A Native Content Discovery Mechanism for the Information-Centric Networks

Onur Ascigil, Vasilis Sourlas, Ioannis Psaras, George Pavlou
Dept. of Electronic & Electrical Engineering,
University College London, UK.
o.ascigil,v.sourlas,i.psaras,g.pavlou@ucl.ac.uk

ABSTRACT

Recent research has considered various approaches for discovering content in the cache-enabled nodes of an Autonomous System (AS) to reduce the costly inter-AS traffic. Such approaches include *i)* searching content opportunistically (on-path) along the default intra-AS path towards the content origin for limited gain, and *ii)* actively coordinate nodes when caching content for significantly higher gains, but also higher overhead. In this paper, we try to combine the merits of both worlds by using traditional opportunistic caching mechanisms enhanced with a lightweight content discovery approach. Particularly, a content retrieved through an inter-AS link is cached *only once* along the intra-AS delivery path to maximize network storage utilization, and *ephemeral forwarding state* to locate temporarily stored content is established *opportunistically* at each node along that path during the processing of Data packets. The ephemeral forwarding state either points to the arriving or the destination face of the Data packet depending on whether the content has already been cached along the path or not. The challenge in such an approach is to appropriately use and maintain the ephemeral forwarding state to minimize inter-AS content retrieval, while keeping retrieval latency and overhead at acceptable levels. We propose several forwarding strategies to use and manage ephemeral state and evaluate our mechanism using an ISP topology for various system parameters. Our results indicate that our opportunistic content discovery mechanism can achieve near-optimal performance and significantly reduce inter-AS traffic.

CCS CONCEPTS

• **Networks → Network architectures**; *Network management*; Network simulations;

KEYWORDS

Off-path caching; Content discovery, Routing, Management.

1 INTRODUCTION

A content discovery mechanism can leverage the distributed system of caches within an AS to retrieve content *locally* as opposed from the possibly external content origin. An effective content discovery mechanism can significantly reduce the costly inter-AS traffic for content retrieval, and also provide lower retrieval latency (*i.e.*, QoS) for the users. Information Centric Networking (ICN) makes it easier to implement a fully-distributed content discovery mechanism at the network layer through its functionalities such as name-based, stateful forwarding and ubiquitous caching.

Despite the supporting functionalities, the two main ICN architectures NDN and CCN [11, 30] implement a very limited form of intra-domain content discovery mechanism, namely *opportunistic* discovery. In an opportunistic content discovery mechanism, content is searched *opportunistically* along the shortest path (or a designated path) towards a content origin. This approach has a very limited search scope (only the nodes along a path) and thus limited gain, but does not require coordination or communication among the nodes.

Conversely, in *coordinated* content discovery approaches, requests are forwarded to some designated cache that is likely to hold this content [3, 20, 22]. These techniques can attain significantly higher gain at the cost of extra coordination and communication overhead (*e.g.*, update protocol or signaling) to maintain *up-to-date* state of the cached content.

In this paper, we try to combine the merits of both worlds by using an *opportunistic coordination* approach, which requires each router to keep track of only a small amount of state and does not require signaling or update protocols to locate content. To that end, we enhance the NDN routing architecture with a new component called *Ephemeral Forwarding Information Base (EFIB)*, which keeps track of the direction (next hop) in which data chunks were *temporarily* cached in the recent past. An entry in the EFIB is created opportunistically by a returning Data packet and comprises a trail for a matching Interest packet to follow towards the direction of a cache, where the corresponding data has been placed. Collectively the EFIB entries at individual routers form *trails* towards cached content.

The establishment of EFIB state for a content is demonstrated in Figure 1, where a Data packet entering an AS at R_i follows the intra-AS path: R_i, R_k, R_p, R_m, R_u to reach a user. At R_p, the content of the Data packet is cached following the content placement policy. The resulting EFIB trail for the named content carried in the packet is shown in the figure with arrows. Having such a mechanism to locate cached content makes it possible to limit the placement of content to be performed *exactly once along the content delivery path* (*i.e.*, in node R_p in Figure 1). This reduces caching redundancy and in turn, makes more efficient utilisation of available cache resources along a delivery path.

Different from the Interest packet processing in legacy NDN, in our approach an interest can be forwarded using both FIB and EFIB. While EFIB-based forwarding takes Interest packets towards a

ICN '17, September 26–28, 2017, Berlin, Germany

ACM ISBN 978-1-4503-5122-5/17/09…$15.00
https://doi.org/10.1145/3125719.3125734

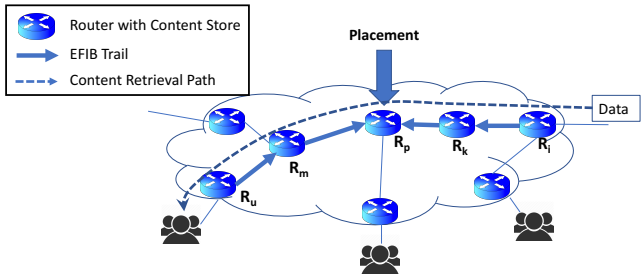

Figure 1: Placement of data and the EFIB trail.

cache, the FIB-based forwarding takes packets towards the content origin[1].

The introduction of the EFIB table requires new *forwarding strategies* for next-hop selection for the matching EFIB and FIB entries, with trade-offs in retrieval latency and overhead in terms of duplicate Data packet responses and circulated Interest packets. After introducing two extreme forwarding options (*i.e.*, *Multicast* and *Stop-and-wait* in Section 3), we propose a *budget-based multicast forwarding* strategy (in Section 4), where each Interest packet is given a "forwarding budget" that can be spent (at the intermediate routers) on forwarding the Interest on-path towards the content origin and/or sending off-path interests towards caches. Particularly, a router can choose to either spend the entire budget on searching the nearby off-path caches (*i.e.*, following the EFIB trails within the AS), or search less aggressively off-path and let the Interest packet leave the AS, and thus reach the external content origin.

We evaluate (in Section 5) the proposed content discovery mechanism through extensive simulations in realistic AS topologies for various system parameters. The proposed budget-based forwarding strategy leads to more than a threefold increase in the number of locally (*i.e.*, intra-AS) satisfied content requests in comparison to NDN's default on-path (*i.e.*, shortest path) discovery mechanism, while incurring negligible overhead. Additionally, the proposed budget mechanism achieves good QoS (*i.e.*, low latency in retrieving content) for users, whereas its overall performance is very close to a theoretically optimal forwarding strategy.

2 RELATED WORK

Apart from the optimization of the packet-level content placement in ICN, which has been extensively studied from different aspects, another important part of the in-network caching research is the actual content discovery mechanism. The content discovery mechanism, or else, *request to cache routing* is the mechanism to direct content requests to the right cache mainly to increase the locally satisfied requests and minimize content delivery latency. By and large, request-to-cache routing can follow one of two approaches: either *opportunistic on-path*, where content is searched on-path as the request is traveling towards the content origin, or *co-ordinated off-path*, where requests are forwarded off the shortest path to the content origin or some designated caches that are likely to hold this content.

The off-path resolution-based routing is a deterministic solution which maps requests to content items cached in nearby (or not) nodes, usually at predetermined rendezvous points *e.g.*, [3, 17, 20, 22, 26]. The off-path request to cache routing solutions usually require an extra routing table at each router to statically or according to a content addressing scheme configure the path towards the corresponding rendezvous points or the off-path caches. Representative proposals in this space are [8], [12], [14] and [19]. This state can be maintained either in a distributed manner, in which case a signaling protocol is required [9], or in a centralized manner as in a Name Resolution Service (NRS) [3], in which case a registration/update communication is required.

The strategies proposed in [7], [21] and [29] use co-ordination techniques between the data and the control plane to place content and re-direct requests to the corresponding caches. In [16] two methods are proposed to route requests to the nearest replica of a content by either flooding requests or meta-requests to discover the content location. In [17, 22], the authors utilize hash-routing techniques, which have been proposed in the past for mapping requests to physically co-located servers. Each router in the network is assigned a part of the hash space and caches the content items whose hashed identifiers fall within that space. This way, hash-routing avoids all the complex request-to-cache resolution steps of similar proposals and minimizes the corresponding signaling overhead. Finally, the authors in [28] have proposed a scoped flooding-based content discovery mechanism. The proposal includes a ring model, which limits the spread of the flood to the immediate neighborhood. The results show that although scoped-flooding introduces some overhead, it is far from prohibitive and can scale and achieve considerable gains.

In Breadcrumbs [15], the authors considered an opportunistic forwarding scheme, where nodes form trails towards users —and not towards the caches where content is stored— opportunistically during the forwarding of Data packets. Similarly, in [2], we built an ICN forwarding mechanism around Breadcrumbs using the NDN forwarding architecture. As shown in [2], this forwarding strategy, where nodes keep track of the direction of Data packets towards the end users, performs the best assuming the exhaustive leave-copy-everywhere (LCE) placement strategy, since any probabilistic placement of content might result in following trails that do not terminate at nodes where the corresponding content is stored. The usage of LCE minimizes though the merits of this opportunistic forwarding strategy. In contrast to [2] and [15], here we examine a more sophisticated forwarding strategy that keeps track of the trails towards the caches where content is stored that allows the usage of better placement schemes, which eventually minimize the inter-AS traffic without introducing unacceptable latency or overhead.

3 OPPORTUNISTIC CONTENT DISCOVERY

In this section, we present the functionality of the proposed content discovery mechanism. The rationale behind our design is to coordinate forwarding and caching without introducing significant overhead. This is achieved by increasing each router's awareness regarding the cached content within its network and benefit from the name-oriented routing and forwarding to retrieve content from an intra-AS off-path cache.

[1]Note that EFIB entries may also point towards the content origin, if a cache with the matching content happens to be on-path towards the origin, in which case the forwarding strategy has fewer next-hop options to consider.

3.1 Ephemeral FIB

In order to augment NDN architecture with a content discovery mechanism, we add an *Ephemeral FIB (EFIB)* table to the original NDN content router design presented in [11]. The functionality of the remaining NDN router components, namely the Content Store (CS), the Pending Interest Table (PIT) and the Forwarding Information Base (FIB) remain the same.

We utilize part of the content store to maintain the EFIB, which maps name prefixes to a set of next hops, similar to the FIB table. However, FIB and EFIB differs in several important aspects. First of all, the FIB table of a router is used to forward requests towards content origins. The EFIB table, on the other hand, is a temporary storage of forwarding information used to forward interests towards local (*i.e.*, intra-AS) caches. Secondly, unlike the FIB table, which is traditionally populated via out-of-band means such as a routing protocol [9], the EFIB table is populated opportunistically (see below), using locally available information gathered from the forwarded Data packets.

An EFIB entry is triggered by a returning Data packet and points towards the direction (*i.e.*, next hop) in which data chunks were temporarily cached in the recent past. Similar to the FIB and PIT entries, EFIB entries also follow the hierarchical content naming scheme of NDN, where a content item is segmented into chunks and each chunk is uniquely identified (*e.g.*, `Music/Artist /Song/chunkID`). This means that a returning Data packet "carrying" any of the chunks of the above item, *e.g.*, `Music/Artist/Song/chunkID/CID1`, will trigger an entry in the EFIB of the corresponding router of the same form, coupled with the interfaces pointing towards the router along the delivery path that cached the aforementioned chunk. This implies that the direction of the EFIB entries will interchangeably point downstream or upstream the corresponding delivery path (*i.e.*, to the arriving or the destination face of the Data packet) upon the caching of a chunk at a router, whereas the FIB and PIT entries point always towards the same direction (*i.e.*, the FIB upstream towards the content origin and the PIT downstream towards the requesting user).

3.2 Placement strategy

In NDN [11], every router along the path from the content origin or responding router towards the requesting user caches the passing-by Data packet. Due to the inefficiency of this ubiquitous caching mechanism we assume here a variation of the probabilistic caching strategy presented in ProbCache [13]. Particularly, we assume that *each Data packet is cached exactly once along the delivery path.* The caching point is decided based on the caching capability of the path and the corresponding position of a given router in the network, similarly to [13]. The use of the EFIB table makes redundant further replication of the given chunk along the same delivery path, since at a very small extra delay any interest that will "meet" the corresponding EFIB entries will retrieve with high probability the corresponding data chunk from the same intra-AS path.

In order to implement this content placement strategy and enable the installment of the EFIB entries, we introduce a "*Placement flag (PL)*" bit to the Data packets. PL set to "1" indicates that a placement is not necessary (*i.e.*, the packet is either responded by a router or has already been cached) and conversely PL set to "0" indicates that a placement should be performed.

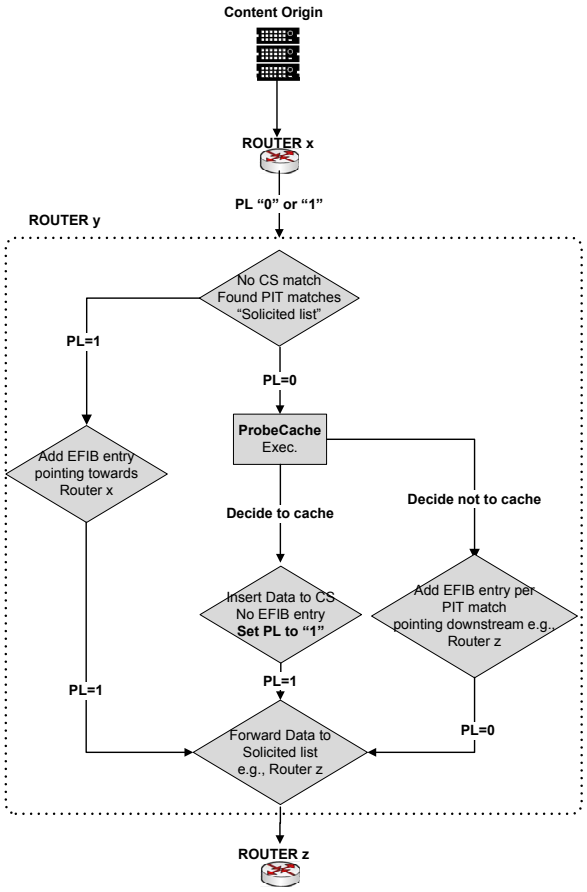

Figure 2: Data packet processing by an intermediate router.

Here our objective is to maximize network storage utilization and we enforce single caching along a delivery path. However, the proposed content discovery mechanism is generic and can support multiple replication of the same content along a path (*e.g.*, to support time sensitive and/or critical applications [23] or to minimize the usage of expensive inter-domain links [1]). In such a scenario, the application can provide its own latency constraints and/or placement strategy by setting the PL field (now a number and not a bit) to the corresponding number of replicas that should be installed along a delivery path. Next we describe the mechanism to populate the EFIB tables and the processing of a returning Data packet by an intermediate router.

3.3 Data packet processing

Generally, the Data packet processing of the proposed content discovery mechanism is exactly the same to NDN, with the exception of the PL flag. Particularly, when a Data packet arrives from an inter-AS link, it is flagged (PL is set to "0") by the border router of the AS. However, when an intra-AS cache forms a Data packet (as a result of a cache hit), it sets PL to "1". The same also occurs when a router decides to cache the corresponding packet, before forwarding it towards the requesting user.

In more details, a match lookup of a Data packet's Content Name takes place upon the arrival of the packet at each router as in NDN

[11]. A CS match means that the Data packet is a duplicate so it is discarded, whereas a PIT match (there may be more than one) means that the Data packet was solicited by interest(s) forwarded by this router. In that case a list is created (*i.e.*, "Solicited list"), that is the union of the Requesting Faces list of each PIT match minus the arrival face of the Data packet and the following three scenarios exist:

(1) If the PL flag of the corresponding incoming Data packet is equal to "1" (*i.e.*, the packet has already been cached along the delivery path) the corresponding router sends the Data packet on each face of the Solicited list, the PIT entries are removed, and a new EFIB entry is created pointing towards the arrival face of the Data packet. The new EFIB entry is exactly the same with anyone of the PIT entries matching the Data packet with the exception that it points upstream towards the incoming face of the Data packet.
(2) If the PL flag of the corresponding incoming Data packet is equal to "0", the corresponding router decides based on ProbCache whether or not to cache the packet. If the router decides to cache the packet, it sets the PL flag to "1" and sends the Data packet on each face of the Solicited list as above. However, the router does not add any EFIB entries since it is the router where the packet is cached and a passing by matching interest following either FIB of EFIB entries will find the packet in its CS. The corresponding EFIB trails (upstream or downstream) will terminate at this router.
(3) Finally, if the PL flag of the corresponding incoming Data packet is equal to "0" and the router decides not to cache the packet, the Data packet is send on each face of the Solicited list as in the above scenarios with the PL set to "0", and for each face of the list a new EFIB entry is created. The new EFIB entries are exactly the same as the PIT entries matching the Data packet and point towards the direction that the Data packet was sent. In this scenario an EFIB entry points towards more than one directions. This however does not violate our single path caching policy described above, since each entry points on different delivery paths belonging on different user requests.

In Figure 2 we depict the processing diagrams of an incoming Data packet at an intermediate NDN router.

According to the above description, a router opportunistically decides whether to cache or not a passing by Data packet based on its PL value. Also, based on the caching decision the EFIB entries point either upstream towards the direction of a cached packet, or downstream towards the direction where the packet will eventually be cached. In order to make sure that at least one router along the delivery path will cache the Data packet we assume that the last router along the delivery path (*i.e.*, the router where the user issued the interest is attached to) alway caches the Data packet if the corresponding PL is equal to "0". Having populated both FIB and EFIB tables to forward packets, a forwarding strategy is necessary to select the next-hop(s) using both the matching EFIB and FIB entries.

3.4 Forwarding Strategies

The goal of any forwarding strategy is to implement an effective content discovery mechanism leveraging the intra-AS caches to limit the retrieval of content using inter-AS links. This should be achieved with low overhead (*i.e.*, duplicate responses and multiple forwarded interests) for the AS and acceptable content retrieval latency (*i.e.*, QoS) for the users.

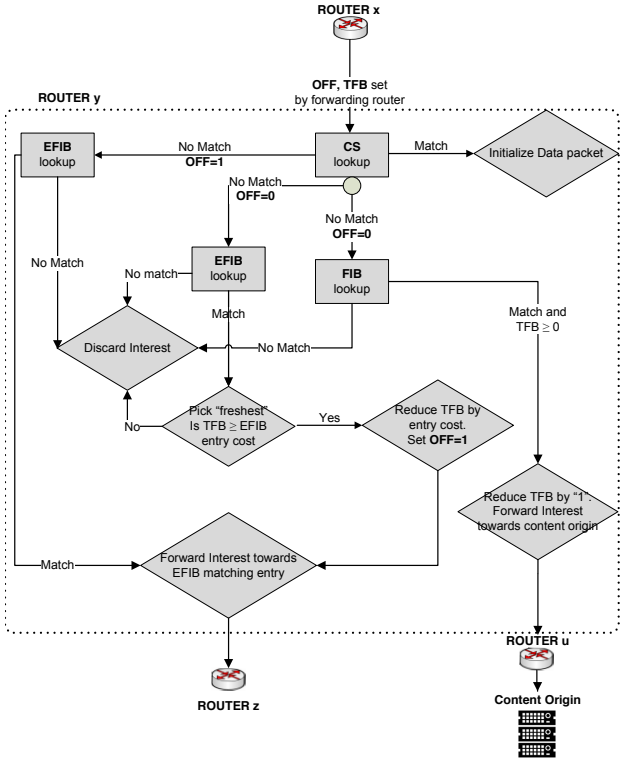

Figure 3: Interest packet processing by an intermediate router including the TFB element introduced in Section 4.

We introduce an *Off-path Forwarding Flag (OFF)* bit to the Interest packet to distinguish whether it is following the FIB entries towards the content origin (OFF is set to "0"), or is heading towards the direction of off-path caches. In the second case (OFF is set to "1"), the Interest packet follows matching entries in the EFIB of each passing-by router.

Whenever a user issues an Interest packet the OFF bit is by default set to "0". This means that when an Interest packet arrives on some face of a router, the router processes the packet in the exact same way as in NDN [11]. Particularly, if a matching chunk is found in the CS, the router sends back to the face the interest arrived on the Data packet (with the PL flag set to "1") and discards it. If the router does not find matching content in CS and there is an exact-match PIT entry, the interest's arrival face is added to the PIT entry's Requesting Faces list and the interest is discarded.

In the proposed native content discovery mechanism, if there is no matching entries both in CS and PIT, the router searches besides FIB and the EFIB for matching entries. If matches are found in the EFIB the arrival face is removed from the face list of the matching entries and if the resulting list is not empty, a number of off-path Interest packets (*i.e.*, OFF set to "1") can be initiated. Additionally with the off-path interests and if there is a matching FIB entry, the router can also forward the Interest packet towards the content origin. In both cases for each forwarded interest packet (off-path or

on-path) a new PIT entry is created from the interest and its arrival face.

When an Interest packet arrives on some face of a router and its OFF bit is set to "1" (*i.e.*, off-path interest), the router checks for a matching content in its CS. If a matching content is found, the router sends back the corresponding Data Packet (also by setting the PL flag to "1"). If the router does not find a matching content in CS, it searches the PIT. If a matching entry is found, the interest's arrival face is added to the PIT entry's Requesting Faces list and the Interest packet is discarded. Otherwise, it skips searching the FIB and checks for an exact-match EFIB entry. If no such entry is found in EFIB the router discards the Interest packet. This means that either the router is the last on the path of routers set by the corresponding EFIB entries, or a matching entry has been replaced/invalidated (see Section 4.2) before the arrival of the interest. Note that interests with the OFF bit set to "1" only follow EFIB entries. In that case only up to one interest can cross the AS boundaries to request content from the content origin. In Figure 3 we depict the processing diagrams of an incoming Interest packet at an intermediate NDN router.

The number of the new off-path interests as well as their scheduling (*i.e.*, order and time of issue) depends on the used forwarding strategy. A router with matching EFIB entries can initialise all (or part) of the off-path interests simultaneously, or follow a more conservative approach by waiting the response in some of them (*e.g.*, one at a time) before initiating new ones. Generally, there are two extreme forwarding strategies. In the one extreme is the *Multicast* strategy where a router can choose to follow all the EFIB matching entries in parallel with the FIB ones, in an attempt to maximize the likelihood of obtaining content from within the AS. The other extreme is the *Stop-and-wait* forwarding strategy (similar in rationale to [15]) which gives priority to off-path interests. Particularly, EFIB entries are followed one at a time and the router "waits" for the responses (either Data packet or NACK) of them before forwarding towards the content origin.

3.5 Preliminary results

In Figure 4 we depict some preliminary results for a default network setup (see Evaluation section) of the two extreme forwarding strategies in terms of content discovery rate, latency and overhead. We also depict the performance of the traditional NDN on-path forwarding strategy as well as the theoretically optimal strategy, named *Nearest Replica Routing (NRR)* strategy, which fetches the requested content from the closest intra-AS replication point (if there is one) regardless of the EFIB entries. The NRR uses the same placement policy with the other two strategies, but since it assumes a global replication knowledge the number of intra-AS replicas of each content item is at most one (never more than one).

Without getting into details at this point, we observe that the Stop-and-wait strategy, which exhausts all possible off-path trails, one at a time, is prone to large retrieval latency in fetching content, but leads to low overhead and no duplicate responses. On the other hand, Multicast can achieve very small latency in retrieving the content, but potentially leads to high overhead, *i.e.*, many copies of data can be fetched simultaneous. In the following section we explore a set of multicast forwarding strategies that trade-off between retrieval latency and overhead using a budget-based multicast mechanism in an attempt to bridge the merits of the two extreme cases and reach as much as possible the optimal performance of the NRR.

Another candidate forwarding strategy is that of scoped flooding presented in [28]. However, as can be seen from the NRR performance, an interest (on or off-path) needs on average six hops to retrieve the requested content in the used network setup and with the used content placement mechanism. This implies that a scope of that size (around six or larger) should have been used with the scoped flooding strategy too. This however, would merely flood the whole network with interests and for that reason we don't consider this strategy here. Scoped flooding performs better when more replicas of a content are placed within the network and the used delivery paths.

4 MULTICAST STRATEGIES USING FORWARDING BUDGET

An Interest packet finding matching EFIB entries at a router can be multicasted towards all matching directions. This, however, might have a significant overhead, as shown in Figure 4. In this paper, we allow only Interest packets that head towards the origin (OFF bit set to "0") to initiate off-path interests, which means that off-path interests are not further branched/multicasted. Additionally, we allow each interest to initiate up to a specific number of off-path interests and not as many as possible as in the Multicast strategy described above.

In order to implement this, we also introduce at the Interest packet a *Total Forwarding Budget (TFB)* counter to control the total number of interests that will be initiated. Particularly, we assume that each router, where a user issues an interest, is aware (*e.g.*, based on the FIB entries) of the distance (*i.e.*, in hops or any other cumulative link metric, following the shortest path) to the content origin (*i.e.*, the egress AS node towards the content origin) and the TFB is set to a value equal or larger than this distance. The TFB value can be either consumed to reach the content origin (by decreasing it accordingly at every hop, *e.g.*, by one if TFB is in hops), or it can be consumed to initiate off-path interests (or both). In this paper we assume that the EFIB functions similar to an LRU cache and each intermediate router can initiate off-path interests, as long as there is enough budget in the Interest packet TFB, and we consider two different TFB charging schemes upon the initialization of an off-path interest. Figure 3 presents how the TFB is consumed during the process of an incoming Interest packet at an intermediate router.

4.1 Static cost scheme

The Static cost scheme assumes that each time an off-path interest is initiated the TFB value is decreased by a static and predetermined value b, regardless of the popularity of the requested content or the "freshness" (*i.e.*, position in the EFIB) of the EFIB entry. This implies that an Interest initiated at a router that is h hops away from the AS egress node (towards the content origin) and assigned a TFB value equal to $t = h + a$, can initiate along the path up to $\left\lfloor \frac{t}{b} \right\rfloor$ off-path interests. Of course a viable strategy would have been to initiate only $\left\lfloor \frac{a}{b} \right\rfloor$ off-path interests and save enough budget to reach the content origin. Whereas the actual number of interests

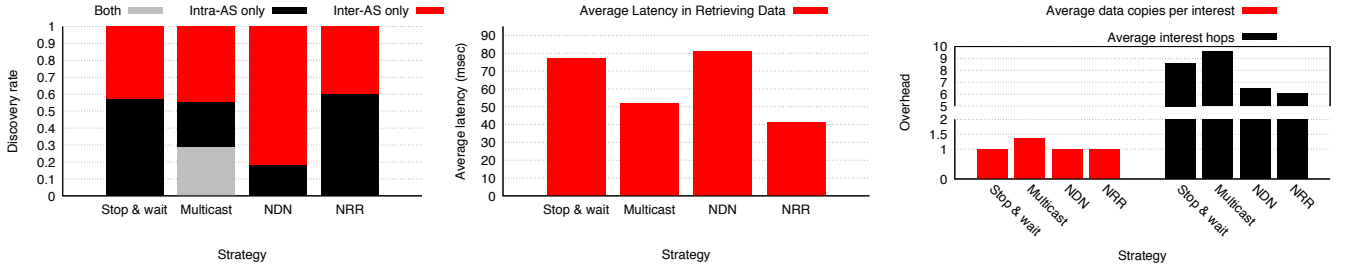

Figure 4: Performance of Stop-and-wait, Multicast, Nearest Replica Routing (NRR) and NDN forwarding strategies.

is not controlled and depends on the EFIB matching entries found along the path, this simple mechanism keeps under control the corresponding overhead. In order to eliminate coordination between the routers we assume here that TFB value is predetermined for each router. However, an out of band coordination scheme can be used to take into consideration both the topological characteristics and the position of a router in the network, as well as the popularity of a requested content to set the TFB in the most appropriate value (*e.g.*, a least popular item might require more off path interests –higher TFB value– to be found and vice versa). However, due to space limitations we leave this option for future investigation.

Note that in this paper and without loss of generality we assume that each intermediate node can initialize only up to one off-path interest, by selecting the "freshest" (*i.e.*, the one that is higher in the EFIB) matching EFIB entry[2]. This is a design choice that does not affect the functionality of the content discovery mechanism (*i.e.*, the maximum number of off-path interest is further reduced to $\left\lfloor \frac{h}{b} \right\rfloor$) and other schemes can also be applied. For instance a feedback mechanism that provides the cost/congestion in retrieving a Data packet from a particular direction/interface can be used so as to choose every time the least expensive/congested option, load balancing the network as well. Also, we assume that once an off-path interest has been initialized it can follow a trail of EFIB entries without any budget constraints. In other words, we only "charge" the initialization of off-path interests and the attempt of an Interest packet to reach the inter-AS content origin.

4.2 Dynamic cost scheme

Despite the effectiveness and simplicity of the Static cost described above, even very small TFB values (*i.e.*, shortest path length plus one or two extra quotas) can still generate significant overhead and duplicate responses, both from within the AS and from the content origin (see Evaluation section). This motivated us to design a Dynamic cost charging scheme that incorporates both the spatial and temporal popularity of the requested content.

Inspired by the popular *additive-increase/multiplicative-decrease (AIMD)* algorithm used in TCP congestion control we assume that a different cost is applied to TFB according to the position of the matching entry in the EFIB. EFIB uses an LRU replacement mechanism and an entry found "higher" in that table implies that the corresponding items was cached earlier in the past than another entry found "lower" at that table (*i.e.*, we always choose the highest/freshest matching entry to follow).

In the proposed Dynamic cost scheme each index i of the EFIB of a router r (and not cached value of the corresponding index) is initially assigned a default charging cost, b_i^r. Every time an off-path interest is initiated because a matching EFIB entry was found at that index i of router r, the charging cost for the index i is increased by an additive value d_i^r. Any follow up off-path interest initiated by an EFIB entry found at the same router and the same index will further increase by d_i^r the corresponding cost.

In NDN a router that forwards (or initiates) an interest (on-path or off-path) is on the reverse delivery path of any matching Data packet. An off-path interest searches opportunistically for cached content and will either retrieve matching data or will be discarded, as described in Section 3.4. In order to keep track of the success of the off-path interests, we assume that a router where the followed EFIB trails terminate will either respond with a matching Data packet (as in NDN) or send backwards a NACK (*i.e.*, an empty Data packet carrying only the Content Name of the failed off-path interest). Whenever a NACK is responded, an *invalidation mechanism* is triggered, where an intermediate router (i) will look in its EFIB for the particular Content Name and will decrease the charging cost of the matching EFIB index by a multiplicative parameter e_i^r ,before forwarding further the NACK (*i.e.*, following the established PIT entries), and (ii) the corresponding EFIB entry will be removed (*i.e.*, the EFIB entries below that will be shifted one position higher)[3].

Note that the new index i might be different from the index where the matching EFIB entry was found and triggered the off-path interest. However, here we couple the proposed cost with the index and not the actual value (*i.e.*, Content Name) of the EFIB entry. An approach where the cost will be associated with the actual Content Name might have been more efficient, but would have required each router to keep track of a cost for the whole content population, probably depleting the routers storage capacity.

In the Dynamic cost scheme, the initialization of an off-path interest can decrease more drastically the TFB value and the initial Interest packet can be dropped due to lack of budget sooner than in the Static cost scheme. For instance, an entry found at the top indexes of an EFIB can potentially consume the whole budget at once. This means that in the recent past, entries found at those indexes managed to retrieve cached content (that's why the increased cost) and there is no need to further search in the AS or out of it. Note

[2]We store a single face along with a name in an EFIB entry, which makes it possible to sort multiple faces matching the same name by freshness (*i.e.*, entry closest to the head among the ones with the same name). The lookup procedure uses <name, face> tuple as a search key, which needs to be performed for each outgoing face with a given name. Alternatively, the EFIB can be implemented using a Linked Hash Map (a hash map mapping to a double-linked list), which provides an already sorted list of the interfaces for each data name.

[3]Another alternative proactive approach, with higher communication overhead, is for routers to initialize NACKs in order to erase EFIB entries with every cache removal, *i.e.*, the cache may initiate EFIB entry erasure after content disposal.

that in the Dynamic cost scheme there might be cases where the charging cost of an EFIB entry might be significantly larger than the TFB value, due to multiple additive increases. For those cases an off-path interest can be initialized as long as the TFB value is larger than zero regardless the actual cost of the corresponding EFIB entry. Afterwards, the TFB value is set to "0" and no further off-path interests are allowed.

Another consideration with the Dynamic cost scheme is the possibility of very small costs. In particular, entries in the EFIB—typically the ones close to the tail of the table (*i.e.*, least recently used ones)—can have very small costs close to zero. This results from subsequent application of multiplicative decreases. This phenomenon can lead to high overhead, as a result of over-generating off-path interests. Therefore, we set a lower-bound on the used cost and set it to be the same value as the initially assigned default charging cost, b_i^r.

5 PERFORMANCE EVALUATION

In this section, we evaluate the performance of the budget-based multicast content discovery strategies based on a wide range of parameters. The objective is to evaluate their performance in terms of intra-AS content retrieval success, latency of content retrieval, and incurred overhead. Next we describe the setup of our evaluations, before presenting the experiments in the remaining sections.

5.1 Evaluation Setup and Metrics

For the evaluation of the proposed opportunistic content discovery mechanism, we used the Icarus simulator [18]—a Python-based discrete-event simulator for ICNs. Because Icarus, by design, is not bound to any specific ICN architecture, we made extensions to the simulator to support the NDN architecture with the new functionalities described in Section 3 and Section 4[4].

We use the Tiscali (AS 3257) network topology in the Rocketfuel dataset [24]. This topology has 161 routers and 328 bidirectional links and covers the European continent. We consider a scenario where the content item population of interest is 10^6 items. Note that 10^6 items is not meant to represent the current Internet content space. Instead, this set consists of the items that an ISP-operated Content Delivery Service [27] can store in its network. We also assume that each content item has the same size to simplify the cache size provisioning in the experiments.

In our experiments, we assume a warm-up period of two hours during which the EFIB table entries are populated. This period is followed by a two-hour observation period during which we measure the performance of the proposed strategies. An average of 100 interests are issued per second from users at exponentially spaced time points (the corresponding value was extracted from the analysis of extended video on demand traces in [25]). In order to simulate the issued interests originating from users, a single user agent is attached to each router of the Tiscali topology, and one agent is chosen uniformly at random for each issued interest.

Out of the 161 routers of the Tiscali topology, we select 16 of the lowest-degree nodes as the border routers of the AS and attach the content origins to those nodes. We associate each content item with exactly one border router, and set the FIB tables of the intra-AS routers accordingly.

Table 1: Default evaluation parameters.

Parameter	Value
Number of nodes V	161
Size of content population C	10^6
Aggregate network storage size S	$0.3 \cdot C$
EFIB size of each node	$128 \cdot \frac{S}{V}$
Average interest rate per second	100
Zipf exponent (content popularity)	0.7
Additive increase component d_i^r	1.0
Multiplicative decrease component e_i^r	2.0
Static Cost b	1.0
TFB for Static and Dynamic costs	$h + 2$

Our evaluation is based on the following metrics:

- **Discovery rate** (in ratio of issued interests): We partition the percentage of successful interests, *i.e.*, interests that are able to discover content before running out of forwarding budget, by the location of content retrieval. In particular, an interest discovers content either through an *intra-AS* path, an *inter-AS* path, or possibly *both*[5].
- **Latency** (in milliseconds): This metric measures the average round-trip time latency in retrieving content per successful interest. The latency metric ignores the processing delays at each node such as cache lookup and transmission delays, and only accounts for the propagation delays for the intra-AS links that are supplied by the Rocketfuel dataset. For inter-AS content retrievals, we incorporate an additional latency between 30 and 70 msec to the overall round-trip delay of the intra-AS latency based on the work of [6, 10]. This additional latency represents the propagation delay of the inter-AS path to the content provider that is typically few domain-level hops far away from the originating AS.
- **Overhead**: We use two overhead metrics. The first is the *average number of data copies* produced per issued (and satisfied) user interest. The ideal value of this metric is one, since the retrieval of extra data copies for an issued interest is redundant. The proposed discovery scheme may result with either multiple intra-AS retrievals or a combination of one or more intra-AS together with one inter-AS retrieval (*i.e.*, the one from the content origin). The second overhead metric is the *average interest hops*, which measures the number of hops that all Interest packets (on-path and off-path) travel within the AS per user issued interest, and reflects the processing overhead of interests within the AS. As such the Overhead metric includes both intra and inter-AS redundant network traffic.

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

[5]The Discovery rate of the evaluation plots is only for the original requests. A failed/dropped interest that is re-issued, always retrieve content from the corresponding content origin (*i.e.*, enabling the invalidation mechanism presented in Section 4.2).

In the next sections, we present the results of our experiments where we measure the performance of the budget-based strategies with the Static and the Dynamic cost as discussed in Section 4 for various system parameters and we compare them with the theoretically optimal NRR strategy defined in Section 3.4.

5.2 Impact of EFIB capacity

In Figure 5, we depict the impact of the EFIB capacity in the performance of the examined forwarding strategies. We express the EFIB capacity as a multiple of the cache size (*i.e.*, the CS router component). For instance, an EFIB capacity of two means that each router has twice as many "slots" in its EFIB as its cache. As mentioned before, the entire EFIB can be stored in the CS of each router and will typically take a negligible amount of space: a single slot (*i.e.*, entry) in EFIB store is merely a name, an outgoing face, and a cost value. In Table 1, we depict the default system parameters used throughout the Evaluation section.

As mentioned in Section 3.1, the EFIB follows an LRU replacement policy, and as such, its size affects the performance of the proposed forwarding strategies, since small EFIB size means that a recently inserted trail for a content may be replaced (*i.e.*, deleted) rather quickly even when the cache (at the end of the trail) still holds the corresponding content in its storage.

The leftmost plot in Figure 5 demonstrates the content discovery performance of the proposed strategies in percentage of issued interests. We observe that the portion of the issued interests finding content within the AS (*i.e.*, black region) increases rapidly for the Dynamic cost strategy until the EFIB size gets equal to 128 times the cache size. For larger EFIB sizes, the intra-domain discovery rate of both strategies remain roughly constant (we use this value for the EFIB size, *i.e.*, 128 times the CS size, in the rest of the experiments).

We observe for both strategies that an increasing percentage of user issued interests obtains multiple copies of content simultaneously through intra-AS and inter-AS paths (*i.e.*, gray region) as EFIB values increase from 2 to 128. Obviously, it is not desirable for an AS to forward an interest off the domain (following FIB entries) to retrieve content through an expensive inter-AS link when an off-path copy of the interest discovers the same content within the AS. In the rest of the discussion, we refer to such traffic as the *unnecessary inter-AS traffic*, which is shown with the gray region (*i.e.*, labeled "Both").

In terms of minimizing such unnecessary inter-AS traffic, the Dynamic cost strategy performs significantly better than the Static one. The Dynamic cost strategy limits the interests leading to such unwanted traffic at around 10.8%, which reaches its peak value at EFIB size of 128. In the Static cost strategy, on the other hand, the percentage of interests that lead to unnecessary inter-AS content retrievals reach nearly 30%. This is also reflected in the overhead metric, where the Dynamic cost strategy retrieves on average smaller number of data copies than the Static cost for larger EFIB sizes, as can be seen on the rightmost plot of Figure 5.

We observe from the leftmost plot in Figure 5 that for some interests the Dynamic cost strategy fails to retrieve content for larger EFIB sizes. This is due to the strategy spending the forwarding budget entirely mid-path (*i.e.*, within the AS) and failing to retrieve content from off-path trails. This happens when the cost of a matching index in an EFIB entry becomes large with multiple subsequent additions of d_i^r without multiplicative decrease as a result of recent success in usage, but fails this time to find a content along the off-path trail. In case all the issued off-path interests fail to retrieve cached content a user has to reissue the interest, which merely will reach the content origin. The usage of the invalidation mechanism through the multiplicative decrease of the Dynamic cost at each failure and the removal of the corresponding EFIB entries along the followed path (see Section 4.2), allows an interest to save enough budget to get out of the AS without special signaling at the Interest packet or any centrally controlled mechanism. The extra retrieval latency of any re-issued interest has been taken into account in the latency plots, but the Discovery rate plot (as mentioned earlier) only depicts the performance of the original interests. However, all users will be merely satisfied after re-issuing the corresponding failed interest.

For the Dynamic cost strategy, the percentage of interests running out of budget stays below 8% for the large EFIB sizes. Overall, the Dynamic cost achieves good performance and limits the amount of unnecessary inter-AS traffic at below 10% for the default used EFIB size (*i.e.*, gray area). Its performance is also comparable to the theoretically optimal strategy (*i.e.*, NRR), especially in terms of latency and intra-AS discovery rate for the selected default EFIB size.

In these set of experiments, we use a network cache size of 0.3, which means that the entire AS has the capacity to store 30% of the overall content population. In the next section, we investigate the impact of the AS storage capacity in the performance of the proposed content discovery strategies.

5.3 Impact of AS Cache Capacity

In Figure 6, we depict the impact of the overall network storage capacity in the performance of the examined budget-based multicast content discovery strategies. The network storage capacity of the entire network is expressed as the fraction of the content population. We evenly distribute the total network storage capacity at the routers of the network and leave more sophisticated distribution of AS storage capacity for future investigation.

From the leftmost plot of Figure 6, we observe that the Dynamic cost strategy becomes increasingly successful in reducing unnecessary inter-AS flows with increasing network storage capacity. For storage capacity of 0.8, this strategy limits such flows to below 4%. At the same time, the Dynamic cost strategy achieves an increasingly higher intra-AS discovery rate (*i.e.*, black region) for increasing storage capacity. This results indicate that the Dynamic cost become increasingly accurate in estimating the cost of trails; that is, the off-path trails with high cost have also very small percentage of failure to discover content.

On the other hand, the Static cost strategy leads to increasingly larger amount of unnecessary inter-AS flows with increasing network storage. In fact, the percentage of such flows reach above 40% for network storage capacity of 0.9 times the content population. We conclude that the interests with Static cost strategy fail to spend their TFB (*i.e.*, total forwarding budget) along the intra-AS path leading to the AS egress node. Such interest packets leave the AS to retrieve content even when an off-path interest successfully discovers content within the AS.

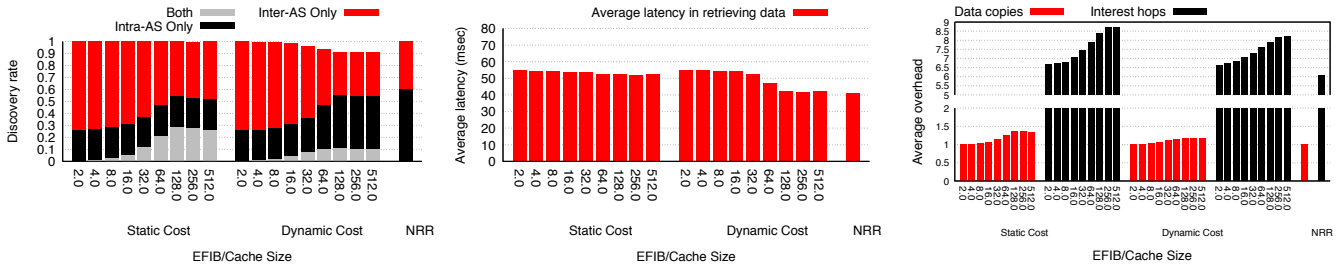

Figure 5: The impact of the EFIB capacity in the performance of the examined forwarding strategies.

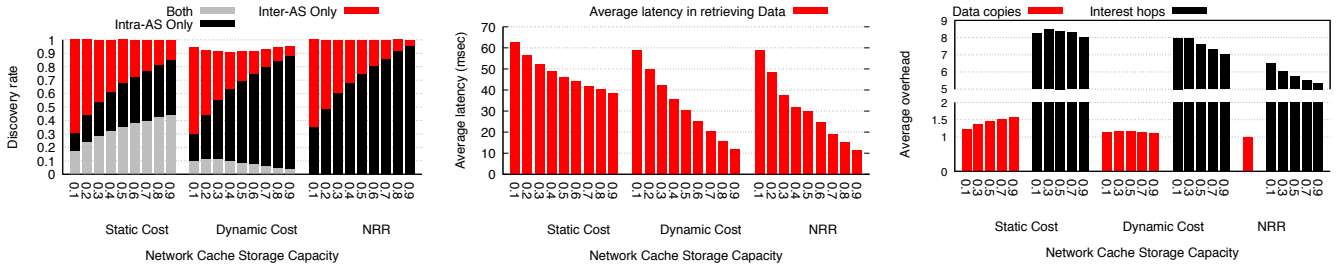

Figure 6: The impact of the overall network storage capacity in the performance of the examined forwarding strategies.

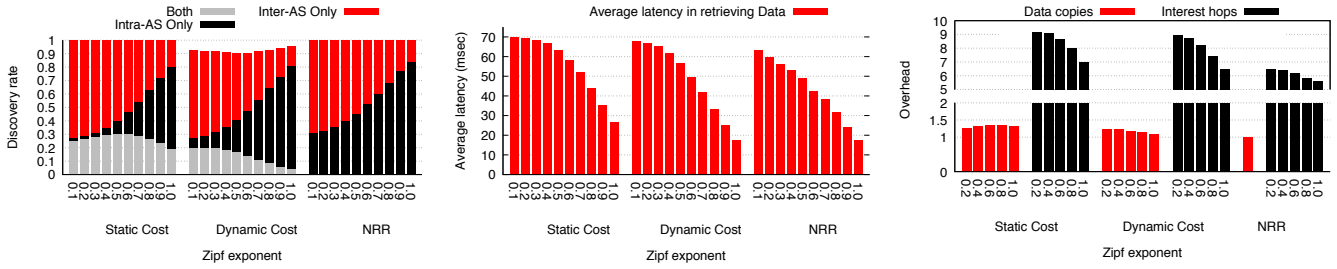

Figure 7: The impact of the popularity distribution in the performance of the examined forwarding strategies.

Despite an increasing amount of unnecessary flows, the Static cost strategy retrieves an increasingly larger percent of content within the AS, similar to the Dynamic cost strategy. As a result, both strategies achieve nearly optimal latency in retrieving content. On the other hand, the overhead of Static cost strategy is significantly higher than the Dynamic cost one. This is the result of unnecessarily many off-path trails that are followed by the interests in Static cost strategy.

5.4 Impact of Content Popularity

In the above scenarios, we used a default Zipf exponent value of 0.7 when determining the items' popularity. Measurement-based studies, such as [4] suggest that the Zipf exponent for web traffic lies in the range of $0.64 - 0.84$, while other types of traffic (*e.g.*, P2P or video) may follow different popularity patterns [5, 25].

In Figure 7, we depict the impact of content popularity in the performance of the proposed strategies. We observe from the leftmost plot that the lower values of Zipf exponent leads to higher percent of unnecessary inter-AS flows in both strategies; particularly in the case of the Static cost strategy. This is because of the low locality of reference in the content requested by the interests, which leads to higher replacement rate in the intra-AS caches. At high replacement rates, even the Dynamic cost strategy fails to determine accurate costs for its EFIB entries. As Zipf exponent increases beyond 0.4, the amount of unnecessary flows rapidly decrease for the Dynamic cost strategy. Overall, the Dynamic cost strategy achieves higher intra-AS discovery rate than the Static cost, and as a result, lower latency in retrieving content.

5.5 Impact of Forwarding Budget

As mentioned in Section 4, the initial total forwarding budget (TFB) of each Interest packet is set to a predetermined value. In the above experiments this value is set to $h + 2$, where h is the shortest path length in hops to reach the egress node towards the content origin. Also, the on-path cost and off-path (per-hop) cost is set to one (*i.e.*, $b = 1$ in Section 4.1 and $b_i^r = d_i^r = 1$, $e_i^r = 2$ for every router r and EFIB index i in Section 4.2).

In Figure 8, we depict the performance of the proposed strategies for different forwarding budgets. The x-axis in these plots represents the additional (*i.e.*, extra) budget value a in TFB, and we consider a values of one to four. From the leftmost plot, we observe that increasing the TFB value beyond two has negligible impact on the discovery performance of the Static cost strategy. The Static cost strategy's indifference to extra budget is because of the relatively high success rate 0.57 of off-path trails and the small number (1.89 on average) of matching EFIB encounters (on the

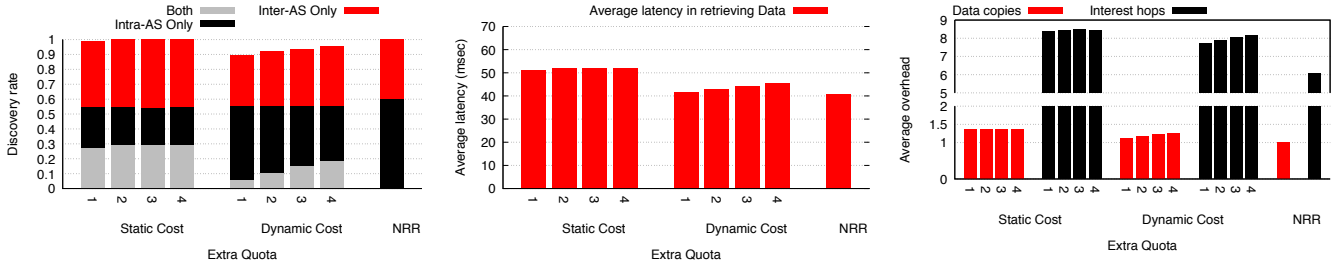

Figure 8: The impact of the forwarding budget in the performance of the examined forwarding strategies.

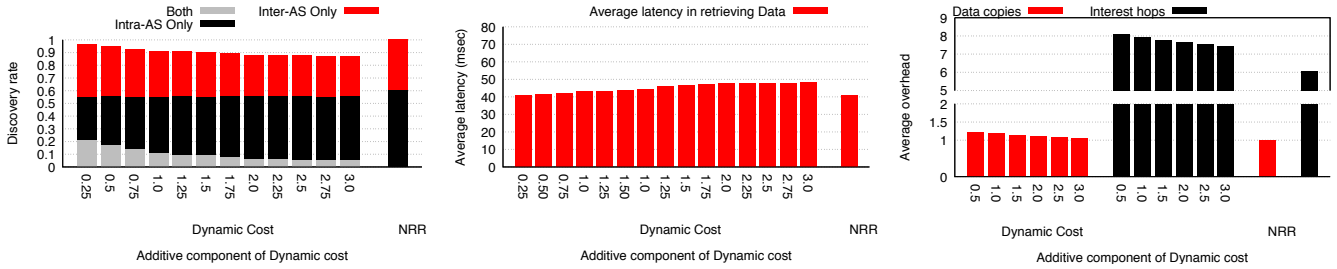

Figure 9: The impact of the additive component in the performance of the Dynamic cost strategy.

path to the egress node) for our default settings. Therefore, raising the TFB beyond $h + 2$ has little impact, since the majority of the interests typically discover items within two off-path trails and do not encounter other matching off-path trails.

On the other hand, a large TFB value increases the likelihood of retrieving content within the AS as well as the amount of unnecessary flows (gray region) for the Dynamic cost strategy. In order to further understand this trend, we looked into statistics on the average cost spent on successful and failing off-path trails. Particularly, a failing off-path trail spends on average a cost of 2.44 budget quotas, and a successful off-path trail spends on average 7.8, as opposed to the constant cost of 1.0 for both cases in the Static cost counterpart. This implies that adding more budget allows the Dynamic cost strategy to issue more off-path interests as opposed to the Static one. This inevitably will increase both the overall content discovery rate, as well as the overhead traffic.

Another dimension of performance is the off-path versus on-path discoveries by the strategies. We observe that the percentage of off-path discoveries in Dynamic cost strategy is significantly higher than the Static cost. For extra budget value of one, Dynamic cost retrieves 38.3% of the requested content from off-path trails, while the Static cost retrieves 26.8% of them. This corresponds also to a difference in the average latency of content retrieval, shown in the middle plot of Figure 8. In particular, the Dynamic cost strategy can retrieve closer off-path copies more often and achieve a lower retrieval latency, whereas in terms of overhead, the performance of the two strategies are very similar.

5.6 Impact of the Additive Component

Finally, we present results on the impact of the additive component (*i.e.*, d_i^r in Section 4.2) of the Dynamic cost strategy in Figure 9. We observe that increasing the additive component reduces the overall discovery rate and at the same time reduces the amount of unnecessary flows, as expected. The latency of content retrieval also increases with increasing the additive parameter because of the additional latency incurred by the failed interests, and subsequently the re-issuing of them by the users.

6 CONCLUSIONS

We have proposed a content discovery mechanism, which coordinates forwarding and caching opportunistically without using signaling or update protocols. The mechanism establishes ephemeral forwarding state along the content retrieval path to locate locally stored content within an AS. This mechanism allows content retrieved through an inter-AS link to be stored exactly once along the delivery path, by leveraging the ephemeral state to direct future interests towards the stored content. The ephemeral forwarding state is stored in an Ephemeral FIB (EFIB) table and maintained as an LRU cache using a very small portion of the Content Store capacity.

We have investigated several multicast budget-based forwarding strategies that effectively use and manage ephemeral forwarding state together with the existing FIB state to discover content locally, and thus reduce the expensive inter-AS content retrievals. The proposed forwarding strategies improve NDN's intra-AS discovery performance by more than 300% and incurs negligible overhead, achieving comparable performance to the optimal (Nearest Replica Routing) forwarding strategy.

ACKNOWLEDGMENTS

This work has been supported by the EC H2020 UMOBILE project (GA no. 645124), the EU-JAPAN initiative EC H2020 ICN2020 project (GA no. 723014) and the EPSRC INSP Early Career Fellowship (EP/M003787/1).

REFERENCES

[1] Andrea Araldo, Dario Rossi, and Fabio Martignon. 2016. Cost-Aware Caching: Caching More (Costly Items) for Less (ISPs Operational Expenditures). *IEEE Trans. Parallel Distrib. Syst.* 27, 5 (May 2016), 1316–1330.

[2] Onur Ascigil, Vasilis Sourlas, Ioannis Psaras, and George Pavlou. 2016. Opportunistic Off-Path Content Discovery in Information-Centric Networks. In *22nd IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. IEEE.

[3] Suzan Bayhan, Liang Wang, Jörg Ott, Jussi Kangasharju, Arjuna Sathiaseelan, and Jon Crowcroft. 2016. On Content Indexing for Off-Path Caching in Information-Centric NetworksâĢď. In *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 102–111.

[4] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. 1999. Web caching and Zipf-like distributions: Evidence and implications. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Vol. 1. IEEE, 126–134.

[5] György Dán and Niklas Carlsson. 2010. Power-law revisited: large scale measurement study of P2P content popularity.. In *IPTPS*. 12.

[6] Qiang Duan. 2010. *Delay Characteristics of Packet Switched Networks*. Springer US, Boston, MA, 203–223.

[7] Suyong Eum, Kiyohide Nakauchi, Masayuki Murata, Yozo Shoji, and Nozomu Nishinaga. 2012. CATT: Potential Based Routing with Content Caching for ICN. In *ICN Workshop on Information-centric Networking (ICN '12)*. ACM, New York, NY, USA, 49–54.

[8] Shuo Guo, Haiyong Xie, and Guangyu Shi. 2012. Collaborative Forwarding and Caching in Content Centric Networks. In *Proceedings of the 11th International IFIP TC 6 Conference on Networking - Volume Part I (IFIP'12)*. 41–55.

[9] A K M Mahmudul 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 (ICN '13)*. 15–20.

[10] B. Huffaker, M. Fomenkov, D. Plummer, D. Moore, and k. claffy. 2002. Distance Metrics in the Internet. In *IEEE International Telecommunications Symposium (ITS)*. IEEE, Brazil, 200–202.

[11] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. 2009. Networking Named Content. In *International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*. ACM, New York, NY, USA, 1–12.

[12] Munyoung Lee, Kideok Cho, Kunwoo Park, T. Kwon, and Yanghee Choi. 2011. SCAN: Scalable Content Routing for Content-Aware Networking. In *Communications (ICC), 2011 IEEE International Conference on*. 1–5.

[13] Ioannis Psaras, Wei Koong Chai, and George Pavlou. 2012. Probabilistic in-network caching for information-centric networks. In *Proceedings of the second edition of the ICN workshop on Information-centric networking*. ACM, 55–60.

[14] Ioannis Psaras, Konstantinos V Katsaros, Lorenzo Saino, and George Pavlou. 2015. Lira: A location independent routing layer based on source-provided ephemeral names. *arXiv preprint arXiv:1509.05589* (2015).

[15] E.J. Rosensweig and J. Kurose. 2009. Breadcrumbs: Efficient, Best-Effort Content Location in Cache Networks. In *INFOCOM 2009, IEEE*. 2631–2635.

[16] Giuseppe Rossini and Dario Rossi. 2014. Coupling Caching and Forwarding: Benefits, Analysis, and Implementation. In *Proceedings of the 1st International Conference on Information-centric Networking (INC '14)*. ACM, New York, NY, USA, 127–136.

[17] Lorenzo Saino, Ioannis Psaras, and George Pavlou. 2013. Hash-routing Schemes for Information Centric Networking. In *3rd ACM SIGCOMM workshop on Information Centric Networking (ICN'13)*. ACM.

[18] 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.

[19] Vasilis Sourlas, Paris Flegkas, and Leandros Tassiulas. 2014. A novel cache aware routing scheme for Information-Centric Networks. *Computer Networks* 59 (2014), 44 – 61.

[20] Vasilis Sourlas, Lazaros Gkatzikis, Paris Flegkas, and Leandros Tassiulas. 2013. Distributed cache management in information-centric networks. *IEEE Transactions on Network and Service Management* 10, 3 (2013), 286–299.

[21] V. Sourlas, L. Gkatzikis, P. Flegkas, and L. Tassiulas. 2013. Distributed Cache Management in Information-Centric Networks. *IEEE Transactions on Network and Service Management* 10, 3 (2013), 286–299.

[22] Vasilis Sourlas, Ioannis Psaras, Lorenzo Saino, and George Pavlou. 2016. Efficient hash-routing and domain clustering techniques for information-centric networks. *Computer Networks* 103 (2016), 67–83.

[23] V. Sourlas, L. Tassiulas, I. Psaras, and G. Pavlou. 2015. Information resilience through user-assisted caching in disruptive Content-Centric Networks. In *IFIP Networking Conference (IFIP Networking), 2015*. 1–9.

[24] 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.

[25] Yi Sun, Seyed Kaveh Fayaz, Yang Guo, Vyas Sekar, Yun Jin, Mohamed Ali Kaafar, and Steve Uhlig. 2014. Trace-driven analysis of ICN caching algorithms on video-on-demand workloads. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 363–376.

[26] Dirk Trossen and G. Parisis. 2012. Designing and realizing an information-centric internet. *Communications Magazine, IEEE* 50, 7 (July 2012), 60–67.

[27] Daphne Tuncer, Vasilis Sourlas, Marinos Charalambides, Maxim Claeys, Jeroen Famaey, George Pavlou, and Filip De Turck. 2016. Scalable cache management for ISP-operated content delivery services. *IEEE Journal on Selected Areas in Communications* 34, 8 (2016), 2063–2076.

[28] 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 International Conference on Information-Centric Networking*. ACM, 9–18.

[29] Yaogong Wang, Kyunghan Lee, B. Venkataraman, R.L. Shamanna, Injong Rhee, and Sunhee Yang. 2012. Advertising cached contents in the control plane: Necessity and feasibility. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*. 286–291.

[30] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, and others. 2014. Named data networking. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 66–73.