On-Demand Routing for Scalable Name-Based Forwarding

Onur Ascigil, Sergi Rene, Ioannis Psaras, George Pavlou
Dept. of Electronic & Electrical Engineering,
University College London, UK.
o.ascigil,s.rene,i.psaras,g.pavlou@ucl.ac.uk

ABSTRACT

Information-centric Networking (ICN) is a future Internet architecture design, where application-level names are directly used to route interests to fetch a copy of the desired content/data from any location. Following the conventions of the Internet Protocol to *store* the *pre-computed* routing/forwarding state for all prefixes at the network nodes raises *scalability* concerns in ICN (where content name prefixes need to be stored), especially at the inter-domain level. Instead, we consider the other extreme; that is, *On-Demand Routing* (ODR) computation for content name prefixes as interests arrive. We demonstrate through extensive simulations that ODR scales the storage of routing/forwarding information through caching and information discovery—two mechanisms inherent to the ICN design. More specifically, ODR makes use of domain-level, per-prefix routing instructions *usable by all the forwarders in a domain*, named *Routing Information Objects (RIO)*. Forwarders discover and retrieve RIOs in a similar way as content and then hand them to a *routing strategy* module to perform a routing decision before relaying the packets. We propose our design as an extension of the Named Data Networking (NDN) architecture and discuss all the proposed enhancements in detail.

1 INTRODUCTION

Information Centric Networking (ICN) is an architectural approach to evolve the "point-to-point connectivity" service of the current Internet to "retrieve data with a given name" service [25]. In ICN, consumers retrieve a copy of the data/content by sending *Interest* packets carrying the name of the desired data, and the service semantics is such that data packets can be discovered and retrieved from any node in the network independent of its location. The two mainstream ICN architectures, Named Data Networking (NDN) [13] and CICN [6], both use a *hop-by-hop name-based routing mechanism* to forward Interest packets towards data in the network. Using stateful forwarding mechanisms, name-based routing can support sophisticated strategies to discover content in the network [3, 24].

An important remaining challenge with name-based routing is its *scalability* properties. The two main "pinch points" that we believe are the primary scalability challenges are: i) storage cost of *"pre-computed"* forwarding information (based on routing information) for a very large number of name prefixes, estimated to be on the order of $O(10^9)$ [8, 20] and ii) bandwidth cost of disseminating and periodically updating routing information. In this paper, we focus mainly on the former problem. To deal with the scalability problem, the NDN architecture uses a namespace mapping mechanism [2]. This mechanism maps the name prefixes in the entire content namespace to a manageable set of topological names, *i.e.*, locators, such as the name of an ISP. As a result, the network maintains forwarding information for a small set of globally routable locators and optionally a set of (*e.g.*, most popular) content name prefixes in the Forwarding Information Base (FIB) tables. The network forwarders use locators—which are attached by end users and are carried end-to-end in the interests as *forwarding hints* together with the name of the desired data—as a fall-back when the forwarding state for the data name is missing in the FIB.

However, NDN's namespace mapping mechanism requires a secure binding of content names to the corresponding locators. This is because both identifiers are i) carried end-to-end in the packets and ii) can be placed in the packets by untrusted end hosts. Unfortunately, the verification of a secure binding between the two separate namespaces (i.e., content and locator) involves heavy computation (i.e., signature verification) and also execution of trust policies to determine which keys are legitimate to sign for a given prefix-to-locator binding. The verification of a secure binding might require verifying a chain of certificates leading to a trust anchor (e.g., DNSSEC). This makes it very difficult to verify the authenticity of the prefix-to-locator binding within the network. Absence of verification, on the other hand, makes it straightforward for malicious end hosts to launch attacks at chosen locations by simply attaching a victim's locator in the forwarding hint and carrying arbitrary data names such as unpopular (or even non-existent ones) names with high likelihood of a FIB miss.

Instead of an end-to-end namespace mapping mechanism, in this paper, we propose an On-Demand Routing (ODR) mechanism. In ODR, forwarders discover and retrieve *routing information objects (RIOs)* for name prefixes and use this information to make on-demand (i.e., as interests arrive) routing decisions as opposed to using forwarding state based on pre-computed routing decisions for all the prefixes. An

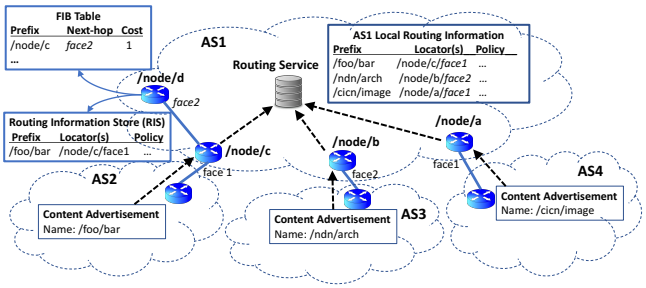

Figure 1: Routing Service collecting content advertisements.

RIO for a name prefix contains Autonomous System (AS) specific "instructions" on where to route the packet within the AS, for a given content name prefix. This information is generic, meaning that it is *usable by all the forwarders within an AS*, as opposed to forwarding information (i.e., prefix-to-next-hop) which is specific to a single forwarder. Therefore, the generic AS-level routing information in ODR is treated in a similar way to content by the forwarders within the same AS (*i.e.*, can be explicitly named and cached).

Specifically, a RIO for a name prefix comprises a set of *locators* (e.g., egress nodes of the AS) and *a policy hint*. The forwarders select the "best" locator as the routing decision through a new component named *a routing strategy*. Once a forwarder makes a routing decision, it expresses this decision in the Interest packet with a *forwarding hint*, similar to NDN. However, these hints are specific to an AS, as opposed to NDN's edge-to-edge forwarding hints, which needs to be globally known information. Within each AS, a logically centralized *Routing Service (RS)* acts as the default producer of routing information. RS collects (both inter- and intra-domain) name prefix advertisements to compute RIOs according to the routing policies of the AS.

As an example, consider Fig. 1 where AS1 (on the top) receives a content advertisement for a single name prefix from each of its three neighbors: AS2, AS3, and AS4 (on the bottom). The incoming content advertisements are routed directly to AS1's local routing service. After applying inter-domain routing policies, the routing service of AS1 builds and stores local RIOs for the AS1 as shown with the rectangular box to the right of Routing Service (RS) in Fig. 1. A locator of a prefix in the Routing Service is the identifier of a node and its face, and it is the initial point of the AS where the advertisement arrived at AS1's network. For example, the RIO for the prefix /foo/bar indicates a locator: interface 1 of node *c*, which is the arrival point for the inter-domain advertisement. Once a node obtains the RIO for a given prefix, it can determine a next-hop by using topology information which associates node locators with next-hops, and such information can be obtained through an intra-domain link-state protocol. The locator-to-next-hop mappings are stored in a "topological-only" FIB table, as shown on the top left hand side of Fig. 1 for node named */node/d*.

We demonstrate through simulations that ODR can perform name-based forwarding on content names with acceptable overhead, while requiring only minimal (topological) forwarding information to achieve scalable storage at the forwarders. The main overhead in our scheme is the potential latency and bandwidth overhead involved in the discovery and retrieval of routing information. In general, forwarders in ODR gather routing information for a name prefix by either i) passively observing forwarding hints in interests (without overhead) or ii) by actively searching for them in the network (with overhead). We minimize the overheads in active searches for routing information of name prefixes by exploiting locality of reference (*i.e.*, both temporal and spatial) in the name prefixes accessed by user interests through caching and discovery mechanisms, which are mechanisms inherent to ICN. Finally, our use of forwarding hints in ODR is strictly within an AS, which defines a trust boundary; therefore, we do not require verification or use of secure prefix-to-locator bindings as in NDN.

The rest of the paper is organized as follows. We discuss related work in Section 2. In Section 3 we present the on-demand routing mechanisms used in our solution, while in Section 4 we detail the routing and forwarding of packets and multiple strategies to retrieve routing information. We describe the performance evaluation in Section 5 and we conclude and discuss future work in Section 6.

2 RELATED WORK

Global routing scalability is one of the research challenges yet to be adequately addressed in the ICN community. One issue is the amount of routing and forwarding information, estimated to be on the order of $O(10^9)$. Prior work to improve ICN routing scalability can be classified in approaches that try to reduce storage cost of PIT information size [5, 22, 23], or FIB size [1, 9, 11, 14, 16] by reducing the routing information required in each of the network nodes.

Closer to our work, in [19, 20], the authors reduce FIB sizes by means of default routes to prefixes. In [20], the authors introduced the concept of *speculative forwarding* which keeps partial FIB information in local nodes and forward packets to a next hop, if no exact match is found. However, interests may experience looping in this case, which would cause them to be dropped. In [19], edge routers resolve and keep track of local name prefixes, and forward all other interests towards backbone routers that create a default-free zone and map name prefixes to globally-routed names. In [2], the authors use a mapping system (similar to DNS) to map all the name prefixes in the entire content namespace to a small set of globally reachable locators (*e.g.*, ISP names). However, this requires a secure binding of content names to the corresponding locators to be verified by the network.

Also using ISP-like prefixes creates duplicates of the same data and suboptimal caching.

In [15], the authors investigate the use of hyperbolic routing in NDN to reduce the FIB size and improve scalability. Hyperbolic routing is a promising solution as it does not require keeping a global routing state in any of the nodes. However, embedding arbitrary names in geometric space is still an unsolved problem in Internet-scale deployments.

Similarly to our proposal, in [8], the authors propose a FIB-as-a-cache scheme, to reduce the FIB size in the routers, using centralized RIB instance that is queried when a new Interest is received. However, in contrast to ODR, information cached in the FIB has only *local, node-specific significance* and cannot be shared with other nodes. That is, nodes need to query the central RIB, even when the requested routing information is cached nearby, reducing performance significantly.

In this paper, in contrast to the related work, we propose only maintaining at each node i) cached "routing information" for content names and ii) forwarding state pertaining to names of local forwarding nodes residing in the same ISP, i.e., Autonomous System (AS). The routing information maps globally reachable content names to local locations *i.e.*, possibly an egress node in the case of remote content or a local node in the case of locally produced content. Since routing information is usable by all the forwarders in the network as opposed to forwarding information that has only local (forwarder-specific) significance, each piece of routing information on a particular name prefix can be treated as cacheable data by the nodes of the same AS.

3 ON-DEMAND ROUTING MECHANISMS

Current technology can support routing entries in the order of millions of routes (see [7]), which is orders of magnitude smaller than the expected number of routes, i.e., $\approx 10^9$ name prefixes [8, 20]. Therefore, storage of routing and forwarding information for a large content namespace at a single forwarding node is not realistic. Instead, we propose an On-Demand Routing (ODR) scheme, where routing information of each domain[1] is collected and maintained by a local service of the domain, namely the **Routing Service (RS)**. In ODR, the network forwarders retrieve AS-specific routing information in the form of **Routing Information Objects (RIO)** and perform on-demand routing as interests arrive.

We assume the existence of a name-based, BGP-like inter-domain protocol and a name-based intra-domain protocol (e.g., NLSR [12]) through which content name prefix advertisements containing reachability information are disseminated. Content prefix advertisements arriving or originating within a domain are sent directly to the RS. We discuss content advertisements in Section 3.1. Then, we briefly discuss the storage of routing information by the distributed nodes of a Routing Service within a domain in Section 3.4. An important challenge with on-demand routing of packets is for the forwarders to consistently make routing decisions on Interest packets according to the routing policies of their domain. We discuss routing policy enforcement in Section 3.2.

In ODR, forwarders maintain only cached routing information in their *Route Information Store (RIS)*, which contains information for only a subset of the content name prefixes. Lookups to cached routing information containing only a subset of the name prefixes makes longest prefix matching not viable, because the longest prefix may not be present in the RIS. Instead, we require that Interest packets explicitly indicate their routable prefixes and the forwarders simply perform exact match lookup in their route cache. We discuss this point in Section 3.3.

3.1 Content Advertisements

We assume that globally propagated content advertisements are originated from content owners and also possibly from persistent storage systems (e.g., CDNs) that can act as producers, but not temporary storage such as in-network caches, for scalability reasons. We do not consider embedding location identifiers in content names, because it is against ICN's flexible content distribution and placement philosophy, *i.e.*, retrieve a copy of information from anywhere. The downside of location-independence is the lack of location-based aggregation, *i.e.*, it is not possible to summarize content stored at a location with the name of the location (e.g., /Comcast/*).

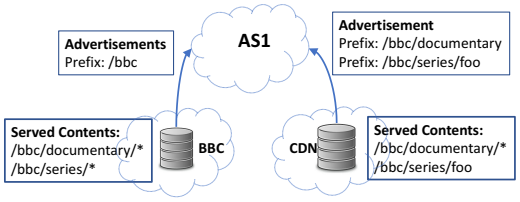

Figure 2: Determining Routable Prefixes.

Content advertisements contain *routable prefixes*, and we assume the following conventions when selecting routable prefixes of content names. First, the routable prefixes contain at least the content owners' name. In particular, a content owner may simply use its own name as the routable prefix such as /foo, when it serves all the content in-house for the entire sub-namespace /foo/*. On the other hand, for producers (e.g., CDNs) that serve a subset of the content for a content owner's namespace, the routing prefix needs to be more specific (we assume that the owners and other third-party producers that serve common content coordinate with each other to determine appropriate routable prefixes).

As an example, consider Fig. 2, where we have BBC as the content owner (bottom left) and a CDN (bottom right). The

[1]We use Autonomous System (AS) and domain interchangeably in the rest of the paper.

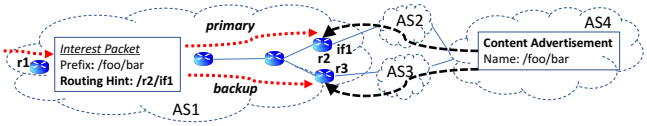

Figure 3: Use of Forwarding Hints to enforce routing policies.

CDN acts as a producer for a subset of BBC's content (*i.e.*, all documentaries and a single series named foo), whereas BBC can serve all its content in-house as shown in the Fig. 2 with a list of "Served Contents". As a result, BBC can advertise a very general routable prefix containing only its name as /bbc. On the other hand, the CDN must advertise more specific routable prefixes—in this case, /bbc/documentary/ and /bbc/series/foo.

3.2 Routing Policy Enforcement

In the proposed On-Demand Routing scheme (ODR), the *Routing Information Object (RIO)* for a content name prefix contains a list of locators and a policy hint. Currently, Autonomous Systems can receive on the order of tens to hundreds of advertisements for an IP prefix over different AS-level paths. We assume that this will be the case with content prefix advertisements, and an AS will have to select preferred route(s) among all the options. Interest packet forwarding typically requires retrieval of RIO for the prefix corresponding to the name in the packet from an external storage unless routing information for the prefix is already stored locally in the RIS cache of the forwarder. In case a RIO for the prefix is obtained, the forwarder selects a single locator among the available ones using the policy hint. A policy hint indicates the routing policy of the AS for a specific prefix and we discuss three example policy hints below.

In a *load-balancing* policy, forwarders alternate between a set of locators, *e.g.*, through hash load-balancing technique. Another possible policy is the *primary-backup*, which indicates one of the locators as the primary one, and only if the primary one is unreachable the forwarders fall back to secondary locators (we discuss failure detection and recovery in Section 4.3). Finally, in a *hot-potato* routing policy, forwarders select the locator that leads to the shortest path within the domain. This policy uses cost information to reach locators from the FIB table. In the ODR mechanism, FIB tables of the forwarders merely contain forwarding information for the intra-domain locators. As an example of the primary-backup policy, consider Fig. 3, where AS1 has two policy-compliant routes: one through AS2 and another one through AS3 for the prefix /foo/bar. AS1 uses the route through AS2 as the primary one. As a result, the forwarder on the left hand side selects the locator corresponding to this route.

In order to enforce the selection of a locator at subsequent forwarding nodes, a forwarder places the locator as *a forwarding hint* in the packet. For this purpose, we use NDN's existing "forwarding hint" field in the Interest packets [2]. This way, once a forwarding hint (*i.e.*, a single locator) is placed in the packet, the subsequent forwarders simply use the hint to forward the packet without having to search for routing information. Also, the hints in the packets can be cached by the forwarders to be used as routing information for future packets.

Unlike NDN's *end-to-end* usage of forwarding hints, in our case, forwarding hints are names of forwarders and are used strictly within an Autonomous System, which also defines a *trust boundary*. We require that forwarders perform checks on the interests at the edges of the trust boundaries *i.e.*, access and border nodes of ASs, in order to make sure that hints inserted by untrusted entities are cleared from the packets. Also, once a packet reaches the forwarder indicated by a hint, the hint is cleared from the packet.

The forwarding hints carried in the interests can be cached by the forwarders in the same AS observing the packets and used as routing information for future packets. Going back to the example in Fig. 3, a forwarder indicates the locator selection; that is, the primary locator by expressing it in the forwarding hint. As a final note, *our routing scheme does not require any changes in the packet format.* We simply change the e2e usage of forwarding hints and exploit them as intra-domain hints to consistently enforce routing decisions.

3.3 Interest Naming and Routable Prefixes

Interest packets contain full, hierarchical content names similar to NDN. However, in our case we require names to explicitly indicate the portion that constitutes the routable prefix. This is needed because each node must determine routes using partial routing information in its Routing Information Store (RIS), which makes longest-prefix matching not viable. Instead, each forwarder performs an exact-match lookup on the given routable prefix of the name in the Interest packets.

We envision that users obtain names of contents through out-of-band mechanisms such as search engines [2]. In addition to names of content, search engines can become aware of all the globally routable name prefixes by simply collecting content prefix advertisements. Therefore, a search engine can provide content names along with a list of routable prefixes for the name to the end users. For instance, consider again the prefix advertisements in Fig. 2. An Interest arriving at AS1 for $/bbc/documentary/X/1$ can use either the routable prefix $/bbc$ or $/bbc/documentaries$. In the former case, the Interest packet will be routed towards the BBC's in-house servers, and in the latter case, the packet will be relayed towards the CDN.

[2]In ICN, end users interested in a content object are assumed to learn its name through some trusted external mechanism [25].

3.4 Routing Information Storage

Within each AS, we use a set of dedicated storage nodes to store and process content advertisements. After the collection of content advertisements, the *Routing Service (RS)* computes a list of routable prefix-to-locator(s) mappings containing policy hints as discussed in Section 3.2. We store these mappings across multiple storage nodes of the routing service using standard sharding techniques [18]. For each RIO, there is an authoritative RS node within the domain that is responsible for serving the object.

The RS nodes advertise themselves and their identifiers in the bootstrapping stage of the forwarding/routing state establishment phase. That is, any node joining the network obtains the RS node identifiers from an adjacent node who was already part of the network. A forwarder simply hashes a name prefix to map it to an identifier of the RS node. A forwarder places the identifier of the authoritative node in the Interest packets when retrieving the RIO from the node. The authoritative server simply responds with the RIO back to the forwarder.

After describing all the necessary mechanisms for on-demand computation of routes, in the next section we describe the routing and forwarding functionality of ODR.

4 PACKET ROUTING AND FORWARDING

In this section, we first describe Interest packet processing in the ODR mechanism in Section 4.1. Data packet processing is identical to the one in NDN, whereas Interest packet processing differs in how routing and forwarding are performed. During Interest packet processing, the forwarders may perform a route discovery procedure, during which routing information is retrieved from the network to be able to direct the packets towards their destination. We will discuss possible strategies to perform routing information retrieval in Section 4.2. Another important function of the network is failure recovery. We discuss failure recovery mechanism with on-demand forwarding in Section 4.3.

4.1 Interest Packet Processing

In the following, we describe the Interest packet processing in ODR. We illustrate the processing steps in Fig. 4, where we also highlight the extensions to the NDN protocol.

Caching and PIT state. In Fig. 4, we depict the processing of Interest packets in the proposed ODR. Interest packet processing startswith a lookup in the Content Store (CS) and then a lookup in the PIT in case of CS miss, *i.e.*, same as in NDN. In case lookup to the PIT also fails, then the name of the Interest packet is added to the PIT. Otherwise, the PIT entry for the Interest name is refreshed (the new Incoming face is added to the face list), and the packet is discarded. Up to this point, Interest processing is identical to NDN.

Routing functionality. After adding the name of the packet to the PIT, the processing continues with checking if the packet carries a forwarding hint (i.e., a single locator). In case there is a hint, the forwarder has the option to cache this (possibly partial) routing information in the RIS cache to use it later for interests arriving in the future. In case, a hint has not been added to the packet (by another forwarder in the same AS), then the forwarder makes a lookup for the routing information associated with the routable prefix of the Interest packet's name from its local RIS.

If there is no entry in the RIS for the routable name prefix, the node initiates *Routing Information Discovery* process. This process, detailed in Section 4.2, will either i) search for routing information in the storage locations of the network (e.g., RIS cache of nearby node or RS) or ii) forward the interest itself towards the RS. In the latter case, the locator of RS is attached to the interest, and in the former case, a search for routing information is initiated as shown in the Fig. 4. In case, an RIO is retrieved, it is added to the RIS, and a single "best" destination locator is selected among possibly many locators.

The selection of a single locator is performed by a *Routing Strategy* module. A routing strategy is selected according to policy hints found in RIOs. Different strategies can be used to support various intra-domain routing policies (e.g., hot-potato routing), as discussed in Section 3.2.

Forwarding functionality. Once a locator is selected by the routing strategy and is added as a forwarding hint, the forwarder makes a lookup to its FIB table for the locator and extracts a set of possible next-hop faces. The result of the FIB lookup is then sent to the *Forwarding Strategy*, similarly to NDN, and a single out-going face is selected according to a forwarding strategy.

4.2 Routing Information Discovery

Using caching and content discovery mechanisms, *we shift the routing and forwarding decision in time and space as we describe in this section.* In case an Interest packet I arrives without containing a forwarding hint, the forwarder looks up (in its local RIS) for routing information associated with the routable prefix in the Interest packet's name. As described in Section 3.3, Interest packets explicitly indicate the routable prefix portion of their names. In case the forwarder is unable to find the routing information, it initiates a *Routing Information Discovery* process. Below, we describe three possible discovery strategies below:

(1) **Search On-path:** The main Interest packet I waits (*i.e.*, stored in the PIT table), while the forwarder sends an Interest packet to retrieve the routing information from the authoritative RS node for the prefix. Because any node may have the routing information cached in its RIS,

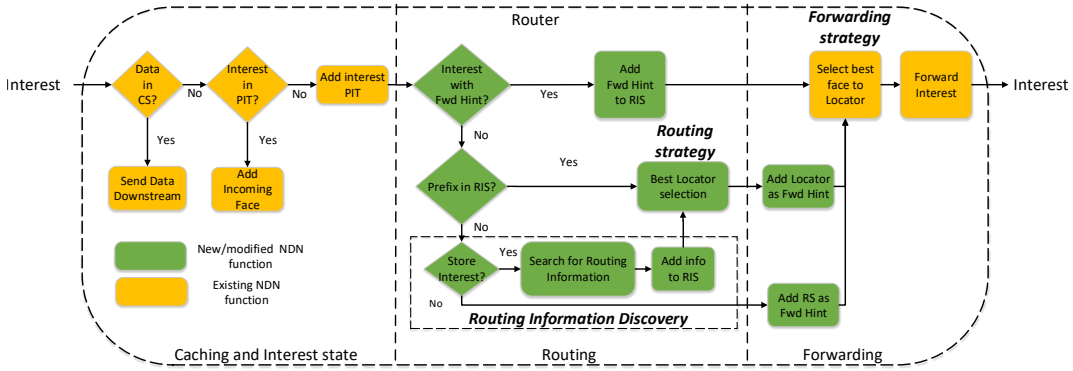

Figure 4: Interest packet processing.

a forwarder on-path to the Route Service can respond with the routing information.

(2) **Search Nearby:** Again, the main Interest packet I waits while the forwarder sends interests to request the routing information from nearby nodes, possibly using scoped flooding. In case the nearby nodes do not return any routing information (*i.e.*, returns nack), then the forwarder sends an interest directly to the authoritative Route Service node to retrieve the information.

(3) **Forward to Neighbor:** In this case, the forwarder relays the Interest I to the first hop neighbor along the path to the authoritative RS node. If a node along the path to the RS node has the routing information for I's prefix, it can divert I towards its destination.

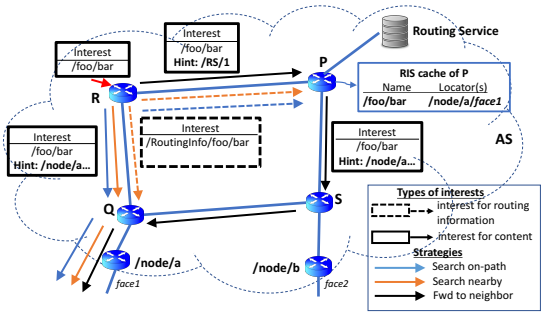

Figure 5: Routing Information Discovery Strategies.

Strategies one and two above shift routing decision in *time* (*i.e.*, delay computation until the necessary information is retrieved), while the last strategy shifts routing/forwarding decision in *space* (*i.e.*, moving the computation to another node). We demonstrate the three strategies with an example in Fig. 5, where an interest for a prefix /foo/bar arrives at R. Because R does not have the routing information for /foo/bar locally, it initiates a routing information discovery strategy. In case of search nearby strategy, R sends interests to the direct neighbors P and Q (shown with dashed orange arrows), and in the search on-path strategy, an interest for the routing information, destined to the RS node, is sent to P (shown with dashed blue arrow). We assume that these Interest packets sent for the purpose of retrieving routing information contain the special prefix /RoutingInfo as shown in Fig. 5. On the other hand, Forward-to-neighbor strategy forwards the actual interest (*i.e.*, for /foo/bar) to the first hop node P towards the RS node (shown with a straight black arrow). Because P has the routing information for /foo/bar in its RIS, the interest is redirected towards the locator (/node/a) indicated in the routing information. On the other hand, in the search strategies, once R retrieves the routing information from P, it forwards the original interest towards its locator.

One thing to note is that the Forward-to-neighbor strategy can cause Interest packets to loop and thus be dropped when forwarders divert Interest packet towards a locator, which was initially forwarded towards an authoritative RS node. In NDN, a nonce is carried in the interests and stored as part of the PIT state to detect loops as part of stateful forwarding. We propose to differentiate interest loops due to such diversions from other routing loops by storing the hints carried in the interests as they arrive together with the nonce as the PIT state. Once a diversion of an interest causes a loop, the node detecting the loop forwards the interest on towards its destination locator (*i.e.*, forwarding hint) instead of dropping the packet. We omit the furher details for space considerations.

4.3 Failure Recovery and Network Updates

NDN uses an *adaptive forwarding* mechanism to recover from (short-term) failures without the involvement of a routing process—*i.e.*, re-convergence of a routing protocol—so that packets can be routed around failing nodes/links. In adaptive forwarding, a forwarder maintains a status code per interface depending on the interface's current working status - this

is done for each name prefix. This information is kept in the FIB table in an NDN forwarder. NDN routing process can provide multiple next-hop (i.e., face) options per content prefix as well as a per-prefix ranking of the next-hop faces.

In the proposed ODR mechanism, each forwarder also keeps such status state for the next-hop faces for each entry in the FIB. However, we maintain *only per locator information i.e., not per content prefix*, in the topological-only FIB table. That said, we retain the benefits of adaptive forwarding and therefore, the forwarders within each domain can react to failures through a forwarding strategy that makes use of such status states. Different from NDN, forwarding a packet in our scheme may involve selecting a locator among the possible set of locators. The routing strategy in this case selects a single locator associated with the routable prefix in the Interest packet. However, failures may still lead to certain locators being unreachable at times. For that reason, we also maintain status codes for locators in the FIB table, in addition to next-hop interfaces, in order for a routing strategy to consider a locator's status when selecting one.

Also, in case of changes or updates to the content advertisements, it may be necessary to update the name prefix-to-locator routing information stored in the network. In that case, the RS would receive an update from the routing protocol for some name prefixes. Once it re-computes the new RIOs on updated prefixes, it will immediately notify the network with a routing update. Nodes with the corresponding RIOs cached have the option to update the cached RIOs.

5 PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed ODR scheme on a range of parameters. The objective is to evaluate the performance of ODR in terms of additional latency in information retrieval, load on the Routing Service and incurred overhead in the network. Next, we describe the setup of our evaluations, before presenting the experiments in the remaining sections.

5.1 Experimental Setup

For the evaluation of the ODR, we used the Icarus[3] simulator [17]—a Python-based discrete-event simulator for ICNs.

Topology: We use the Tiscali (AS 3257) network topology in the Rocketfuel dataset [21]. This topology has 161 routers, 328 bidirectional links and covers the European continent. We use this topology to simulate our name-based routing mechanism in a Tier-1 ISP, which must collect and store the entire name prefix advertisements disseminated globally at the inter-domain level. We attach two hosts to each router of the Tiscali topology; one to simulate a user and another one for a content producer. Out of the 161 nodes of the Tiscali topology, 80 nodes with the lowest degrees are designated as border routers of the domain. We refer to the rest of the 81 routers as *internal* routers. Users and producers that are attached to border routers are assumed to be located *external* to the domain, while the ones attached to internal routers are assumed to be located within the domain.

Parameters: We focus mainly on the parameters related to the scalability of routing and forwarding of Interests at the forwarders. These are the access distribution (i.e., popularity) of name prefixes and the size of the Routing Information Store (RIS) cache on each forwarder. Based on the assumption that the global prefix size is 10^9, we extrapolate a realistic ratio of the size of a forwarder's RIS cache to the global prefix size (i.e., $|RIS|/|P|$) using technical specifications of reasonably powerful routers. In particular, a BGP router can store $7.5M$ routes in its FIB table with the current technology [7]. This results in $|RIS|/|P|$ ratio of 0.0075, and we use this as the default ratio in the simulations. We investigate the impact of the RIS cache on the performance of the routing strategies in Section 5.4.

We assume that the Routing Service can provide scalable storage for global routing information within the domain through horizontal scaling. In all the simulations, the Routing Service consists of 10 authoritative servers, each randomly placed and storing an equal (statically assigned) portion of the global name prefixes through sharding. In order to identify the authoritative server responsible for a name prefix, a forwarder simply hashes the prefix and maps the result onto server identifiers, whose locations are discovered through intra-domain protocol. Also, we set the default size of Content Store (i.e., used only for caching content) of each forwarder to zero in order to focus on the performance of routing information discovery.

Table 1 provides the default simulation parameters for the experiments. Due to scalability issues with the simulator, we limit the size of name prefix at $|P| = 1M$ for all the experiments, which should not impact the results as we take appropriate (i.e., scaled-down) proportions of this value to obtain the size of RIS accordingly. As mentioned before, the distribution of content name popularity has been measured in different contexts (e.g., web caching) [4, 10] and was found to be of Zipf type with exponent value being approximately one. For the scalability of our routing scheme, what matters is the popularity distribution of the *name prefixes and not the full content names*. We expect an even more skewed distribution of name prefixes compared to full names, because of the many-to-one ratio between names and prefixes. Nevertheless, we use the default value for the Zipf exponent value as one in the experiments.

Workload: In the experiments, we assume a warm-up period of two hours during which the RIS caches of the forwarders are populated. This period is followed by a two-hour observation period during which we measure the performance.

[3]The simulator code with our extensions is publicly available in github.com/oascigil/icarus_route_service

Table 1: Default evaluation parameters.

Parameter	Value
Number of nodes V	161
|RIS|/|P|	0.0075
Number of name prefixes ($\|P\|$)	1M
Name prefix popularity (Zipf exponent)	1.0
Transit flow ratio	90%
Number of Routing Service nodes	10
Average number of producers per prefix	3

An average of 100 flows are initiated per second at exponentially spaced time points. A flow starts with a user issuing an interest and results with data being returned back to the user. Depending on the locations of the user and the producer of requested content, *i.e.*, interval or external, a flow is categorized as: transit, local, ingress, or egress. A transit (local) flow is one where an external (internal) user issues an Interest for a content whose producer is also external (internal) to the domain. Similarly, ingress (egress) traffic is one where an external (internal) user issues an Interest for a content whose producer is internal (external).

Flow categories are assigned randomly according to a probability distribution determined by a per-category ratio parameter. We assume that transit flows dominate the distribution of the overall traffic in a Tier-1 domain whose main business is relaying packets. Consequently, we assign a high ratio (i.e., 90%) as the percentage of transit flows over all the flows. The rest of the categories (i.e., ingress, egress, and local) equally share the remaining 10% of the flows. Similarly, we randomly partition the name prefixes into two as internal or external depending on whether they are served by external or internal producers.

Following the ratio of transit flows in the overall traffic, we assume that 90% of name prefixes are external and the rest are internal. In this work, we consider only the case of a transit domain, and we leave for future work the investigation of other types of domains such as content producers and eyeballs. To generate the workload for the experiments, we first determine the category of the flow according to the pdf based on the per-category ratios. Then, we uniformly at random select a user and pick a content prefix according to a Zipf distribution that are appropriate for the flow's category (e.g., pick an external prefix for a transit flow).

Similarly, we partition the set of name prefixes P as either external or internal depending on where its producer(s) is external or internal to the domain. We assume that 90% of prefixes are external to the domain. Then, we associate each routable prefix with on average three randomly chosen producers in the simulations. Because each producer is attached to a distinct router, each producer of a name prefix represents a different policy-compliant route to the prefix. The producer locators of each prefix is part of the routing information along with the policy hint. As the intra-domain routing policy, we use hot-potato routing in the experiments for all the name prefixes in all the simulations. We leave for future work, the investigation of applying different routing policies (e.g., per-prefix policies) and having different numbers of routes to name prefixes.

5.2 Performance Metrics and Strategies

Our evaluation is based on the following metrics:

- **Discovery Rate** (in ratio of issued interests): We measure the percentage of interests whose routing information is discovered from the RIS cache of a forwarder. When the discovery of routing information from the RIS caches of the network is not successful, routing information is obtained from a Routing Service (RS) node. This metric is a measure of the load on the RS.
- **Latency** (in milliseconds): This metric measures the average round-trip time (RTT) delay in retrieving content per issued interest. The RTT delay includes the amount of time it takes for the forwarders to retrieve routing information (for Search strategies) when there is a RIS miss at the first hop forwarder from the end user.
- **Overhead**: We use average of the total number of hops that routing information and interests for routing information travel in the network per issued interest as the overhead metric. This metric mainly indicates the bandwidth and packet processing overhead caused by on-demand retrieval of routing information by the forwarders.

We present the performance of the three strategies described in Section 4.2 together with two schemes that use FIB-as-a-cache based on the work by Detti *et al.* [8]:

- **FIB-Cache**: In this strategy, forwarders use their FIB tables as a cache to store forwarding information on content name prefixes. In case of a FIB miss, forwarders retrieve forwarding information (i.e., name prefix to output face identifier) from a centralized RIB service in an on-demand fashion. Because the forwarding information is only significant to a specific forwarder, this information is not cached by other forwarders.
  We enhanced this strategy with a "push mechanism" so that the RIB service can push forwarding information to multiple forwarders in order to avoid repeated FIB misses for the same Interest packet at subsequent hops along the packet's path. With this mechanism, a FIB miss for an Interest packet typically happens only once at a forwarder; once this forwarder queries the RIB service, the service pushes the necessary forwarding information on each subsequent hop along the future path of the packet.
- **FIB-Cache with Forwarding Hints**: This is an extension of the FIB-cache strategy, where forwarders retrieve *forwarding hints* (i.e., a single locator) from the RIB service together with forwarding information. Similar to the

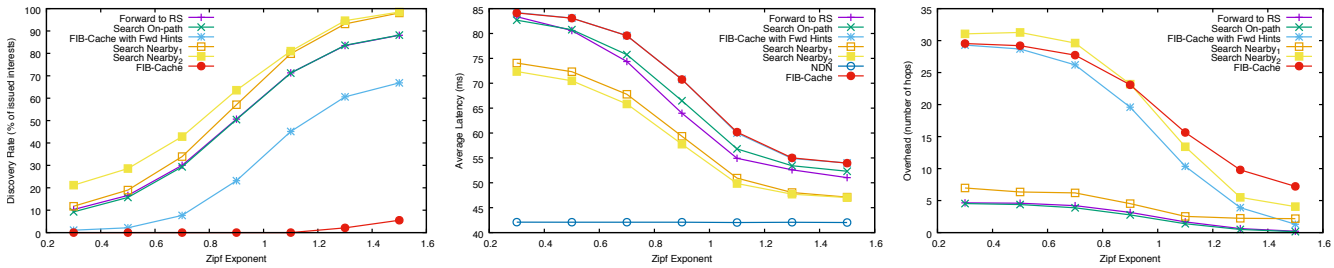

Figure 6: The impact of the Name Prefix Popularity in the performance of the routing strategies.

proposed strategies, the forwarding hint indicates a location within the AS. However, the locator provided by the RIB service is specific to the querying forwarder unlike routing information, which contains generic information usable by all forwarders in the AS. We assume that only the querying forwarder caches the hint and the forwarding information locally in this strategy.

In the rest of the section, we investigate the impact of several parameters and demonstrate the performance of the proposed strategies listed in Section 4.2 and the two FIB-as-a-cache schemes listed above.

5.3 Impact of Name Prefix Popularity

In Figure 6, we demonstrate the impact of name prefix popularity on the performance of the routing strategies. We observe from the leftmost plot that the Search Nearby strategies achieve the highest discovery rate for all the Zipf exponent values. The subscript x in Search Nearby$_x$ indicates the search range in terms of number of hops from the forwarder that is searching for the routing information. A two-hop search performs around 30% better than the other strategies for lower Zipf exponent values.

The Forward-to-RS and Search on-path strategies achieve nearly identical discovery rates, because both strategies simply search for routing information along the same set of nodes on the path leading to the Routing Service. On the other hand, both of the FIB-as-a-cache schemes achieve significantly worse discovery rates compared to the rest of the strategies. Particularly, the FIB-Cache strategy fails to discover routing information for Zipf exponent values ≤ 1.1. This is because FIB-cache strategy can only exploit locally cached forwarding information at each forwarder. A successful discovery requires each and every forwarder along the interest path to have the forwarding information cached locally (i.e., FIB hit), which is unlikely for low Zipf exponent values due to limited size of FIB tables. We set the FIB table size of the FIB-as-a-cache strategies to the same value as the RIS cache of the proposed strategies for a fair comparison.

We observe that the FIB-cache strategy can barely achieve 5% discovery rate for very high Zipf exponent values. In the rest of the experiments, we omit the FIB-Cache strategy from the results. On the other hand, the second FIB-as-a-cache scheme, i.e., FIB-Cache with Fwd Hints, performs significantly better than the first one. This improvement results from forwarders obtaining forwarding hints in addition to forwarding information from the RIB service and caching them both in their FIB tables. As a result, a FIB hit results with forwarding the Interest packet with a hint attached to it, and the subsequent forwarders simply use the hint to forward the packet. However, in this strategy, forwarding hints are cached only by the forwarders who originally obtain the hint along with the forwarding information from the RIB service, because the hints are specific to the forwarder who requested it as opposed to generic routing information retrieved in the proposed strategies.

The high discovery rates of the Search Nearby strategies leads to reduction in average latency in retrieving content as shown in the middle plot of Fig. 6. In all the experiments, we add a latency penalty of 5 msecs to retrievals of routing information from the Routing Service. We use the propagation delays obtained from the Rocketfuel dataset as the link latencies of the topology, and assigned a zero latency to the access link of the producers and end hosts attached to each forwarder. As a lower-bound on the achievable average latency, we use NDN with "best route" forwarding strategy, which simply picks the minimum latency path from the consumer to the producer with the link costs set proportional to the propagation delay of each link. We remind that the CS of the forwarders are disabled in all the experiments; therefore, the latency results reflect the overhead of routing strategies without any confounding factors such as the CS hit ratios on the results. Disabling content caching results in constant latency for NDN with increasing Zipf exponent values.

We observe that all the strategies achieve decreasing latencies as the Zipf exponent increases. For higher Zipf values, the Search Nearby strategies achieve near the lower-bound latency of NDN. Particularly, for Zipf exponents ≥ 1, the difference in latency drops to nearly 10 msecs. As expected, Forward-to-RS strategy achieves slightly better latency in comparison to Search On-path strategy, because the former strategy's search for routing information does not involve stalling the Interest packet. Instead, the Interest packets are forwarded towards the "producer" node of the routing information as described in Section 4.2.

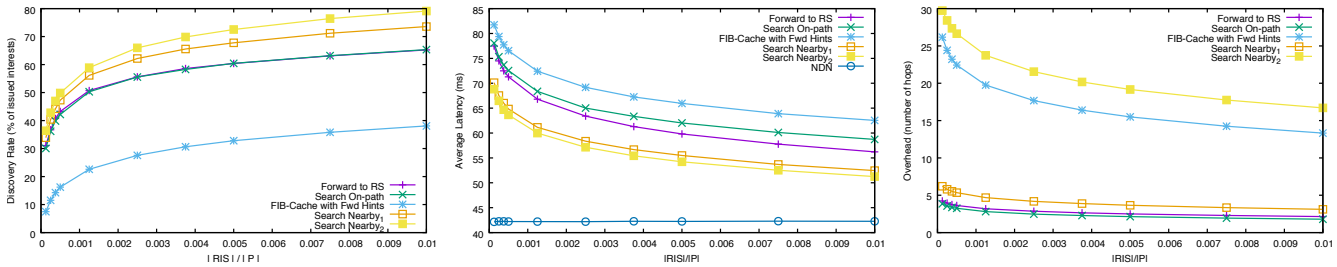

Figure 7: The impact of the RIS cache size in the performance of the routing strategies.

The overhead of the strategies (rightmost plot in Fig. 6) demonstrate the average number of hops traveled by the routing information (Data packet) and the corresponding Interest packets per issued Interests by end users. We observe that Search Nearby strategy with scope one achieves similar overhead with the Search On-path and Forward-to-RS strategies. On the other hand, increasing the radius of the search scope to two results in considerably higher overhead. As expected the FIB-Cache strategy performs worst for higher Zipf exponent values because a FIB misses happen frequently because a FIB hit only results with relaying the packet to the next-hop as opposed to getting the packet to its intended location with a forwarding hint.

In this section, we demonstrated that the ODR schemes can exploit locality of reference in the name prefixes of content requested by the interests to achieve scalable name-based forwarding using realistic amount of RIS storage at the forwarders. In the next section, we investigate the impact of RIS storage size on the performance of the strategies.

5.4 Impact of RIS Size

In Fig. 7, we depict the impact of Routing Information Store (RIS) capacity on the performance of the examined strategies. The RIS capacity of each node is expressed as a fraction of the name prefix population size, i.e., $|P|$. As we discussed earlier, we extrapolate the realistic capacity for RIS as $0.0075 \times |P|$ according to the FIB size of state-of-the-art BGP routers. In the following, we assume that each forwarder has the same RIS capacity and leave more sophisticated distribution of RIS storage capacity for future investigation.

We observe similar results as before in terms of discovery rates of strategies with increasing RIS size as shown in the leftmost plot in Fig. 7. Even with very small RIS sizes that are significantly below the realistic RIS size of 0.0075, the Search strategies achieve over 50% discovery rate. As expected, small RIS size leads to higher latency and overhead as shown in the middle and rightmost plots.

The above results demonstrate that the proposed routing strategies can perform name-based forwarding with acceptable overhead by each forwarder storing a very small portion of the routing information. The Search Nearby strategy with a scope of one and Forward-to-RS strategies both achieve high discovery rate with acceptable overhead. An important overhead of the "Search" strategies, is the processing of the Interest packets waiting at the forwarders for routing information to arrive. This overhead includes storage of the Interest packets in case of a RIC miss and scheduling their departure upon the arrival of routing information and can be significant. On the other hand, the Forward-to-RS can result with a stretch in the paths traveled by the Interest packets. One can alternate between the Search and Forward strategies in order to maintain the processing overhead of Interest packets, and we leave the investigation of limiting such processing overheads to future work.

6 CONCLUSIONS AND FUTURE WORK

The conventional wisdom dictates that in a hop-by-hop routing system, routers pre-compute and store consistent routes for the entire set of destination prefixes. Adaptive and stateful forwarding mechanisms of NDN relaxes this requirement so that forwarders can work with partial or even inconsistent forwarding/routing information without suffering serious problems such as persistent loops. In this work, we take this one step further and compute and store routing/forwarding information in a fully on-demand manner in order to scale the storage of routing/forwarding information. The on-demand computation makes use of Routing Information Objects (RIOs), which are generic AS-specific instructions on selecting policy-compliant routes for prefixes. In this work, we demonstrated that with a modest amount of storage at each forwarder to cache routing information, we can perform *purely* on-demand routing with reasonable bandwidth and latency overheads. In order to limit the overheads, we exploit the locality of reference in the destination routable prefixes contacted by users through caching and discovery of routing information.

As a future work, we plan to investigate hybrid mechanisms combining pre-computation and on-demand routing mechanisms. In that case, forwarders can use their slower memory (e.g., DRAM) to store less popular routing information and cache pre-computed forwarding information for popular content in their fast memory. We also plan to investigate the overheads of scheduling interests that are waiting for routing decisions under various traffic patterns.

REFERENCES

[1] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2013. Scaling ndn routing: Old tale new design. http://named-data.net/techreports.html. In *Technical Report NDN-0004*. 281–286. Online; accessed 29 April 2018.

[2] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. 2015. SNAMP: Secure namespace mapping to scale NDN forwarding. In *Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on*. IEEE, 281–286.

[3] Onur Ascigil, Vasilis Sourlas, Ioannis Psaras, and George Pavlou. 2017. A native content discovery mechanism for the information-centric networks. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*. ACM, 145–155.

[4] Paul Barford, Azer Bestavros, Adam Bradley, and Mark Crovella. 1999. Changes in web client access patterns: Characteristics and caching implications. *World Wide Web* 2, 1-2 (1999), 15–28.

[5] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, and Diego Perino. 2015. Pending Interest Table Sizing in Named Data Networking. In *Proceedings of the 2Nd ACM Conference on Information-Centric Networking (ACM-ICN '15)*. ACM, New York, NY, USA, 49–58. https://doi.org/10.1145/2810156.2810167

[6] Cisco. 2018. Cicn - Fd.io. https://wiki.fd.io/view/Cicn. (2018). Online; accessed 29 April 2018.

[7] Cisco. 2018. Cisco ASR 1000 Series Route Processor Data Sheet. https://www.cisco.com/c/en/us/products/collateral/routers/asr-1000-series-aggregation-services-routers/data_sheet_c78-441072.html. (2018). Online; accessed 29 April 2018.

[8] Andrea Detti, Matteo Pomposini, Nicola Blefari-Melazzi, and Stefano Salsano. 2012. Supporting the web with an information centric network that routes by name. *Computer Networks* 56, 17 (2012), 3705–3722.

[9] J.J. Garcia-Luna-Aceves, Maziar Mirzazad-Barijough, and Ehsan Hemmati. 2016. Content-Centric Networking at Internet Scale Through The Integration of Name Resolution and Routing. In *Proceedings of the 3rd ACM Conference on Information-Centric Networking (ACM-ICN '16)*. ACM, New York, NY, USA, 83–92. https://doi.org/10.1145/2984356.2984359

[10] Martin Halvey, Mark T Keane, and Barry Smyth. 2006. Mobile web surfing is the same as web surfing. *Commun. ACM* 49, 3 (2006), 76–81.

[11] Ehsan Hemmati and J.J. Garcia-Luna-Aceves. 2015. A New Approach to Name-Based Link-State Routing for Information-Centric Networks. In *Proceedings of the 2Nd ACM Conference on Information-Centric Networking (ACM-ICN '15)*. ACM, New York, NY, USA, 29–38. https://doi.org/10.1145/2810156.2810173

[12] AKM Hoque, Syed Obaid Amin, Adam Alyyan, Beichuan Zhang, Lixia Zhang, and Lan Wang. 2013. NLSR: named-data link state routing protocol. In *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. ACM, 15–20.

[13] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. 2009. Networking Named Content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*. ACM, New York, NY, USA, 1–12. https://doi.org/10.1145/1658939.1658941

[14] K. V. Katsaros, X. Vasilakos, T. Okwii, G. Xylomenos, G. Pavlou, and G. C. Polyzos. 2015. On the inter-domain scalability of route-by-name Information-Centric Network Architectures. In *2015 IFIP Networking Conference (IFIP Networking)*. 1–9. https://doi.org/10.1109/IFIPNetworking.2015.7145308

[15] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang. 2016. An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*. 1–10. https://doi.org/10.1109/IWQoS.2016.7590394

[16] Jie Li, Jiachen Chen, Mayutan Arumaithurai, Xingwei Wang, and Xiaoming Fu. 2015. VDR: A Virtual Domain-Based Routing Scheme for CCN. In *Proceedings of the 2Nd ACM Conference on Information-Centric Networking (ACM-ICN '15)*. ACM, New York, NY, USA, 187–188. https://doi.org/10.1145/2810156.2812600

[17] Lorenzo Saino, Ioannis Psaras, and George Pavlou. 2014. Icarus: a caching simulator for information centric networking (icn). In *Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 66–75.

[18] Lorenzo Saino, Ioannis Psaras, and George Pavlou. 2016. Understanding sharded caching systems. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*. IEEE, 1–9.

[19] Thomas C Schmidt, Sebastian Wölke, Nora Berg, and Matthias Wählisch. 2016. Let's collect names: How PANINI limits FIB tables in name based routing. In *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*. IEEE, 458–466.

[20] Tian Song, Haowei Yuan, Patrick Crowley, and Beichuan Zhang. 2015. Scalable name-based packet forwarding: From millions to billions. In *Proceedings of the 2nd ACM conference on information-centric networking*. ACM, 19–28.

[21] Neil Spring, Ratul Mahajan, and David Wetherall. 2002. Measuring ISP Topologies with Rocketfuel. In *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '02)*. 133–145.

[22] C. Tsilopoulos, G. Xylomenos, and Y. Thomas. 2014. Reducing forwarding state in content-centric networks with semi-stateless forwarding. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. 2067–2075. https://doi.org/10.1109/INFOCOM.2014.6848148

[23] Matteo Virgilio, Guido Marchetto, and Riccardo Sisto. 2013. PIT Overload Analysis in Content Centric Networks. In *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking (ICN '13)*. ACM, New York, NY, USA, 67–72. https://doi.org/10.1145/2491224.2491225

[24] Liang Wang, Suzan Bayhan, Jörg Ott, Jussi Kangasharju, Arjuna Sathiaseelan, and Jon Crowcroft. 2015. Pro-diluvian: Understanding scoped-flooding for content discovery in information-centric networking. In *Proceedings of the 2nd ACM Conference on Information-Centric Networking*. ACM, 9–18.

[25] George Xylomenos, Christopher N Ververidis, Vasilios A Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V Katsaros, and George C Polyzos. 2014. A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials* 16, 2 (2014), 1024–1049.