paper, we proposed a novel name based overlay routing scheme αRoute and an effective strategy for mapping an overlay network to the Internet AS-topology. αRoute guarantees content lookup while ensuring efficient bandwidth usage and small routing table size. The proposed mapping strategy, on the other hand, produces small expansion in routing path; according to the simulation results one overlay hop is mapped to 1.26 underlay hops on average.

Compared to the existing routing techniques, our approach has a number of advantages. First, routing can be done on names without sacrificing efficiency or completeness. Second, in contrast to hierarchical routing mechanisms, there is no bottleneck node in the proposed system. A capacity proportional load distribution can be achieved by placing the ASs at different levels in the partitioning tree based on capacity. Third, we can conveniently select the cardinalities of the partitioning sets ($|S_i|$), to tune the depth of the tree. This will allow us to easily decrease routing hops by increasing the number of routing table size, and vice versa. However, the proposed partitioning algorithm for constructing S_is has a shortcoming. We currently select S_is off-line in such a way that the sample names are uniformly distributed over the leafs of the tree. For a fairly large sample size, offline computation should give nearly uniform distribution of names over the resolution nodes. But, we intend to investigate other techniques for online computation of S_is for adaptive load balancing over a long period of time. In addition, the performance of αRoute can be greatly improved by adopting the caching strategies proposed in Section 4.4. We intend to investigate αRoute's performance in presence of indexing and content caching and experiment in a large scale testbed.

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. Braynard, "Networking named content," in *CoNEXT*, 2009.

[2] "BGP Routing Table Analysis Reports," http://bgp.potaroo.net/.

[3] J. Gantz, C. Chute, A. Manfrediz, and S. Minton, "The diverse and exploding digital universe, an updated forecast of worldwide information growth through 2011." IDC White Paper, Tech. Rep., 2008.

[4] We Knew The Web Was Big. [Online]. Available: http://googleblog.blogspot.com/2008/07/we-knew-webwas-big.html

[5] "Domain Counts & Internet Statistics," http://www.domaintools.com/internet-statistics.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., 1990.

[7] S. Bokhari, "On the Mapping Problem," *Comp., IEEE Trans.*, vol. C-30, no. 3, pp. 207 –214, March 1981.

[8] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the Internet topology," *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 4, pp. 251–262, Aug. 1999.

[9] M. Boguñá, F. Papadopoulos, and D. Krioukov, "Sustaining the internet with hyperbolic mapping," *Nature Communications*, vol. 1, p. 62, 2010.

[10] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz, "Characterizing the Internet hierarchy from multiple vantage points," in *IEEE INFOCOM*, vol. 2, 2002, pp. 618–627.

[11] S. A. Krashakov, A. B. Teslyuk, and L. N. Shchur, "On the universality of rank distributions of website popularity," *Comput. Netw.*, vol. 50, no. 11, pp. 1769–1780, Aug. 2006.

[12] O. Saleh and M. Hefeeda, "Modeling and Caching of Peer-to-Peer Traffic," in *ICNP 2006*, pp. 249 –258.

[13] F. Chung and L. Lu, "The average distance in a random graph with given expected degrees," *Internet Mathematics*, vol. 1, no. 1, pp. 91–113, 2004.

[14] C. Godsil and G. Royle, *Algebraic graph theory.* Springer New York, 2001, vol. 8.

[15] B. H. Simov, S. B. Tridandapani, and M. S. Borella, "Approximate bounds and expressions for the link utilization of shortest-path multicast network traffic," *Performance Eval.*, vol. 35, no. 12, pp. 49 – 74, 1999.

[16] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 181–192, August 2007.

[17] C. Dannewitz, "NetInf: An Information-Centric Design for the Future Internet," in *Proc. GI/ITG KuVS Workshop on The Future Internet 2009.*

[18] M. D'Ambrosio, C. Dannewitz, H. Karl, and V. Vercellone, "MDHT: a hierarchical name resolution service for information-centric networks," in *SIGCOMM ICN*, 2011, pp. 7–12.

[19] K. Visala, D. Lagutin, and S. Tarkoma, "LANES: an inter-domain data-oriented routing architecture," in *Proc ReArch 2009*, pp. 55–60.

[20] D. Lagutin, K. Visala, and S. Tarkoma, "Publish/subscribe for internet: PSIRP perspective," *Towards the Future Internet Emerging Trends from European Research*, vol. 4, pp. 75–84, 2010.

[21] W. K. Chai, N. Wang, I. Psaras, G. Pavlou, C. Wang, G. de Blas, F. Ramon-Salguero, L. Liang, S. Spirou, A. Beben, and E. Hadjioannou, "Curling: Content-ubiquitous resolution and delivery infrastructure for next-generation services," *Communications Magazine, IEEE*, vol. 49, no. 3, pp. 112 –120, march 2011.

[22] D. Cheriton and M. Gritter, "TRIAD: a scalable deployable NAT-based Internet architecture," Technical Report, January 2000.

[23] A. Carzaniga, M. J. Rutherford, and A. L. Wolf, "A Routing Scheme for Content-Based Networking," in *INFOCOM 2004*.

[24] T. Vu, A. Baid, Y. Zhang, T. Nguyen, Y. Fukuyama, R. Martin, and D. Raychaudhuri, "Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet," in *ICDCS '12*.

[25] A. Detti, N. Blefari Melazzi, S. Salsano, and M. Pomposini, "Conet: a content centric inter-networking architecture," in *Proc SIGCOMM ICN 2011*, pp. 50–55.