AN INFORMATION-CENTRIC NETWORKING ARCHITECTURE WITH SMALL ROUTING TABLES

J.J. Garcia-Luna-Aceves and Maziar Mirzazad Barijough
Computer Science and Engineering Department, University of California, Santa Cruz, CA 95064, USA

NOTE: Corresponding author: J.J. Garcia-Luna-Aceves, jj@soe.ucsc.edu

Abstract – *The basic design of the Named Data Networking (NDN) architecture is shown to incur problems, in that Interests (content requests) may go unanswered even if content is available in the network, and Pending Interest Tables (PIT) are shown to provide limited performance benefits in the presence of in- network caching. A new approach to content-centric networking is introduced that eliminates the need to maintain PITs while providing the benefits sought by NDN. Content-Centric Networking with Data Answer Routing Table (CCN-DART) replaces PITs with Data Answer Routing Tables (DARTs) to forward Interests that do not state their sources. The size of a DART is proportional to the number of routes used by Interests traversing a router, rather than the number of Interests traversing a router. It is shown that undetected Interest loops cannot occur in CCN-DART, and that Interests and responses to them are forwarded correctly independently of the state of the network. The results of simulation experiments comparing CCN-DART with NDN using the ndnSIM simulation tool show that CCN-DART attains similar or better latencies than NDN when no looping problems occur in NDN, while using a similar number of Interests and storing an order of magnitude fewer forwarding entries.*

Keywords – Addressing, content routing, information-centric networks

1. INTRODUCTION

Several architectures for Information-Centric Networking (ICN) have been proposed to improve the performance and the end-user experience of the Internet [1, 38]. The leading approach in ICN is based on the use of content requests called Interests, and can be characterized as *Interest-based*. This approach consists of: populating Forwarding Information Bases (FIB) maintained by routers with routes to name prefixes denoting content, sending Interests for specific Content Objects (CO) over paths implied by the FIBs, and delivering data packets with content objects along the reverse paths traversed by Interests. The main advantages that Interest-based content-centric networking offers compared to the IP Internet are that: (a) content providers and caching sites do not know the identity of the consumers requesting content; (b) content can be obtained by name from those sites that are closer to consumers; (c) data packets carrying content cannot traverse loops, because they are sent over the reverse paths traversed by Interests; and (d) content-oriented security mechanisms can be implemented as part of the content delivery mechanisms.

Today, Named Data Networking (NDN) [28] is arguably the most prominent Interest-based ICN approach. However, several results have been reported regarding the large PIT sizes required for NDN to operate at Internet scale [10, 31, 32, 33]. Furthermore, the vulnerability of NDN (and per-Interest forwarding state in general) to Interest flooding attacks has been studied by a number of authors [3, 21, 33, 36, 37]. As a result, a number of ICN approaches have been advanced aimed at allowing ICN to be deployable at Internet scale.

One type of approach consists of embedding ICN in the Internet Protocol (IP) in order to allow the incremental deployment of ICN. The methods used to attain this include encapsulation and tunneling. An attractive representative example of this type of approach is the hybrid-ICN (hICN) architecture [44], which supports ICN while transparently interconnecting hICN routers with standard IP routers and enabling the forwarding of hICN packets as normal IP packets. The hICN architecture accomplishes this by using hICN names that consist of a name prefix and a name suffix, with name prefixes being routable addresses.

Another type of approach consist of implementing ICN functionality at the application or transport layers, without making any modifications to the Internet routing infrastructure other than the establishment of overlays. Application-layer solutions implemented in the past are exemplified by Peer-to-Peer (P2P) applications and over-lays [46] and Content Delivery Networks (CDN) [47, 48, 49, 50], which are widely used today. Named-Data Trans- port (NDT) [51] is a recent example of enabling ICN functionality at the transport layer. The advantage of this approach over P2P applications, CDNs, and other ICN architectures is that NDT does not require the deployment of overlays of name resolvers or content routers, or changes to the Internet routing infrastructure to operate. However, NDT requires the introduction of a new transport protocol, the Named-Data Transport Protocol (NDTP) that replaces TCP, and modifications to the Domain Name System (DNS) that together map content names to one or multiple locations where the content is offered, deliver the content reliably to consumers from content servers or transparent caches without using end-to-end connections, and enforce consumer privacy.

This paper focuses on NDN as the representative approach of Interest-based ICN architectures. Since the introduction of Content-Centric Networking (CCN) [23], the following assumptions have been made in several Interest-based ICN architectures and NDN in particular [8, 28, 40]:

- Interest loops can be detected by identifying each Interest by the name of the requested content and a nonce assigned to the Interest with a very low probability of collisions.
- Per-Interest forwarding state maintained in Pending Interest Tables (PITs) is required for Interests and responses to them to be forwarded without revealing the identities of the sources of Interests.
- Interests requesting the same content need to be aggregated in the PITs to attain efficiency.

This paper shows that these assumptions need not hold, which impacts the potential adoption of content-centric networking at Internet scale. Section 2 addresses the impact of Interest aggregation in NDN. First, it shows that Interest aggregation in NDN can lead to Interests not being answered when they traverse routing loops, even if the requested content is available in the network. Second, it shows that Interest aggregation is ineffective when in-network caching is used. This is done by means of simulation experiments ran using the implementation of NDN in ndnSIM [2] without modifications. The experiments assume networks with on-path caching and average Round-Trip Times (RTTs) that are representative of recent IP latency statistics. The results show that the percentage of Interests that are aggregated and the performance benefits derived from Interest aggregation are negligible even when the capacity of in-network caches is small compared to the number of published content objects.

There has been recent work aimed at improving the forwarding performance of NDN [45] and it has been argued that NDN should be used only at the edge of the Internet. However, the complexity of NDN forwarding merits the discussion of ICN forwarding approaches that are inherently more efficient or cheaper to implement, and which could further take advantage of edge deployments in which content caching can be implemented in just a few nodes rather than along forwarding paths [**9**]. Section 3 introduces Content-Centric Networking with Data Answer Routing Table **CCN-DARTs**, which replaces PITs with Data Answer Routing Tables (DARTs). A DART stores forwarding state regarding the routes traversing the router, rather than the Interests forwarded by the router. An Interest in CCN-DART states the name of the requested content, a hop count, and a destination-and-return token (*dart*). The hop count is used to avoid forwarding loops. The dart leaves a trace of the path traversed by the Interest using local identifiers of the previous hop and the current hop, without revealing the identity of the source of the Interest. CCN-DART is a simpler approach to CCN based on DARTs we introduced recently [19, 54] and takes advantage of the loop-free Interest-forwarding mechanisms we have proved to be correct [17, 18, 20]. CCN-DART is inherently more efficient and a less vulnerable alternative for content-centric networking than NDN because it eliminates the need to maintain a per-Interest forwarding state.

Section 4 proves that no forwarding loops can occur in CCN-DART and that responses to Interests are forwarded correctly. Section 5 compares the performance of CCN-DART with NDN when routes to name prefixes are loop-free and static, and either on-path caching or edge caching is used in a 200-router network. CCN-DART attains similar or better end-to-end latencies and incurs very similar Interest traffic than NDN to retrieve content, even though Interests are not aggregated in CCN-DART. However, at high Interest rates, NDN requires 10 to 20 times the number of forwarding entries needed in CCN-DART.

2. INTEREST AGGREGATION IN NDN

2.1 Elements of NDN operation

Routers in NDN use Interests, data packets, and Negative Acknowledgments (NACKs) to exchange content. An Interest is identified in NDN by the name of the CO requested and a nonce created by the origin of the Interest. A data packet includes the CO name, a security payload, and the payload itself. A NACK carries the information needed to denote an Interest and a code stating the reason for the response. We denote the name of CO j by $n(j)$, and the name prefix stored in a FIB that includes that CO name by $n(j)^*$. In the context of NDN, we use $I[n(j), id_j(s)]$ to denote an Interest that requests CO with name $n(j)$ and that is assigned nonce $id_j(s)$. A data packet sent in response to an Interest $I[n(j), id_j(s)]$, denoted $DP[n(j), id_j(s), sp(j)]$, states the name and nonce of the Interest, a security payload $sp(j)$ used to validate the content object, and the object itself.

Three data structures are used by a given router r to process Interests, data packets, and NACKs: A content store (CS^r), a forwarding information base (FIB^r), and a pending Interest table (PIT^r).

CS^r is a cache for COs indexed by their names. With on-path caching, routers cache the content they receive in response to Interests they forward.

FIB^r is populated using content routing protocols [16, 22] or static routes and router r matches Interest names stating a specific CO $n(j)$ to FIB^r entries of prefix names using *longest prefix match*. The FIB entry for a given name prefix lists the interfaces that can be used to reach the prefix. In NDN, a FIB entry also contains additional information [28].

PITs are used in NDN to determine the interfaces to which data packets or NACKs should be sent back in response to Interests, allow Interests to not disclose their sources, and enable Interest aggregation. The entry in PIT^i for CO with name $n(j)$ is denoted by $PI^i_{n(j)}$ and consists of one or multiple tuples stating a nonce received in an Interest for the NDO and the incoming interface where it was received, and a list of the outgoing interfaces over which the Interest was forwarded.

When a router receives an Interest, it checks whether there is a match in its CS for the CO requested in the Interest. The Interest matching mechanisms used can vary, and for simplicity we focus on exact Interest matching only. If a match to the Interest is found, the router sends back a data packet over the reverse path traversed by the Interest. If no match is found in the CS, the router determines whether the PIT stores an entry for the same content.

In NDN, if the Interest states a nonce that differs from those stored in the PIT entry for the requested content, then the router "aggregates" the Interest by adding the incoming interface from which the Interest was received and the nonce to the PIT entry without forwarding the Interest. If the same nonce in the Interest is already listed in the PIT entry for the requested CO, the router sends a NACK over the reverse path traversed by the Interest.

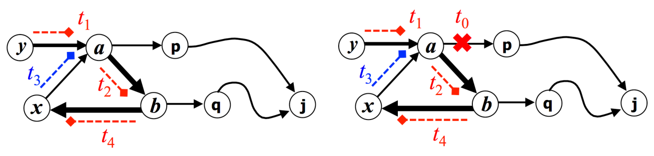
If a router does not find a match in its CS and PIT, the router forwards the Interest along a route (or routes) listed in its FIB for the best prefix match. In NDN, a router can select an interface to forward an Interest if it is known that it can bring content and its performance is ranked higher than other interfaces that can also bring content. The ranking of interfaces is done by a router independently of other routers based on information obtained through probing or the control plane [40].

2.2 Undetected Interest loops in NDN

The key aspect of the current forwarding strategy in NDN (see NDN Packet Format Specification 0.3) [53] is that a router determines whether or not an Interest is a duplicate Interest based solely on the content name and a nonce. Packet forwarding loops may occur even when routers update their FIBs based on loop-free routing protocols, and hence Interests may loop independently of the routing protocol used in the control plane. To discuss the correctness of this approach, we define an Interest loop as follows.

Interest loop: An Interest loop of h hops for a CO with name $n(j)$ occurs when one or more Interests asking for $n(j)$ are forwarded and aggregated by routers along a cycle $L = \{v_1, v_2, ..., v_h, v_1\}$ such that router v_k receives an Interest for CO $n(j)$ from v_{k-1} while waiting for a response to the Interest it has forwarded to v_{k+1} for the same NDO, with $1 \leq k \leq h$, $v_{h+1} = v_1$, and $v_0 = v_h$. According to the NDN forwarding strategy, a router can select a neighbor to forward an Interest if it is known that it can bring content and its performance is ranked higher than other neighbors that can also bring content. The ranking of neighbors is done by a router independently of other routers, which can result in long-term routing loops implied by the FIBs if the routing protocol used in the control plane does not guarantee instantaneous loop freedom (e.g., NLSR [**22**]).

Fig. 1 illustrates Interest looping in NDN. Arrowheads in the figure indicate the next hops to content advertised by router j according to the FIB entries stored in routers. Thick lines indicate that the perceived performance of a neighbor is better than neighbors shown with thinner lines. Dashed lines indicate the traversal of Interests over links and paths. The time when an event is processed at a router is indicated by t_i.



(a) Long-term loop (b) Temporary loop

Fig. 1 – Undetected Interest loops in NDN [**20**]

Fig. 1(a) shows the case of a long-term Interest loop caused by multipaths implied in FIBs not being loop-free, even though all routing tables are consistent. In this case, the ranking of interfaces in a FIB can be such that a path with a larger hop count may be ranked higher than a path with a smaller hop count, because of the perceived performance of the interfaces or paths towards prefixes. Fig. 2(b) shows the case of a temporary Interest loop when single-path routing is used and FIBs are inconsistent due to a topology change at time t_1 (link (a, p) fails).

In both cases, router a aggregates the Interest from x and router x aggregates the Interest from y, and the combined steps preclude the detection of any Interest looping. In this example, it would appear that the looping problems could be avoided by forcing router b to use q rather than x for Interests regarding prefixes for which router j is an origin. However, the same looping problems would exist even if link (b, q) were removed in the example, and the ways in which FIBs are populated and interfaces are ranked are independent of updates made to PITs.

Theorem 1 below proves that the NDN forwarding strategy specified in [23, 40, 43] cannot ensure that Interest loops are detected when Interests are aggregated, even if nonces were to denote Interests uniquely. The theorem assumes that all messages are sent correctly and that no routing-table changes occur to show that the NDN forwarding strategy can fail to return any content or NACK in response to Interests independently of network dynamics. Furthermore, Theorem 2 shows that no correct forwarding strategy can be defined that allows Interest aggregation and attempts Interest-loop detection by the matching of Interest-identification data (e.g., CO names, nonces, or the path traversed by the Interest). We have published these results [17, 18] before, together with simulation results illustrating the negative impact of undetected Interest loops in NDN.

Theorem 1: Interest loops can go undetected in a stable, error-free network in which NDN or CCN is used, even if nonces were to denote Interests uniquely.

Proof Consider the NDN or CCN forwarding strategy running in a network in which no two nonces created by different nodes for the same content are equal, all transmissions are received correctly, and no topology or routing-table changes occur after time t_0. Let $LT^{v_k}(I[n(j),\, id_j(s)])$ denote the lifetime of $I[n(j),\, id_j(s)]$ at router v_k.

Assume that Interests may traverse loops when they are forwarded according to the forwarding strategy, and let a loop $L = \{v_1, v_2, ..., v_h, v_1\}$ exist for NDO j, and let Interest $I[n(j),\, id_j(x)]$ start traversing the chain of nodes $\{v_1, v_2, ..., v_k\} \in L$ (with $1 < k < h$) at time $t_1 > t_0$.

Assume that $I[n(j), id_j(x)]$ reaches router v_k at time $t_3 > t_1$ and that router v_k forwards Interest $I[n(j), id_j(y)]$ to its next hop $v_{k+1} \in L$ at time t_2, where $t_1 \leq t_2 < t_3$, $id_j(x) \neq id_j(y)$, and v_{k+1} may be v_1.

According to the Interest processing strategy in NDN and CCN, router v_k creates an entry in its PIT for $I[n(j), id_j(y)]$ at time t_2, and perceives any Interest for name $n(j)$ and a nonce different than $id_j(y)$ received after time t_2, and before its PIT entry for $I[n(j), id_j(y)]$ is erased, as a subsequent Interest.

Let $|t_2 - t_3| < LT^{v_k}(I[n(j), id_j(y)])$ when router v_k receives $I[n(j), id_j(x)]$ from router $v_{k-1} \in L$ at time t_3, where $1 < k - 1$. According to the Interest processing strategy in NDN and CCN, router v_k must treat $I[n(j), id_j(x)]$ as a subsequent Interest for content $n(j)$ that is aggregated, because v_k is waiting for $DP[n(j), id_j(y), sp(j)]$ at time t_3.

Because of the existence of L, Interest $I[n(j), id_j(y)]$ must be forwarded from v_k to v_1. Let t_4 denote the time when $I[n(j), id_j(y)]$ reaches v_1, where $t_4 > t_2 \geq t_1$, and assume that $|t_1 - t_4| < LT^{v_1}(I[n(j), id_j(x)])$. According to NDN's Interest processing strategy, v_1 must treat $I[n(j), id_j(y)]$ as a subsequent Interest, because it is waiting for $DP[n(j), id_j(x), sp(j)]$ at time t_4.

Given the Interest aggregation carried out by nodes v_k and v_1, nodes in the chain $\{v_1, v_2, ..., v_{k-1}\} \in L$ process only $I[n(j), id_j(x)]$, nodes in the chain $\{v_{k+1}, v_{k+2}, ..., v_h\} \in L$ process only $I[n(j), id_j(y)]$, and no Interest loop detection can take place. Therefore, no content can be submitted in response to $I[n(j), id_j(x)]$ and $I[n(j), id_j(y)]$.

Similar results to Theorem 1 can be proven for NDN and the original CCN operating in a network in which routing tables are inconsistent as a result of network or content dynamics. In this case, Interest loops can go undetected even if the control plane supports only single-path forwarding of Interests.

Let Interest-identification data refer to such information carried in Interests as nonces, unique identifiers, or a combination of nonces and the path traversed by an Interest.

Theorem 2: No correct forwarding strategy exists with Interest aggregation and Interest loop detection based on the matching of Interest-identification data.

Proof: Assume any forwarding strategy in which a router remembers an Interest it has forwarded as long as necessary to detect Interest loops, and detects the occurrence of an Interest loop by matching the Interest-identification data carried in an Interest it receives with the Interest-identification data used in the Interest it forwarded previously asking for the same content. Let $I[n(j), id_j(s)]$ denote the Interest asking for $n(j)$ with Interest-identification data $id_j(s)$ created by router s.

Assume that an Interest loop $L = \{v_1, v_2, ..., v_h, v_1\}$ for NDO with name $n(j)$ exists in a network using the forwarding strategy. Let Interest $I[n(j), id_j(x)]$ start traversing the chain of nodes $\{v_1, v_2, ..., v_k\} \in L$ (with $1 < k < h$) at time t_1.

Assume that $I[n(j), id_j(x)]$ reaches router v_k at time $t_3 > t_1$ and that router v_k forwards Interest $I[n(j), id_j(y)]$ to its next hop $v_{k+1} \in L$ at time t_2, where $t_1 \leq t_2 < t_3$, $id_j(x) \neq id_j(y)$. Let $I[n(j), id_j(y)]$ traverse the chain of nodes $\{v_k, v_{k+1}, ..., v_1\} \in L$, reaching v_1 at time t_4, where $t_4 > t_2 \geq t_1$.

By assumption, Interest aggregation occurs, and hence v_k aggregates $I[n(j), id_j(x)]$ at time t_3, and v_1 aggregates $I[n(j), id_j(y)]$ at time t_4. Therefore, independently of the amount of information contained in $id_j(x)$ and $id_j(y)$, v_1 cannot receive $I[n(j),\, id_j(x)]$ from v_h and v_k cannot receive $I[n(j),\; id_j(y)]$ from v_{k-1}. It thus follows that no node in L can successfully use the matching of Interest-identification data to detect that Interests for $n(j)$ are being sent and aggregated along L and the theorem is true.

The results in theorems 1 and 2 can also be proven by mapping the Interest processing strategy of NDN, and any forwarding strategy that attempts to detect Interest loops by matching Interest-identification data, to the problem of distributed termination detection over a cycle, where Interests serve as the tokens of the algorithm [11, 27]. Because Interest aggregation erases a token traversing the ring (Interest loop) when any node in the ring has previously created a different token, correct termination detection over the ring (i.e., Interest loop detection) cannot be guaranteed in the presence of Interest aggregation.

Based on the results in theorems 1 and 2, we have proposed a simple modification to the NDN forwarding strategy called CCN-ELF [20]. However, it has not been adopted in NDN, and the few alternative solutions to CCN- ELF that have been proposed in the past for NDN (e.g.,[52]) have not been proven to be correct, rely on resolving loops, and are still based on the same NDN forwarding strategy using Interest identification, which Theorem 2 shows to be insufficient to prevent Interest looping. Ob- viously, a loop traversed by an Interest can be detected easily if each Interest is identified with the route it should traverse. This is easy to implement but requires routers in the network to have complete topology information (e.g.,[22, 30, 34]) or at least path information or partial topol- ogy information (e.g., [5, 30]). However, there is no need for using nonces to detect Interest loops and, more importantly, a source route reveals the identity of the source router requesting content and hence defeats one of the key objectives of NDN.

Another view of the problem would be to say that Interest aggregation is not common and hence undetected Interest loops should be too rare to cause major performance problems. However, if Interests need to be aggregated only rarely, then the very existence of PITs should be questioned. The following section addresses the need for Interest aggregation in the presence of in-network caching.

2.3 Performance benefits of Interest aggregation

We analyze the impact of interest aggregation on the performance of NDN using simulations carried out with the ndnSIM simulation tool [2]. We used the implementation of NDN in ndnSIM without modifications. Our study is independent of the Interest retransmission strategy. For simplicity, we assume that routers use exact Interest matching to decide whether an Interest can be answered. We consider the percentage of aggregated Interests in the network and the average number of PIT entries created per second per router as the performance metrics.

2.3.1 Scenario parameters and scenarios

The simulation parameters we consider include the average latencies between routers, the storage capacity of caches, the Interest request rates from routers, the popularity of content, and the temporal correlation of content requests.

The scenarios we use consist of random networks with 200 nodes corresponding to routers distributed uniformly in a 100m$\times$100m area. Routers with 12m or a shorter distance are connected to each other with a point- to-point link, which results in a topology with 1786 edges. Each router acts as a producer of content and also has lo- cal consumers generating Interests.

Producers are assumed to publish 1,000,000 different content objects that are uniformly distributed among routers. For simplicity, we assume that *all* routers have the same storage capacity in their caches, which depending on the experiment ranges from 0 to up to 100,000 cache entries per router, or 10% of the published objects.

The distribution of object requests determines how many Interests from different users request the same content. It has been argued [13, 14] that Internet traffic follows a Zipf distribution with a parameter (α) value close to 1. A smaller Zipf parameter value results in a lower Interest aggregation amount. Accordingly, we model object popularity using a Zipf distribution with values of α equal to 0.7 and 1.

We considered different values of the *total Interest rate per router*, corresponding to the sum of Interests from all local users. Increasing values of Interest rates can be viewed as higher request rates from a constant user population of local active users per router, or an increasing population of active users per router. For example, 50 to 500 Interests per second per router can be just 10 Interests per second per active user for a local population of 5 to 50 concurrently active users per router. The Interest rates we assume per router are not large compared to re- cent results on the size that PITs would have in realistic settings [10, 32, 3, 33].

The percentage of Interests that benefit from Interest aggregation is a function of the time elapsed from the time when a router receives an Interest, until the time the requested CO or a NACK is received by that router. In turn, this time is a function of the RTT between the router originating the Interest and the site with the requested content, as well as the PIT entry expiration time when the Interest is not answered with a data packet or a NACK.

Recent Internet latency statistics [4, 7] show that Internet traffic latency varies from 11ms for regional European traffic to 160ms for long-distance traffic. Accordingly, we consider point-to-point delays of 10ms between neighbor routers in many of our simulations, which leads to RTTs of about 200ms. We also carried our experiments varying the RTT of the network below and above 200ms.

2.3.2 Simulation results

The following simulation results can be viewed as applicable to the steady-state behavior of a network using NDN.

Fig. 2 shows the percentage of aggregated Interests for four different total request rates per router from 50 to 500

Interests per second, and seven different storage capacities of the caches, ranging from 100 to 10,000 objects (i.e., up to 1% of the published objects). The Zipf parameter value assumed is $\alpha = 1$, which is the best case for Interest aggregation.

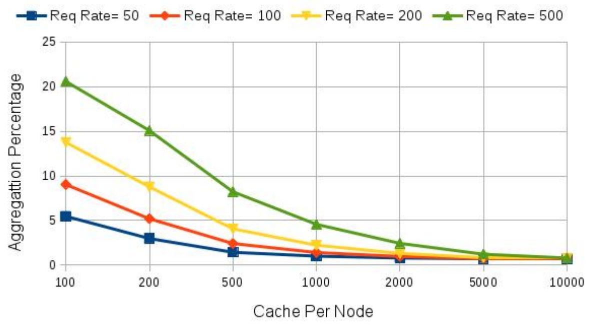


Fig. 2 – Interest aggregation as a function of the storage capacity per content store ($\alpha = 1$).

The impact of in-network caching on Interest aggregation is very clear. Interest aggregation is useful when caches have insignificant capacities and request rates are high. However, as the capacity of a cache in each router increases, the percentage of Interests that are aggregated drops quickly. The percentage of aggregated Interests drops to less than 5% for a request rate per router of 500 requests per second when a cache can store only 0.1% of the published objects.

Fig. 3 shows the effect of the α parameter and RTTs when the total request rate per router is only 50 Interests per second. The latencies between neighbor routers are set to 5 and 15 ms, which produce RTTs of 66 to 70 ms and 193 to 200 ms, respectively. It is clear that Interest aggregation is far less important when consumers are less likely to request similar content ($\alpha = 0.7$). It is also clear that the benefits of Interest aggregation vanish as caches are allowed to cache more content. When caches can store up to 1% of the total number of objects published, the percentage of Interests that are aggregated is less than 2% for $\alpha = 1$ and less than 0.8% for $\alpha = 0.7$.

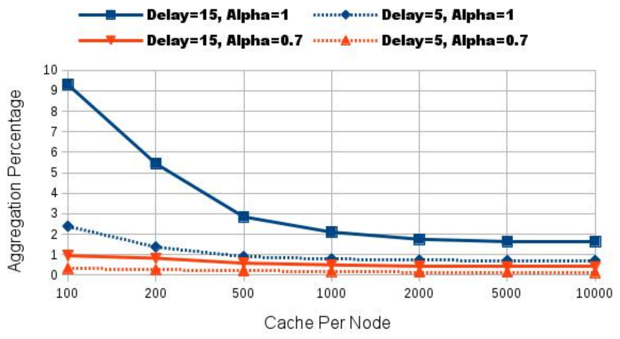


Fig. 3 – Interest aggregation as a function of the values of Zipf parameter, storage capacity, and RTTs

Fig. 4 shows the average number and variance of PIT entries created per second per router with total requests rates per router varying from 50 Interests per second to 500 Interests per second. Results are presented for six different values of cache capacity ranging from no caching to 10% of the published objects. In all cases, the average number of PIT entries created grows proportionally with the request rate per router. The large variance indicates that only some routers benefit from Interest aggregation. Furthermore, orders of magnitude increases in the storage capacity of content stores do not produce a similar reduction in the number of PIT entries created per second per router, which is a function of the total request rates after caches are large enough.

In theory, Interest aggregation is most useful when Interests exhibit temporal correlation, such as when popular live events take place. Fig. 5 shows the impact of caching on Interest aggregation when Interests have temporal correlation and either no caching is used or caches with capacity for 1000 objects are used (only 0.1% of total objects published in the network). Localized Interests are generated using the model proposed by Dabirmoghaddam et al [9] with a Zipf parameter value of $\alpha = 0.7$ and results for three total Interest rates per router and four temporal localization factors for Interests are shown. A higher temporal locality factor indicates a higher degree of popularity of objects in the same time period. The results in Fig. 5 show that, without caching, Interest aggregation *is* very important for all values of temporal locality of Interest popularity, and is more important when Interest locality is high (large localization factor). However, once caching is allowed and even if caches can store only up to 0.1% of the published objects, the percentage of aggregated Interests is minuscule and actually decreases with the temporal correlation of Interests. This experi- ment further illustrates the overlapping nature of PITs and caches in NDN.

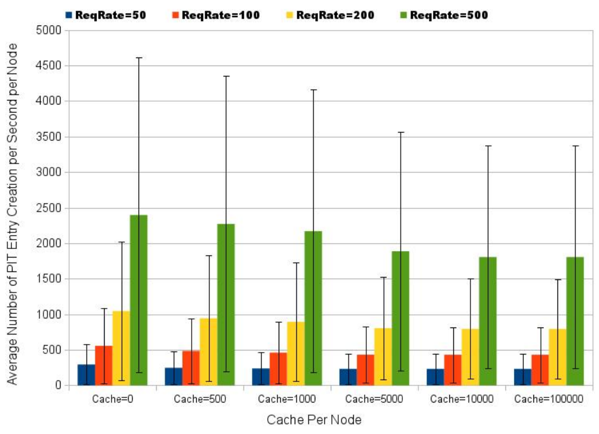


Fig. 4 – Average number of PIT entries created per second at each router

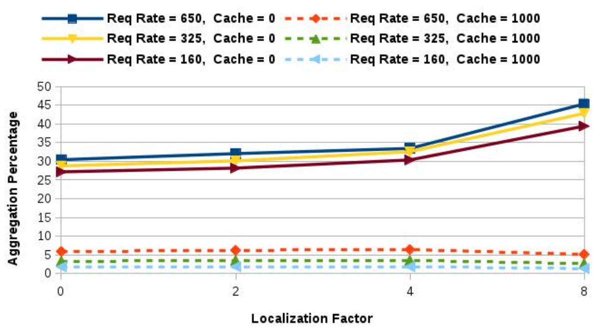


Fig. 5 – Impact of caching on the aggregation of Interests with temporal locality

3. CCN-DART

3.1 Design rationale and assumptions

The design of CCN-DART is based on three observations. First, the results in Section 2 show that content caching, which is essential for content-centric networking, makes the occurrence of Interest aggregation extremely rare an obviates the need for PITs as a means of reducing the number of Interests being forwarded. In-network caching tends to make popular content available locally before subsequent requests for the same content arrive. This is important, because using PITs comes at a big price. PITs enable users to mount Interest flooding attacks aimed at overflowing PITs [3, 33, 36, 37], and the storage overhead they incur is significant [10, 31, 32].

Second, the number of routers in a network is orders of magnitude smaller than the number of COs accessed through them. Hence, maintaining forwarding state based on the *routes* going through a router, each used by many Interests, is by nature orders of magnitude smaller than forwarding state based on the *Interests* traversing a router.

Third, as we have proven in the previous section, preventing Interests from traversing loops when Interest aggregation is allowed cannot be attained by identifying Interests uniquely using names of content objects and nonces. However, it can be done based on an ordering of the routers that forward the Interests, and content routing protocols [16, 22] can provide all the needed information to attain proper ordering in the forwarding plane.

We make the following assumptions in the description of CCN-DART, none of which should be considered design requirements for the basic approach we introduce.

Interests are retransmitted only by the users that originated them, rather than routers that relay Interests. Of course, "local repair" mechanisms can be used in CCN-DART to react more quickly to congestion or topology changes.

We assume that routers use exact Interest matching. Given the name of a CO, a router can determine whether or not the exact same CO is stored locally.

The COs corresponding to a name prefix could be stored in subsets of the prefix at multiple sites, with each site announcing that the entire prefix is local. However, resolving an Interest for a given CO in this case would require contacting all the sites where the prefix is local. For simplicity, in our description of CCN-DART we assume that a router that announces being an origin of a name prefix stores all the COs in that prefix locally, and call such a router an *anchor* of the prefix. If all the COs of a prefix are mirrored at multiple sites, each router connected to the site storing the COs is an anchor of the prefix.

Routers know which interfaces are neighbor routers and which are local consumers, and forward Interests on a best-effort basis. For convenience, a request for content from a local user is sent to its router in the form of an Interest stating an empty hop count to content and an empty dart.

3.2 Information exchanged and stored

CCN-DART uses Interests, NACKs and data packets to support content exchange among routers. Our description of these messages addresses only that information needed to attain correct forwarding, which consists of the names of COs, the hop counts to prefixes, and *destination-and-return tokens* (**darts**). The terms neighbor and interface are used interchangeably. The name of CO j is denoted by $n(j)$, the name prefix that is the best match for $n(j)$ in a FIB is denoted by $n(j)^*$, and $S^i_{n(j)^*}$ denotes the set of neighbors of router i considered to be next hops to pre- fix $n(j)^*$. Darts are local identifiers used to uniquely denote routes established between source and destination routers over which Interests, data packets, and NACKs are sent. Accordingly, darts can be very small (e.g., 32 bits).

An Interest forwarded by router k requesting CO j is denoted by $I[n(j), h^I(k), dart^I(k)]$. It states the name of the requested CO $(n(j))$, the hop count $(h^I(k))$ from k to prefix $n(j)^*$, and the dart $(dart^I(k))$ used to establish an anonymous route back to the router that originates the Interest.

A data packet sent in response to an Interest is denoted by $DP[n(j), sp(j), dart^I(i)]$, and states the name of the CO requested in the Interest being answered $(n(j))$, a security payload $(sp(j))$ used optionally to validate the con- tent object, the dart $(dart^I(i))$ from the Interest being answered, and the CO itself.

A NACK to an Interest is denoted by $NA[n(j)$, CODE, $dart^I(i)]$ and states the name of the CO $(n(j))$, a code (CODE) indicating the reason why the NACK is sent, and the dart $(dart^I(i))$ from the Interest being answered.

Reasons for sending a NACK include: an Interest loop is detected, no route is found towards requested content, and no content is found.

Router i maintains three tables: Forwarding Information Base (FIB^i) , a Data-Answer Routing Table ($DART^i$) , and an optional Requested-Content Table(RCT^i) . All routers must maintain FIBs and DARTs, and only those routers with local users and routers supporting content caching need to maintain an RCT.

A *predecessor* of router i for Interests related to name pre- fix $n(j)^*$ is a router that forwards Interest for COs with names that are best matched by $n(j)^*$. Similarly, a *suc- cessor* of router i for Interests related to $n(j)^*$ is a router to whom router i forwards Interest regarding COs with names that are best matched by name prefix $n(j)^*$.

FIB^i is indexed using name prefixes. The entry for prefix $n(j)^*$ consists of a set of tuples, one for each next hop to prefix $n(j)^*$. The tuples for prefix $n(j)^*$ are ranked based on their utility for forwarding. As a minimum, the tuple for next hop $q \in S^i_{n(j)^*}$ states:

1. $h(i, n(j)^*, q)$: The distance to $n(j)^*$ through q.

2. $a(i, n(j)^*, q)$: The nearest anchor of $n(j)^*$ through q.

$DART^i$ stores the mappings of predecessors to successors along loop-free paths to name prefixes. The entry created for Interests received from router p (predecessor) and forwarded to router s (successor) towards a given anchor a of a name prefix is denoted by $DART^i(a, p)$ and specifies:

1. $a^i(a, p)$: The anchor a for which the entry is set.
2. $p^i(a, p)$: The predecessor p of the path to $a^i(a, p)$.
3. $pd^i(a, p)$: The *predecessor dart*, which equals the dart received in Interests from p forwarded towards $a^i(a, p)$.
4. $s^i(a, p)$: The successor s selected by router i to forward Interests received from p towards $a^i(a, p)$.
5. $sd^i(a, p)$: The *successor dart* included in Interests sent by router i towards $a^i(a, p)$ through the successor.
6. $h^i(a, p)$: The hop-count distance to prefix a through successor s when i establishes the DART entry.

DART entries can be removed using a least-recently used policy or a maximum lifetime, for example. An entry in a DART can remain in storage for long periods of time in the absence of topology changes. The removal of a DART entry simply causes a router to compute a new entry for Interests flowing towards an anchor of prefixes.

RCT^i serves as an index of local content as well as local requests for remote content. It is indexed by the CO names that have been requested by the router. The entry for CO name $n(j)$ states the name of the CO ($n(j)$), a pointer to the local storage where the CO ($p[n(j)]$) is stored, and a list of zero or more identifiers of local consumers ($lc[n(j)]$) that have requested the CO. The RCT could be implemented as two separate indexes, one for local content and one for requests for remote content.

If router i is an anchor of name prefix $n(j)^*$ then it stores all the COs with names that match the name prefix. This is denoted by $n(j)^* \in RCT^i$. If CO $n(j)$ has been requested by one ore more local consumers and no copy of the CO is yet available, then $n(j) \in RCT^i$, $p[n(j)] = nil$, and $lc[n(j)] \neq \phi$. On the other hand, if router i caches CO $n(j)$, then $n(j) \in RCT^i$, $p[n(j)] \neq nil$, and $lc[n(j)] = \phi$.

3.3 Preventing forwarding loops

We have shown [17, 18] that undetected Interest loops can occur in NDN when Interests are aggregated while traversing routing loops resulting from inconsistencies in FIB entries or inconsistent rankings of routes at different routers. CCN-DART uses the same approach we proposed for the elimination of undetected Interest loops in the context of NDN [18, 20] to prevent forwarding loops when DART entries are created.

Dart Entry Addition Rule (DEAR:
Router i accepts $I[n(j), h^I(k), dart^I(k)]$ from router k and creates a DART entry for prefix $n(j)^*$ with k as its predecessor and a router $v \neq k$ as its successor if:

$$\exists\, v \in S^i_{n(j)^*} (\, h^I(k) > h(i, n(j)^*, v)\,)$$

The distance information that must be stored in the FIBs to implement DEAR can be obtained easily from the control plane. Such content routing protocols as DCR [16] and NLSR [22] are able to compute the required minimum-hop distances, which can then be copied into the FIBs.

Figures 6(a) and (b) illustrate how using DEAR prevents Interests from traversing loops when a multipath routing protocol is used and FIB entries are consistent but local rankings of multiple routes available at each router (e.g., NLSR) cause routing loops. The pair of numbers next to a node in Fig. 6(a) indicate the hop count from that router to $n(j)^*$ over an interface and the ranking of the interface according to the FIB of the router.
Let the triplet (v, h, r) denote an interface, its hop count and its ranking. In Fig. 6(a), FIB^a states $(b, 4, 1)$, $(p, 4, 2)$, $(x, 6, 3)$, and $(y, 6, 4)$; FIB^b states $(x, 6, 1)$, $(a, 5, 2)$, and $(q, 3, 3)$; and FIB^x states $(a, 5, 2)$ and $(b, 5, 1)$. As Fig. 6(b) shows, router a receives $I[n(j), h^I(y) = 5, dart^I(y)]$ from router y at time t_1 and forwards $I[n(j), h^I(a) = 4$, $dart^I(a)]$ to b because $5 = h^I(y) > h(a, n(j)^*, b) = 4$ and b is ranked above p. Similarly, router b receives the Interest at time t_2 and accepts it because $4 = h^I(a) > h(b, n(j)^*, q) = 3$. Router b uses router q as the next hop for the Interest, because q is the highest ranked neighbor satisfying DEAR. This example illustrates that, independently of local rankings of multiple routes to prefixes,

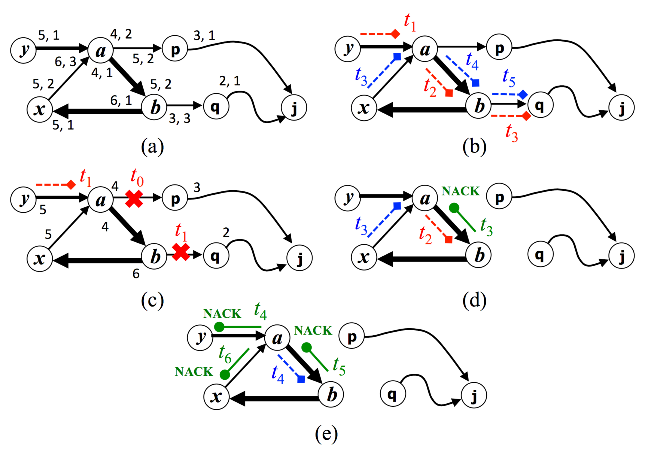


Fig. 6 – CCN-DART avoids forwarding loops [**54**]

Interests traverse simple paths by requiring each relaying router to satisfy DEAR.

Figures 6(c) to (e) illustrate how DEAR operates when FIBs are inconsistent due to topology changes. Routers a and b update their FIBs at times t_0 and t_1, respectively. We assume that the routing updates have not been processed at routers y and a when they forward Interests at times t_1 and t_2, respectively. As shown in Fig. 6(d), router b *must* send a NACK to router a because it does not have a neighbor with a shorter hop count to prefix $n(j)^*$ than $h^I(a) = 4$. In turn, router a forwards a NACK to router y, and the Interest from x also prompts a NACK from b because DEAR is not satisfied. Within a finite time after t_1, the FIBs of routers are updated to show that prefix $n(j)^*$ cannot be reached and Interests from local users for COs in that prefix cannot forwarded by routers a, b, x and y.

By contrast, assuming NDN in the same example results in the Interests sent by y and x to be forwarded along the forwarding loop involving a, b and x. Router a aggregates the Interest from x, and router x aggregates the Interest from y. Those Interests must then “wait to infinity” in the PITs until their lifetimes expire or they are otherwise evicted from the PITs. Using nonces in Interests incurs considerable PIT storage overhead.

Similar rules based on more sophisticated lexicographic orderings could be defined based on the same general approach stated in DEAR. The requirement for such forwarding rules is that more information needs to be maintained in the FIBs, such as distance values to name prefixes that take into account such factors as end-to-end delay, reliability, cost, or bandwidth available.

DEAR is very similar to sufficient conditions for loop-free routing introduced in the past, in particular sufficient conditions for loop-free routing based on diffusing computations [15, 34, 42]. Indeed, DEAR can be viewed as a case of termination detection based on diffusing computations [11]. The difference between DEAR and loop-free routing protocols based on diffusing computations is that DEAR operates in the data plane using existing FIB entries, while routing protocols operate in the control plane using flooding to build routing tables and hence FIB entries.

It should be pointed out that, because DEAR is not *necessary* to avoid forwarding loops, there are cases in which DEAR is not satisfied even though no forwarding loops exist. However, prior results on multipath routing based on diffusing computations [41] indicate that this does not constitute a performance problem. Given the speed with which FIBs are updated to reflect correct distances computed in the control plane, false loop detection using DEAR should be rare, and it is preferable by far than storing PIT entries that expire only after many seconds without receiving responses.

3.4 Maintaining forwarding state

Algorithms 1 to 4 specify the steps taken by routers to process and forward Interests, data packets, and NACKs. The algorithms we present assume that the control plane updates FIB^i to reflect any changes in hop counts to name prefixes and anchors resulting from the loss of connectivity to one or more neighbors or link-cost changes. In addition, a DART entry is silently deleted when connectivity with the successor or predecessor of the entry is lost, or it is not used for a prolonged period of time.

Algorithm 1 shows the steps taken by router i to process Interests received from local consumers. For convenience, content requests from local consumers are assumed to be Interests stating the name of a CO, together with an empty hop count to content and an empty dart.

Algorithm 1 Processing Interest from user c at router i

```
function Interest_Source
INPUT: $RCT^i, FIB^i, DART^i, I[n(j), nil, nil]$;
if $n(j) \in RCT^i$ then
   if $p[n(j)] \neq nil$ then
      retrieve CO $n(j)$; send $DP[n(j), sp(j), nil]$ to $c$
   else
      $lc[n(j)] = lc[n(j)] \cup c$; $p[n(j)] = nil$ (% Interest is aggregated)
   end if
else
   if $n(j)^* \in RCT^i$ then
      send $NA[n(j)$, no content, $nil]$ to $c$ (% $n(j)$ does not exist)
   else
      if $n(j)^* \notin FIB^i$ then
         send $NA[n(j)$, no route, $nil]$ to $c$ (% No route to $n(j)^*$ exists)
      else
         create entry for $n(j)$ in $RCT^i$: (% Interest from $c$ is recorded)
         $lc[n(j)] = lc[n(j)] \cup c$; $p[n(j)] = nil$;
         for each $v \in S^i_{n(j)^*}$ by rank in $FIB^i$ do
            $a = a(i, n(j)^*, v)$;
            if $\exists DART^i(a, i)\ (\ s^i(a, i) = v\ )$ then
               $h^I(i) = h^i(a, i)$; $dart^I(i) = sd^i(a, i)$;
               send $I[n(j), h^I(i), dart^I(i)]$ to $v$; return
            else
               create entry $DART^i(a, i)$:
               compute $SD \neq sd^i(p, q)\ \forall\ DART^i(p, q)$;
               $pd^i(a, i) = SD$; $sd^i(a, i) = SD$;
               $p^i(a, i) = i$; $s^i(a, i) = v$; $h^i(a, i) = h(i, n(j)^*, v)$;
               $h^I(i) = h^i(a, i)$; $dart^I(i) = sd^i(a, i)$;
               send $I[n(j), h^I(i), dart^I(i)]$ to $v$; return
            end if
         end for
      end if
   end if
end if
```

Router i first looks up its RCT to determine if the requested CO is stored locally or a request for the CO is pending. If the CO is stored locally, a data packet is sent back to the user requesting it. If a request for the same content is pending, the name of the user is added to the list of customers that have requested the CO. Router i sends back a NACK if it is an anchor of name prefix $n(j)^*$ and the specific CO is not found locally, or the CO is remote and no FIB entry exists for a name prefix that can match $n(j)$.

If possible, router i forwards the Interest through the highest ranked neighbor in its FIB for the name prefix matching $n(j)$. How such a ranking is done is left unspeci- fied, and can be based on a distributed or local algorithm.

If a DART entry exists for the selected successor that should receive the Interest, the existing route is used; otherwise, a new DART entry is created before the Interest is sent. The successor dart assigned to the new DART entry is a locally unique identifier that must be different from all other successor darts being used by router i.

Algorithm 2 outlines the processing of data packets. If the router has local consumers that requested the content, the data packet is sent to those consumers based on the information stored in RCT^i. If the data packet is received in response to an Interest that was forwarded from router k, router i forwards the data packet after swapping the successor dart received in the data packet for the pre- decessor dart stored in $DART^i$. Router i stores the data object if caching is supported.

Algorithm 2 Processing data packet at router i

```
function Data Packet_Handling
INPUT: $DART^i, RCT^i, DP[n(j), sp(j), dart^I(q)]$;
[o] verify $sp(j)$;
[o] if verification fails then discard $DP[n(j), sp(j), dart^I(q)]$
if $\exists DART^i(a,k)\ (\ dart^I(q) = sd^i(a,k) \wedge p^i(a,k) = i\ )$
  (% router $i$ was the origin of the Interest) then
    for each $c \in lc[n(j)]$ do
        send $DP[n(j), sp(j), nil]$ to $c$; $lc[n(j)] = lc[n(j)] - \{c\}$
    end for
end if
if $\exists DART^i(a,k)\ (\ dart^I(q) = sd^i(a,k) \wedge p^i(a,k) = k \in N^i\ )$ then
    (% Data packet can be forwarded to $k$:)
    $dart^I(i) = pd^i(a,k)$; send $DP[n(j), sp(j), dart^I(i)]$ to $k$
end if
[o] if no entry for $n(j)$ exists in $RCT^i$ then
    create $RCT^i$ entry for $n(j)$: $lc[n(j)] = \emptyset$
  end if
[o] store CO in local storage; $p[n(j)] =$ address of CO in local storage
```

Algorithm 3 states the steps taken to handle NACKs, which are similar to the forwarding steps taken after receiving a data packet. Router i forwards the NACK to local consumers if it was the origin of the Interest, or to a neighbor router k if it has a DART entry with a successor dart matching the dart stated in the NACK.

Algorithm 4 shows the steps taken by router i to process an Interest received from a neighbor router k. If the requested content is cached locally, a data packet is sent back. As in Algorithm 1, router i sends back a NACK if it is an anchor of $n(j)^*$ and the CO with name $n(j)$ is not found locally, or the CO is remote and no FIB entry exists for $n(j)^*$. In contrast to Algorithm 1, Interests received from other routers are not aggregated.

If the Interest must be forwarded and an entry exists in $DART^i$ with router k as the predecessor and $dart^I(k)$ as the predecessor dart, then DEAR has been satisfied by a previous Interest from k over the existing path that k is requesting to use. Accordingly, router i simply swaps $dart^I(k)$ for the successor dart stated in the entry in $DART^i$ and forwards the Interest.

Alternatively, if no DART entry exists with k as a predeces- sor and $dart^I(k)$ as the predecessor dart, router i tries to find a neighbor that satisfies DEAR. The highest-ranked router v satisfying DEAR is selected as the successor for the Interest and router i creates a successor dart diffe- rent from any other successor darts in $DART^i$, stores an entry with v and the new successor dart, and forwards the Interest to v. If DEAR is not satisfied, then router i sends a NACK back to router k.

Algorithm 3 Process NACK at router i

```
function NACK_Handling
INPUT: $DART^i, RCT^i, NA[n(j), \text{CODE}, dart^I(q)]$;
if $\exists DART^i(a,k)\ (\ dart^I(q) = sd^i(a,k) \wedge p^i(a,k) = i\ )$
  (% router $i$ was the origin of the Interest) then
    for each $c \in lc[n(j)]$ do
        send $NA[n(j), \text{CODE}, nil]$ to $c$; $lc[n(j)] = lc[n(j)] - \{c\}$
    end for
    delete entry for $n(j)$ in $RCT^i$
end if
if $\exists DART^i(a,k)\ (\ dart^I(q) = sd^i(a,k) \wedge p^i(a,k) = k \in N^i\ )$ then
    (% NACK can be forwarded to router $k$:)
    $dart^I(i) = pd^i(a,k)$; send $NA[n(j), \text{CODE}, dart^I(i)]$ to $k$
end if
```

Algorithm 4 Processing Interest from router k at router i

```
function Dart_Swapping
INPUT: $RCT^i, FIB^i, DART^i, I[n(j), h^I(k), dart^I(k)]$;
if $n(j) \in RCT^i \wedge p[n(j)] \neq nil$ then
    retrieve CO $n(j)$; send $DP[n(j), sp(j), dart^I(k)]$ to $k$
else
    (% $n(j) \notin RCT^i \vee p[n(j)] = nil$ )
    if $n(j)^* \in RCT^i$ then
        send $NA[n(j), \text{no content}, dart^I(k)]$ to $k$
    else
        if $n(j)^* \notin FIB^i$ then
            send $NA[n(j), \text{no route}, dart^I(k)]$ to $k$
        else
            if $\exists DART^i(a,k)\ (\ pd^i(a,k) = dart^I(k)\ )$ then
                $h^I(i) = h^i(a,k)$; $dart^I(i) = sd^i(a,k)$;
                send $I[n(j), h^I(i), dart^I(i)]$ to $s^i(a,k)$
            else
                for each $v \in S^i_{n(j)^*}$ by rank in $FIB^i$ do
                    if $h^I(k) > h(i, n(j)^*, v)$ (% DEAR is satisfied) then
                        $a = a(i, n(j)^*, v)$;
                        create entry $DART^i(a,k)$:
                        compute $SD \neq sd^i(p,q)\ \forall\ DART^i(p,q)$;
                        $h^i(a,k) = h(i, n(j)^*, v)$;
                        $p^i(a,k) = k$; $pd^i(a,k) = dart^I(k)$;
                        $s^i(a,k) = v$; $sd^i(a,k) = SD$;
                        create Interest:
                        $h^I(i) = h^i(a,k)$; $dart^I(i) = sd^i(a,k)$;
                        send $I[n(j), h^I(i), dart^I(i)]$ to $v$;
                        return
                    end if
                end for (% Interest may be traversing a loop)
                send $NA[n(j), \text{loop}, dart^I(k)]$ to $k$
            end if
        end if
    end if
end if
```

3.5 Supporting multipoint communication

NDN supports multicasting by the Reverse-Path Forwarding (RPF) of data packets over paths traversed by aggregated Interests. Interests serve the dual purpose of maintaining Multicast Forwarding Trees (MFTs) and pacing multicast sources. CCN-DART also supports multipoint communication using the RPF approach, but separates the establishment of an MFT from the mechanisms used to pace a source or disseminate multicast data over the tree.

CCN-DART uses RCTs and Multicast Data Answer Routing Tables (MDARTs) to maintain MFTs. A single dart is used to denote all the predecessors and successors in the MFT of a group at each router . This means that a single dart must be used to label all the branches of the MFT of a multicast group. The dart used for multicast group name $g(j)$ is denoted by $d[g(j)]$, and can be made part of the group name to simplify its dissemination.

A router with local receivers of a multicast group maintains the mapping of the names of local receivers to the name of the multicast group in its RCT. The MDART at router i is denoted by $MDART^i$ and is indexed by the names of the multicast groups for which the router forwards traffic. The entry for a multicast group with name $g(j)$ in $MDART^i$ states: the dart of the group ($d[g(j)]$), the successor selected by router i to join the group, the set of routers (predecessors) that requested to join $g(j)$ through router i, and the hop-count distance to the anchor of $g(j)$ when router i established the MDART entry for the group ($h^i(g(j))$).

If router i has local receivers for group $g(j)$, then it sends a Join Request (JR), denoted by $JR[g(j), h^J(i), dart^J(i)]$, stating the name of the group, $h^J(i) = h^i(g(j))$, and $dart^J(i) = d[g(j)]$. The forwarding of JRs is based on FIB entries and is similar to the forwarding of Interests. A relay router can forward a JR towards the anchor of $g(j)$ in two cases. If no MDART entry exists and DEAR is satisfied, an MDART entry is created for the group. If an MDART entry exists, then the router simply adds a new predecessor for the group in the existing MDART entry. Negative acknowledgments may be sent if no routes to $g(j)$ are found, DEAR is not satisfied, or MDART entries become invalid due to topology changes.

The design of algorithms for multicast data dissemination or pacing of multicast sources is outside the scope of this paper. However, similar approaches to those designed to address the ACK implosion problem in reliable multicasting [24] can be used to pace sources and pull or push data over multicast trees. Multicast data packets have the same format of data packets and are forwarded from sources towards receivers based on the darts of the multicast groups.

3.6 CCN-DART forwarding example

Fig. 7 illustrates how darts are used to label Interests and associate Interests with data packets and NACKs. In the example, routers a , b, and x have local consumers originating Interests, and those Interests are assumed to request COs with names that are best matched with name prefixes for which router d is an anchor.

The arrowheads in the links of the figure denote the next hops stored in the FIBs of routers, and $y(i)$ denotes the ith dart in $DART^y$. The figure shows the DART entries maintained at all routers for name prefixes for which router d is an anchor, and the RCT entries stored at routers a, b, and x. Consumers *A, C, N,* and *P* request the same CO with name $n(j)$, and router a aggregates their requests and needs to send only one Interest for $n(j)$ towards d. Similarly, it aggregates the Interests from consumers *A, C,* and *Q*. Similar Interest aggregation of local requests occur at routers b and x.

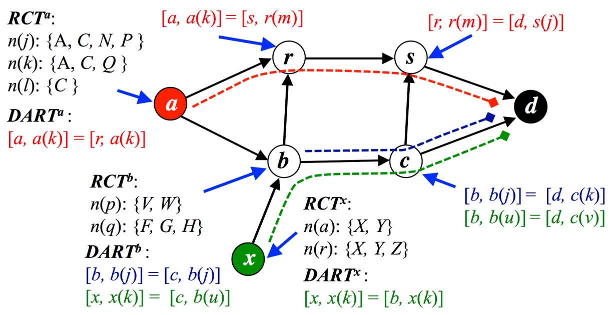


Fig. 7 – Interest forwarding in CCN-DART [**54**]

Router a forwards Interests intended for anchor d to neighbor r, and routers b and x forwards their Interests to neighbors s and c, respectively. Routers a, r, and s establish the following mappings in their DARTs: $[a; a(k)] \leftrightarrow [r; a(k)]$ at a, $[a; a(k)] \leftrightarrow [s; r(m)]$ at r, and $[r; r(m)] \leftrightarrow [d; s(j)]$ at s. These mappings denote the route (a, r, s, d) uniquely. Similarly, routers establish the DART mappings shown in the figure that denote the routes (x, b, c, d) and (b, c, d).

All the Interests from consumers local to routers a, b, and x regarding COs with names in prefixes for which d is an anchor can be routed towards d using the same few darts shown. Given that a data packet or NACK specifies the successor dart stated the Interest it answers, data packets and NACKs can be forwarded correctly from d (or a router caching the requested CO along the way to d) to routers a, b, or x unambiguously. In turn, routers a, b, and x can determine how to deliver the responses to local consumers based on the the RCT entries mapping each CO requested with the names of the customers that requested them.

4. CORRECTNESS OF CCN-DART

The following two theorems show that CCN-DART operates correctly. Theorem 3 shows that CCN-DART prevents Interests from being propagated along loops, independently of whether the topology is static or dynamic or the FIBs are consistent or not.

To discuss the correctness of Interest forwarding in CCN-DART, we say that a forwarding loop of h hops for a CO with name $n(j)$ occurs when Interests requesting the CO are forwarded by routers along a cycle

$L = \{v_1, v_2, ..., v_h, v_1\}$, such that router v_k receives an Interest for CO $n(j)$ from v_{k-1} and forwards the Interest to v_{k+1}, with $1 \le k \le h$, $v_{h+1} = v_1$, and $v_0 = v_h$.

Theorem 3: Interests cannot traverse forwarding loops in a network in which CCN-DART is used.

Proof: Consider a network in which CCN-DART is used. Assume for the sake of contradiction that routers in a forwarding loop L of h hops $\{v_1, v_2, ..., v_h, v_1\}$ send Interests for $n(j)$ along L, with no router in L detecting the incorrect forwarding of any of the Interests sent over the loop.

Given that L exists by assumption, $v_k \in L$ must send $I[n(j), h^I(v_k), dart^I(v_k)]$ to router $v_{k+1} \in L$ for $1 \le k \le h-1$, and $v_h \in L$ must send $I[n(j), h^I(v_h), dart^I(v_h)]$ to router $v_1 \in L$.

For $1 \le k \le h-1$, let $h(v_k, n(j)^*)^L$ denote the value of $h^I(v_k)$ when router v_k sends $I[n(j), h^I(v_k), dart^I(v_k)]$ to router v_{k+1}, with $h(v_k, n(j)^*)^L = h(v_k, n(j)^*, v_{k+1})$. Let $h(v_h, n(j)^*)^L$ denote the value of $h^I(v_h)$ when router v_h sends $I[n(j), h^I(v_h), dart^I(v_h)]$ to router $v_1 \in L$, with $h(v_h, n(j)^*)^L = h(v_h, n(j)^*, v_1)$.

Because no router in L detects the incorrect forwarding of an Interest and forwarded Interests are not aggregated in CCN-DART, each router in L must send its own Inte- rest as a result of the Interest it receives from the previous hop in L. This implies that router $v_k \in L$ must accept $I[n(j), h^I(v_{k-1}), dart^I(v_{k-1})]$ for $1 \le k < h$, and router $v_1 \in L$ must accept $I[n(j), h^I(v_h), dart^I(v_h)]$. According to DEAR, if router v_k sends Interest $I[n(j), h^I(v_k), dart^I(v_k)]$ to router v_{k+1} as a result of receiving $I[n(j), h^I(v_{k-1}), dart^I(v_{k-1})]$ from router v_{k-1}, then it must be true that $h^I(v_{k-1}) > h(v_k, n(j)^*)^L = h^I(v_k)$ for $1 < k \le h$. Similarly, if router v_1 sends $I[n(j), h^I(v_1), dart^I(v_1)]$ to router v_2 as a result of receiving $I[n(j), h^I(v_h), dart^I(v_h)]$ from router v_h, then $h^I(v_h) > h(v_1, n(j)^*)^L = h^I(v_1)$.

It follows from the above argument that, for L to exist and be undetected when each router in the loop uses DEAR to create DART entries, it must be true that $h^I(v_h) > h^I(v_1)$ and $h^I(v_{k-1}) > h^I(v_k)$ for $1 < k \le h$. However, this is a contradiction, because it implies that $h^I(v_k) > h^I(v_k)$ for $1 \le k \le h$. Therefore, the theorem is true.
Theorem 4 below addresses the ability for routers to send data packets and NACKs correctly to the consumers who issued the corresponding Interests using only the information stated in messages and stored in DARTs and RCTs. The theorem assumes that transmissions are sent correctly.

Theorem 4: CCN-DART ensures that, in the absence of failures, data packets and NACKs are sent correctly to the consumers that submitted the corresponding Interests.

Proof: If a router can resolve an Interest from a local consumer, it follows from Algorithm 1 that the result is true. Let router s be the origin of an Interest and let router $d \neq s$ be the router that replies to the Interest from s with a data packet or a NACK.

As Theorem 3 shows, Interests cannot traverse forwarding loops. Accordingly, if router d receives the Interest originated by s, then router d and all routers in the simple path from s to d must have established forwarding state according to Algorithm 4. Each router uses a different successor dart for each path traversing the router and defined by a predecessor name and a predecessor dart; therefore, each DART entry at a router uniquely denotes a different route traversing the router. Accordingly, given that router d can respond to an Interest only it it receives the Interest correctly, the proof can assume that the routers along the path from the source s to router d have established correct forwarding state in their DARTs. The rest of the proof must show that each router from d to s is able to demultiplex correctly the data packets and NACKs that traverse the reverse path established by Interests delivered from s to d.

Let h be the number of hops in the path traversed by a data packet or a NACK in response to an Interest originated at router s and answered by router d, and let f_i denote the router at the it h hop in such a path.

Basis Case: Let $h = 1$, then $s = f_1$. Router s labels its Interest with a dart assigned uniquely for its one-hop route to d, and remembers the user(s) that requested any CO in its RCT. According to Algorithm 4, router d responds to the Interest from s directly and includes the dart from the Interest in its response. According to algorithms 2 and 3, router s associates the CO name in the response with the local consumer(s) that submitted requests for that CO. It follows that the basis case is true.

Inductive Step: Assume that each router up to $k-1$ hops away from router d in the path from router d to router s receives and forwards a data packet or NACK from router d correctly over the path to s. We need to show hat the result is true for the router that is k hops away from router d in the path to router s, with $1 \le k \le h$.

When router f_k r eceives a data packet or NACK from router f_{k-1} c ontaining $dart^I(f_{k-1})$, it uses Algorithm 2 or Algorithm 3, respectively. Given that routers have established correct forwarding state in their DARTs, f_{k-1} and $dart^I(f_{k-1}))$ must be the successor and successor dart in an entry in $DART^{f_k}$, respectively. Furthermore, the predecessor of the entry in $DART^{f_k}$ must equal either f_k or f_{k+1}. In the first case ($k = h$ and $s = f_k$) , router f_k was the origin of the Interest being answered. In the second case ($k < h$) , router f_k forwards the response to router f_{k+1} swapping $dart^I(f_{k-1})$ for the predecessor dart listed in the DART entry, which is the dart that router f_{k+1} included in the Interest it sent to router f_k. Hence, router f_k must forward its response correctly to either the previous hop in the path from d to s or the origin of the Interest.

It follows by induction that a data packet or NACK traverses correctly the path of length h hops from router d to router s. Furthermore, the name of the CO in the data packet or NACK is the one stated in the Interest originated by router s on behalf of one or multiple local consumers. Because s uses its RCT to associate each CO name stated in a data packet or NACK with the correct set of local con- sumers that requested the CO, router s can forward the response to the correct local consumers. Therefore, the theorem is true.

5. PERFORMANCE COMPARISON

We compare the performance of CCN-DART and NDN using simulation experiments based on implementations of NDN and CCN-DART in the ndnSIM simulation tool [**2**].

The NDN implementation was used without modifications, and CCN-DART was implemented based on algorithms 1 to 4. The performance metrics used to compare NDN with CCN-DART are the number of entries needed in PITs and DARTs, the number of Interests handled by routers, and the delay incurred in obtaining a data packet or a NACK in response to an Interest.

The simulation parameters used for this study are the same as those presented in Section 2.3. We set the data rates of the links to 1Gbps to minimize the effects that link congestion or a sub-optimal implementation of CCN-DART or NDN may have on the results, especially for the case of end-to-end delays. The assumption that each router is locally attached to a content producer and local users requesting content constitutes the worst-case scenario for CCN-DART compared to NDN, because it results in many more DART entries. In a realistic deployment, only a small subset of the total number of routers in the network are attached to local producers of content. We consider on-path caching and edge caching. For the case of on-path caching, every router on the path traversed by a data packet from the producer to the consumer caches the CO. On the other hand, with edge caching, only the router directly connected to the requesting consumer caches the resulting CO.

5.1 Size of PITs and DARTs

Fig. 8 shows the average size and standard deviation of the number of entries in PITs used in NDN and the number of entries in DARTs and RCTs used in CCN-DART as a function of the total content-request rates per router. The vales shown for RCTs represent only the number of local pending Interests; the number of COs cached locally is not shown, given that the number of such entries would be very large and would be the same for NDN and CCN-DART.

As the figure shows, the size of PITs grows dramatically as the rate of content requests increases, which is expected given that PITs maintain per-Interest forwarding state. By contrast, the size of DARTs remains constant with respect to the content-request rates. The small average size of RCTs compared to the average size of PITs indicates that the average size of a PIT is dominated by the number of Interests a router forwards from other routers.

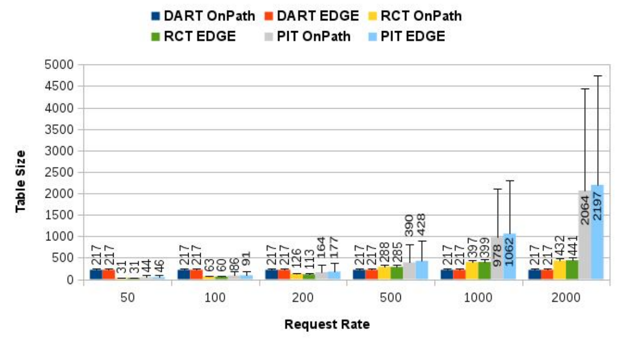


Fig. 8 – Size of PITs, DARTs and RCTs [**54**]

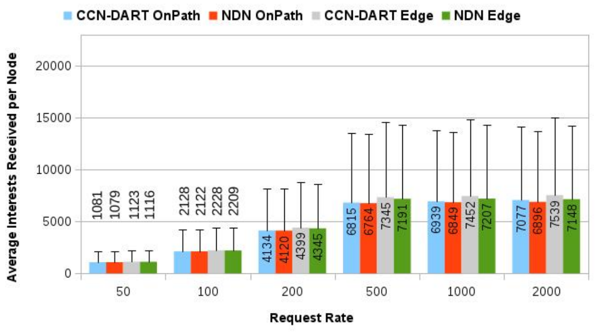


Fig. 9 – Number of Interests received by routers [**54**]

For low request rates, the average number of entries in a DART is actually larger than in a PIT. This is a direct consequence of the fact that a PIT entry is deleted immediately after an Interest is satisied, while a DART entry is kept for long periods of time (seconds) in our implementation, independently of whether or not it is used to forward Interests. Given the small sizes of DARTs, the cost of maintaining DART entries that may not be used at light loads is more than compensated for by the signiicant reduction in forwarding state derived from many Interests being forwarded using existing DART entries at higher request rates. This should be the case in real deployments, where the number of routers that are origins of routes to preix es is much smaller than the total number of routers. However, optimizing the length of time that a DART entry lasts as a function of its perceived utility for content forwarding is an area that deserves further study.

As the total content-request rate per router increases, the size of a PIT can be more than 10 to 20 times the size of a DART, because a given DART entry is used for many Interests in CCN-DART, while NDN requires a different PIT entry for each Interest. It is also interesting to see the effect of on-path caching compared to edge-caching. The average size of DARTs is independent of where content is being cached, and on-path caching in NDN does not make a significant difference in the size of a PIT compared to edge-caching.

5.2 Interest traffic and end-to-end delays

Fig. 9 shows the average number of interests received by each router in NDN and CCN-DART as a function of the content request rates for on-path caching and edge caching. The number of Interests received in CCN-DART is larger than the corresponding number for NDN. However, it is clear from the figure that the average numbers of Interests received by each router in NDN and CCN-DART are almost the same for all request rates.

The benefit of on-path caching is apparent for both NDN and CCN-DART, but appears slightly more pronounced for the case of CCN-DART. This should be expected, because CCN-DART does not aggregate Interests and on-path caching results in fewer Interests being forwarded.

Fig. 10 shows the average end-to-end delay for NDN and CCN-DART as a function of content-request rates for on-path caching and edge caching. As the figure shows, the average delays for NDN and CCN-DART are essentially the same for all content-request rates. This should be expected, given that in the experiments the routes in the FIBs are static and loop-free, and the number of Interests processed by routers is similar.

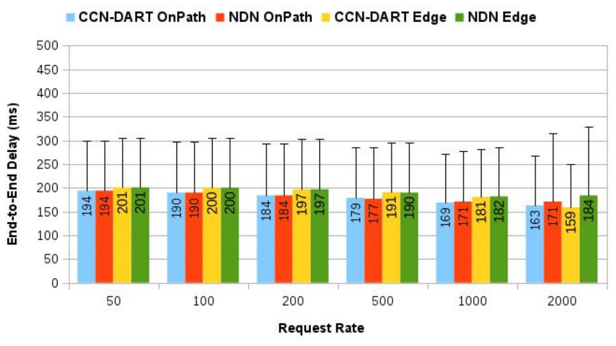


Fig. 10 – Average end-to-end delays [**54**]

5.3 Impact of Interest flooding attacks

Fig. 11 illustrates the impact of Interest flooding attacks on the size of PITs and DARTs. Fig. 11 shows the simulation results of an experiment illustrating the impact effect of Interest-flooding attacks on the size of PITs and DARTs. The network topology assumed is the same as in the other experiments. Only 10 of the 200 routers have local attackers generating random Interests corresponding to valid prefixes and invalid data objects with a constant request rate and uniformly distributed among prefixes. The other 190 routers have local users generating Interests for valid data objects according to a Zipf distribution with a request rate of 200 interests per second, which corresponds to a large population of local users to highlight the fact that Interest flooding attacks do not require many attackers.

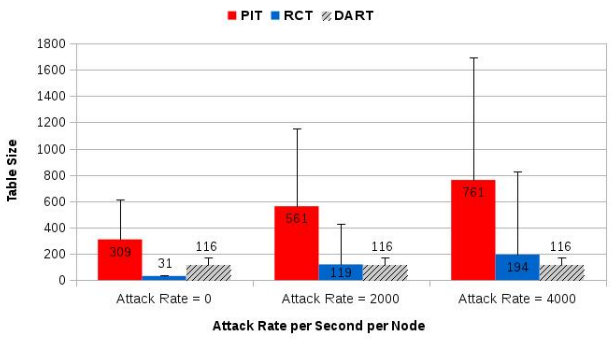


Fig. 11 – Size of PITs and DARTs under Interest flooding attacks

We ran simulations for three cases. The first case serves as the baseline; all routers receive 200 valid Interests per second from local users. In the second case, each router with local attackers receives 2000 invalid Interests per second. The resulting average size of the PITs almost doubles with respect to the case of no attacks, and the average size of DARTs does not change. In the third case, each router with local attackers receives 4000 invalid Interests per second, the average size of the PIT table increases dramatically, and the size of the DART still remains the same.

Interest flooding attacks in NDN translate into PITs that can easily be overwhelmed and much more traffic (Interests and NACKs). It is important to note that, by the very nature of Interest flooding attacks, Interest aggregation is not useful. Given the results from the other experiments and Section 2, it is clear that using PITs to maintain for- warding state is to the detriment of the system.

It is clear from the results of this experiment that the size of DARTs is not affected by the presence of Interest flooding attacks. The size of RCTs grows for those routers with local attackers in the presence of Interest flooding attacks, because RCTs maintain per-Interest state to enable the aggregation of local requests. However, limiting the size of RCTs can be done by limiting the rate at which any one local consumer is allowed to submit content requests, which impacts only that specific local user.

CCN-DART eliminates a major vulnerability of NDN, because forwarding tables cannot be attacked. However, an approach is still needed to address Interest flooding attacks in CCN-DART, given that invalid Interests still consume valuable bandwidth and can overwhelm content providers. Any viable solution to Interest flooding attacks requires RCTs (or content stores in the NDN case) to act as filters of valid requests. The design of such an approach is beyond the scope of this paper and is the subject of future work.

5.4 Implementation and deployment

The mappings stored in DARTs are equivalent to the label mappings first introduced for packet switching based on virtual circuits [26] and used today in high-performance routers running Multiprotocol Sabel Switching (MPLS). The small fixed-size darts and the relatively small number of DART entries needed for CCN-DART to operate at Internet scale are preferable by far to the long variable-length names (plus large nonces for the case of NDN) and the large number of PIT entries needed to maintain forwarding state in NDN [10, 32].

Both DARTs and PITs must be updated when the paths traversed by Interests and their responses must change due to congestion, topology changes, or mobility of consumers and providers. Yi et al [40] argue that a stateful forwarding plane enables a fast response to topology changes and congestion. However, the existence of MPLS fast rerouting mechanisms demonstrates that maintaining per-Interest forwarding state is not necessary to enable fast restoration of paths in the data plane. Similar mechanisms can be adopted in CCN-DART with much less signaling overhead than attempting to update forwarding tables with forwarding state for each Interest. Furthermore, approaches similar to those introduced in the past for congestion-oriented multipath routing and dynamic load balancing [34, 35] can be used in the context of CCN- DART, taking advantage of the fact that Interests cannot traverse loops.

CCN-DART provides native support for single-source multicasting. However, it separates the maintenance of multicast forwarding trees from the mechanisms used for source pacing and data dissemination over such trees. The benefit of this separation is that both pull and push-based mechanisms for multipoint communication can be used, and both are important for content-centric networking [**6**].

End users cannot mount Interest flooding attacks [3, 21, 33] to overflow DARTs in the same way that PITs can be attacked. A DART entry can be added only for valid anchors of name prefixes and for routes that satisfy the ordering constraint imposed with DEAR. Given that both conditions are managed in the control plane, mounting attacks on DARTs must be much more sophisticated than simply having users send Interests for COs corresponding to valid prefixes.

Countermeasures to different types of attacks on DARTs that are more sophisticated than Interest flooding attacks can be implemented based on configuration data or information protected in the control plane. For example, a neighbor router can be designated as an edge router and be limited to creating one dart per anchor. Similarly, Interests from neighbor routers must satisfy DEAR or be rejected, and an upper bound on the number of darts allowed from each neighbor for any prefix can be set based on the network size and the maximum number of routes to an anchor that can flow through that neighbor.

The performance results for edge and on-path caching we have presented have important consequences for CCN-DART deployments. An efficient deployment of CCN-DART could consist of using full content routers with FIBs, DARTs and RCTs only at the edge, and using “dart routers” elsewhere, which are are dedicated to content forwarding and maintain only FIBs and DARTs.

6. CONCLUSIONS

We proved that the way in which Interest loops are detected in NDN does not work correctly when Interests are aggregated along routing loops resulting from inconsistencies in FIB rankings or FIB entries. We have previously shown [17, 18, 20] how sufficient conditions like DEAR can be used in NDN to enable correct Interest loop detection when Interests are aggregated. However, as we have shown in Section 2, Interest aggregation is not needed in practice when in-network caches are used. Given the cost and vulnerabilities associated with using PITs, new designs are needed to apply content-centric networking at Internet scale.

To address the limitations of existing approaches to content-centric networking, we introduced CCN-DART, the first approach to Interest-based content-centric networking that supports Interest forwarding without revealing the sources of Interest and with no need to maintain forwarding state on a per-Interest basis.

CCN-DART replaces PITs with Data Answer Routing Tables (DARTs) that establish forwarding state for each route traversing the router over which many Interests are multiplexed, rather than establishing state for each different Interest using routes traversing the router.

We proved that forwarding loops cannot occur in CCN-DART, even if routing loops exist in the FIBs maintained by routers, and that data packets and NACKs are forwarded over the correct paths to consumers. The results of simulation experiments based on implementations of NDN and CCN-DART in ndnSIM show that CCN-DART is far more efficient than NDN. Compared to NDN, CCN-DART rendered the same end-to-end delays, incurred similar signaling overhead in the data plane, and resulted in a forwarding state with a number of entries smaller than one order of magnitude the number required in NDN.

Our results open up several research avenues for a far more scalable forwarding plane in content-centric networks, including the design of security mechanisms that prevent cache poisoning without maintaining per-Interest forwarding state. Additional research is also needed on the design of routing protocols and control-plane mechanisms that support routing to name prefixes based on FIBs with sizes that are orders of magnitude smaller than the growing number of domain names in the Internet.

REFERENCES

[1] B. Ahlgren et al., "A Survey of Information-centric Networking," *IEEE Commun. Magazine*, July 2012, pp. 26–36.

[2] A. Afanasyev et al., "ndnSIM: NDN simulator for ns-3", *University of California, Los Angeles, Tech. Rep*, 2012.

[3] A. Afanasyev et al., "Interest Flooding Attack and Countermeasures in Named Data Networking," *Proc. IFIP Networking '13*, May 2013.

[4] AT&T, "The Quality of Internet Service: AT&T's Global IP Network Performance Measurements," 2003. http://ipnetwork.bgtmo.ip.att.net/pws/paper.pdf

[5] J. Behrens and J.J. Garcia-Luna-Aceves, "Hierarchical Routing Using Link Vectors," *Proc. IEEE INFOCOM '98*, April 1998.

[6] A. Carzaniga et al., "Content-Based Publish/Subscribe Networking and Information-Centric Networking," *PRoc. ACM ICN '11*, August 2011.

[7] L. Ciavatone et al., "Standardized Active Measurements on a Tier 1 IP Backbone," *IEEE Comm. Magazine*, June 2003.

[8] Content Centric Networking Project (CCN) [online]. http://www.ccnx.org/releases/latest/doc/technical/

[9] A. Dabirmoghaddam et al., "Understanding Optimal Caching and Opportunistic Caching at The Edge of Information Centric Networks," *Proc. ACM ICN '14*, Sept. 2014.

[10] H. Dai el al., "On Pending Interest Table in Named Data Networking," *Proc. ACM ANCS '12*, Oct. 2012.

[11] E.W. Dijkstra and C.S. Scholten "Termination Detection for Diffusing Computations," *Information Processing Letters*, Vol. 11, No. 1, 1980.

[12] E.W. Dijkstra, W. Feijen, and A.J.M. van Gasteren, "Derivation of a Termination Detection Algorithm for Distributed Computations," *Information Processing Letters*, Vol. 16, No. 5, 1983.

[13] S. Fayazbakhsh et al., "Less Pain, Most of the Gain: Incrementally Deployable ICN," *Proc. ACM SIGCOMM '13*, 2013.

[14] C. Fricker et al., "Impact of traffic mix on caching performance in a content-centric network," *Proc. IEEE NOMEN Workshop '12*, 2012.

[15] J.J. Garcia-Luna-Aceves, "A Unified Approach to Loop-Free Routing Using Distance Vectors or Link States," *Proc. ACM SIGCOMM '89*, Aug. 1989.

[16] J.J. Garcia-Luna-Aceves, "Name-Based Content Routing in Information Centric Networks Using Distance Information," *Proc. ACM ICN '14*, Sept. 2014.

[17] J.J. Garcia-Luna-Aceves, "A Fault-Tolerant Forwarding Strategy for Interest-based Information Centric Networks," *Proc. IFIP Networking '15*, May 2015.

[18] J.J. Garcia-Luna-Aceves and M. Mirzazad-Barijough, "Enabling Correct Interest Forwarding and Retransmissions in a Content Centric Network," *Proc. ACM ANCS '15*, May 2015.

[19] J.J. Garcia-Luna-Aceves, "A More Scalable Approach to Content Centric Networking," *Proc. IEEE ICCCN '15*, Aug. 3-6, 2015.

[20] J.J. Garcia-Luna-Aceves, "Eliminating Undetected Interest Looping in Content Centric Networks," *Proc. IEEE NOF '15*, Sept. 30-Oct. 2, 2015.

[21] P. Gasti et al., "DoS and DDoS in Named Data Networking," *Proc. IEEE ICCCN '03*, 2013.

[22] E. Hemmati and J.J. Garcia-Luna-Aceves, "A New Approach to Name-Based Link-State Routing for Information-Centric Networks," *Proc. ACM ICN '15*, Sep. 30 - Oct. 2, 2015.

[23] V. Jacobson et al., "Networking Named Content," *Proc. IEEE CoNEXT '09*, Dec. 2009.

[24] B.N. Levine et al., "The Case for Reliable Concurrent Multicasting Using Shared ACK Trees," *Proc. ACM Multimedia '96*, Nov. 1996.

[25] A.K.M. Mahmudul-Hoque et al., "NSLR: Named-Data Link State Routing Protocol," *Proc. ACM ICN '13*, 2013.

[26] G. Markowsky and F.H. Moss, "An Evaluation of Local Path ID Swapping in Computer Networks," *IEEE Trans. Commun.*, Vol. COM-29, pp. 329-336, March 1981.

[27] J. Matocha and T. Camp, "A Taxonomy of Distributed Termination Detection Algorithms," *Journal of Systems and Software*, 1998.

[28] NDN Project [online]. http://www.named-data.net/

[29] W. So et al., " Toward Fast NDN Software Forwarding Lookup Engine Based on Hash Tables," *Proc. ACM ANCS '12*, 2012.

[30] M. Spohn and J.J. Garcia-Luna-Aceves, "Scalable Link-State Internet Routing," *Proc. IEEE ICNP '98*, Oct. 1998.

[31] C. Tsilopoulos et al., "Reducing Forwarding State in Content-Centric Networks with Semi-Stateless Forwarding," *Proc. IEEE INFOCOM '14*, April 2014.

[32] M. Varvello et al., "On The Design and Implementation of a Wire-Speed Pending Interest Table," *Proc. IEEE Infocom NOMEN Workshop '13*, April 2013.

[33] M. Virgilio et al., "PIT Overload Analysis in Content Centric Networks," *Proc. ACM ICN '13*, Aug. 2013.

[34] S. Vutukury and J.J. Garcia-Luna-Aceves, "A Simple Approximation to Minimum-Delay Routing," *Proc. ACM SIGCOMM '99*, Aug. 1999.

[35] S. Vutukury and J.J. Garcia-Luna-Aceves, "A Multipath Framework Architecture for Integrated Services," *Proc. IEEE Globecom '00*, Nov. 2000.

[36] M. Wahlisch et al., "Lessons from the Past: Why Data-driven States Harm Future Information-Centric Networking," *IFIP Networking '13*, May 2013.

[37] M. Wahlisch et al., "Backscatter from The Data Plane ? Threats to Stability and Security in Information-Centric Network Infrastructure," *Computer Networks*, Vol. 57, No. 16, Nov. 2013.

[38] G. Xylomenos et al., "A Survey of Information-centric Networking Research," *IEEE Communication Surveys and Tutorials*, July 2013.

[39] C. Yi et al., "Adaptive Forwarding in Named Data Networking," *ACM CCR*, Vol. 42, No. 3, July 2012.

[40] C. Yi et al., "A Case for Stateful Forwarding Plane," *Computer Communications*, pp. 779-791, 2013.

[41] W.T. Zaumen and J.J. Garcia-Luna-Aceves, "Dynamics of Distributed Shortest-Path Routing Algorithms," *Proc. ACM SIGCOMM '91*, Sept. 1991.

[42] W.T. Zaumen and J.J. Garcia-Luna-Aceves, "System for Maintaining Multiple Loop-free Paths between Source Node and Destination Node in Computer Network," US Patent 5,881,243, 1999.

[43] L. Zhang et al., "Named Data Networking," *ACM SIGCOMM Computer Communication Review*, Vol. 44, No. 3, July 2014.

[44] G. Carofiflio et al., "Enabling ICN in the Internet Protocol: Analysis and Evaluation of the Hybrid-ICN Architecture," *Proc. ACM ICN '19*, Sept. 2019.

[45] A. Drescher et al., "Analyzing the Performance of ICN Forwarders on the Wire," *Proc. ACM ICN '20*, Sept. 2020.

[46] E.K. Lua et al., "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes," *IEEE Communications Survey and Tutorial*, March 2004.

[47] J.J. Garcia-Luna-Aceves, "System and Method for Discovering Information Objects and Information Object Repositories in Computer Networks," U.S. Patent 7,162,539, January 2007.

[48] J. Raju, J.J. Garcia-Luna-Aceves and B. Smith, "System and Method for Information Object Routing in Computer Networks," U.S. Patent 7,552,233, June 2009.

[49] J.J. Garcia-Luna-Aceves and B. Smith, "System and Method for Using Uniform Resource Locators To Map Application Layer Content Names to Network Layer Anycast Addresses," U.S. Patent 7,343,422, March 2008.

[50] B. Zolfaghari et al., "Content Delivery Networks: State of the Art, Trends, and Future Roadmap," *ACM Computing Surveys*, April 2020.

[51] A. Ali Albalawi, and J.J. Garcia-Luna-Aceves, "Named-Data Transport: An End-to-End Approach for an Information-Centric IP Internet," *Proc. ACM ICN '20*, Sept. 2020.

[52] K. Schneider et al.,, "Near Loop-free Routing: Increasing Path Choices with Stateful Forwarding," *Proc. ACM ICN '17*, Sept. 2017.

[53] Z. Li et al, "Packet Forwarding in Named Data Networking Requirements and Survey of Solutions," *IEEE Communications Surveys and Tutorials*, 2019.

[54] J.J. Garcia-Luna-Aceves and M. Mirzazad-Barijough, "A Light-Weight Forwarding Plane for Content-Centric Networks," *Proc. IEEE ICNC 2016*, Feb. 2016.

AUTHORS

J.J. Garcia-Luna-Aceves is a distinguished professor of Computer Science and Engineering (CSE)at the University of California, Santa Cruz (UCSC) and serves as CSE Department Chair and CITRIS Campus Director. He was elected a corresponding member of the the Mexican Academy of Sciences in 2013. He is an IEEE life fellow, an ACM fellow, a AAAS fellow, a AAIA fellow, and a NAI fellow. He received the IEEE Computer Society Technical Achievement Award in 2011, the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee (AHSN TC) Technical Recognition Award in 2012, and the IEEE MILCOM Technical Achievement Award in 2016.

Maziar Mirzazad received his B.S. and M.S. degrees in computer science and IT engineering from the University of Tehran in 2008 and 2012, respectively. He received his Ph.D. degree in computer engineering from the University of California Santa Cruz in 2016. He is currently a staff software engineer at Apple AI/ML. His current interests include networks, distributed systems, and data Infrastructure and platforms.