Name-Based Content Routing in Information Centric Networks Using Distance Information

J.J. Garcia-Luna-Aceves [1]

Abstract
The Distance-based Content Routing (DCR) protocol is introduced, which enables routers to maintain multiple loop-free routes to the nearest sites advertising a named data object or name prefix in an information centric network (ICN), and establish content delivery trees over which all or some sites advertising the same named data object or name prefix can be contacted. In contrast to all prior routing solutions for ICNs, DCR operates without requiring routers to establish overlays, know the network topology, use complete paths to content replicas, or know about all the sites storing replicas of named content. It is shown that DCR is correct and that is orders of magnitude more scalable than recent name-based routing approaches for ICNs, in terms of the time and signaling overhead needed to obtain correct routing to named content.

Keywords
Content centric networks, information centric networks, name-based content routing, multi-path routing, name prefix routing, loop-free routing

[1] *Palo Alto Research Center*
*__Email address__: jjgla@parc.com

Contents

1. Introduction

In response to the shift in Internet usage patterns over the past few years from host-oriented communication to peer-to-peer and user-generated content, several information centric network (ICN) architectures have been proposed [1, 40, 4] as alternatives to the current Internet architecture. The goal of ICN architectures is to enable access to content and services by name, independently of their location, in order to improve system performance and end-user experience.

At the core of all ICN architectures are name resolution and routing of content, and several approaches have been proposed. In some ICN architectures, the names of data objects are mapped into addresses by means of directory servers, and then address-based routing is used for content delivery. By contrast, a number of ICN architectures use name-based routing of content, which integrates name resolution and content routing. With name-based routing, some of the routers (producers or caching sites) advertise the existence of local copies of named data objects (NDO) or name prefixes denoting a set of objects with names sharing a common prefix, and routes to them are established; the consumers of content issue content requests that are forwarded along the routes to the routers that issued the NDO or name prefix advertisements. This paper focuses on name-based routing in an ICN, which is an area that has not received much attention.

Section 2 summarizes the prior work on name-based routing in ICNs. Interestingly, no prior work has been reported using only distance information. This is perhaps due to the commonly-held view that routing based on distance vectors cannot provide multiple routes to destinations replicated multiple times in a graph, as is the case of cached or mirrored content in an ICN.

We show that efficient name-based routing to the nearest instances of content can be attained using only distance information, without requiring routers to know the network topology, exchange path information, maintain routes to all network sites, or even know about all the instances of an NDO or name prefix. Section 3 presents DCR (*Distance-based Content Routing*), which is the first name-based content routing approach for ICNs based solely on distance information. DCR provides an integrated approach for routing to *any, some, or all* instances of the same NDO or name prefix in an ICN. This is important, because many applications of name-based con-

tent routing in ICNs may require the ability to route to some or all instances of a given NDO or name prefix. DCR builds a *multi-instantiated destination spanning tree* (MIDST) using signaling that is much more efficient than the signaling introduced in the past for shared multicast trees (e.g., [3, 21, 30]) or the spanning-tree approach for publish-subscribe signaling introduced for content-based networking (CBN) [6]. DCR is an example of routing to multi-instantiated destinations [12] in which a destination is an NDO or name prefix.

Section 4 shows that DCR provides multiple paths to NDOs or name prefixes without ever creating a routing-table loop, and that it converges to shortest paths to the nearest copies of content over which content requests and content can flow.

Section 5 discusses the signaling needed for a routing protocol based on DCR, and the importance of using signaling messages that do not require routers to ask for updates.

Section 6 compares the control-plane overhead incurred by DCR and routing approaches for ICNs based on DHTs, link-state routing, and distance-vector routing. DCR incurs far less signaling overhead and is much faster to converge to correct routing tables than prior approaches, because it does not require routers to know the network topology or all the instances of content replicas. Section 7 shows the results of a simple simulation experiment comparing DCR with a link-state approach similar to NLSR and OSPFN [22, 38]. The result of the experiment illustrates the fact that DCR is far more efficient than name-based content routing that relies on information about all replicas of content.

Section 8 shows that DCR also has performance benefits in the data plane compared to routing protocols like NLSR that do not enforce loop-free routes to content.

2. Related Work

Directed Diffusion [15] was one of the first proposals for name-based routing of content. Requests for named content (called interests) are diffused throughout a sensor network, and data matching the interests are sent back to the issuers of interests. DIRECT [36] and LFBL [**?**] use a similar approach to named-based content routing in MANETs subject to connectivity disruption. Nodes use opportunistic caching of content and flood interests persistently.

Gritter and Cheriton [13] proposed one of the earliest proposals for name-based routing of content; namely, a name-based routing protocol (NBRP) as an extension of BGP. In essence, name-prefix reachability is advertised among content routers, and path information is used to avoid permanent loops. Another early development on name-based routing of content was the CBCB (combined broadcast and content based) routing scheme for content-based networking [6] CBCB consists of two components. First, a spanning tree of the network or multiple per-source trees based spanning the network are established. Then publish-subscribe requests for content are sent between consumers and producers of content over the tree(s) established in the network.

The ICN architectures proposed recently advocate various ways to accomplish name resolution and routing, and all of them use on-path caching of content [4], [40]. We summarize representative approaches below.

DONA [18] uses flat names for content and either global or local IP addressing and routing to operate. If only local IP routing is used, content requests (FIND messages) gather autonomous-system (AS) path information as they are forwarded, and responses are sent back on the reverse paths traversed by requests. Within an AS, IP routing is used.

Content Centric Networking [16] assumes the use of distributed routing protocols to build the routes over which content requests (Interest messages) are forwarded. A content request (called "Interest") may be sent over one or multiple routes to a name prefix. The use of a link-state routing approach for intra-domain routing is advocated, such that routers describe their local connectivity and adjacent resources (content); and adding content prefixes to BGP is proposed for inter-domain content routing.

Several ICN projects have content routing modalities based on the original CCN routing approach (e.g., [7, 8, 9, 29, 34]). NLSR [22] and OSPFN [38] are two protocols for name-based routing of content based on this approach. Routers exchange topology information by flooding two types of link states advertisements (LSA). LSAs can describe the state of physical links just as it is done in traditional link-state routing protocols. In addition, routers flood LSAs about the prefixes for which they have local copies. Routing in the Mobility First project [26] is similar to DONA and NBRP, in that it requires using either network addresses or source routing or partial source routing.

A number of ICN projects (e.g., [31, 34]) have addressed content routing modalities based on distributed hash tables (DHT) running in overlays over the physical infrastructure to accomplish name-based routing.

We observe that all prior content routing approaches use one or more of the following types of mechanisms: (a) maintaining paths to named content or using source routes to content; (b) flooding of information about the network topology and the location of replicas of content; (c) flooding of content requests; (d) establishing trees spanning the network over which name-based publish-subscribe signaling is performed; and (e) maintaining overlays for distributed hash tables (DHT).

3. DCR

The operation of DCR assumes that: (a) each network node is assigned a name or identifier with a flat or hierarchical structure; (b) each piece of content is a *named data object* (NDO) that can be requested by name; (c) NDOs can be denoted using either flat or hierarchical naming, and the same naming convention is used for the entire system; and (d) routers cache content opportunistically. DCR provides multiple loop-free routes to the nearest replicas of prefixes using anchor names and the sequence numbers they create to establish a

lexicographic ordering among routers. DCR extends prior sequence-numbering approaches used in protocols designed for routing to single-instance destinations (e.g., [5, 10, 24]). In addition, DCR establishes a MIDST (multi-instantiated destination spanning tree) for each NDO or prefix that requires routing to all or some of its replicas.

We denote the name of a specific NDO or a name prefix simply as *prefix*. A router that advertises having some or all the content corresponding to a prefix is called an *anchor* of the prefix. Each anchor of a prefix originates updates for the prefix periodically, and the update states the prefix, the name of the anchor, a distance to the prefix, and a sequence number that only the anchor is allowed to change.

We denote the lexicographic value of a name i by $|i|$, the set containing router i and its neighbor routers by N^i, and the set of next hops of router i for prefix j by S^i_j. The link from router i to router k is denoted by (i,k) and its cost is denoted by l^i_k. The cost of the link (i,k) is assumed to be a positive number that can be a function of administrative constraints and performance measurements made by router i for the link. The specific mechanism used to update l^i_k is outside the scope of this paper.

3.1 Information Stored and Exchanged

A router i running DCR maintains four tables: (a) a *link cost table* (LT^i) listing the cost of the link from router i to each of its neighbors; (b) a *neighbor table* (NT^i) stating routing information reported by each neighboring router for each prefix; (c) a *routing table* (RT^i) that stores routing information for each known prefix; and (d) a *multipoint routing table* (MRT^i) that stores routing information about routing trees created for those prefixes requiring multipoint communication support.

The entry in LT^i for link (i,k) consists of the name of neighbor k and the cost of the link to it (l^i_k).

The information stored in NT^i for each router $k \in N^i$ regarding prefix j is denoted by NT^i_{jk}, and consists of routing information for the nearest anchor and the root anchor of the prefix. The routing information for the nearest anchor reported by k consists of: the distance from neighbor k to j (d^i_{jk}); an anchor (a^i_{jk}) storing j; and the sequence number created by a^i_{jk} for j (sn^i_{jk}). The routing information for the root anchor of the prefix consists of: a root anchor (ra^i_{jk}); the distance from neighbor k to that anchor (rd^i_{jk}); and the sequence number created by ra^i_{jk} for j (rsn^i_{jk}). If prefix j is locally available at router i, then $a^i_{ji} = i$ and $d^i_{ji} = 0$. In this case router i is its own nearest anchor for prefix j, but need not be the root anchor for j.

The row for prefix j in RT^i specifies: (a) the name of the prefix (j); (b) the routing update information for prefix j (RUI^i_j); (c) the set of neighbors that are valid next hops (S^i_j); (d) a neighbor that offers the shortest distance to j ($s^i_j \in S^i_j$); and (e) an anchor list (A^i_j) that stores a tuple for each different valid anchor reported by any next-hop neighbor. Each tuple $[m, sn(m)] \in A^i_j$ states the name of an anchor m and the sequence number $sn(m)$ reported by that anchor.

RUI^i_j states: (a) a flag for each neighbor k denoting whether or not the information needs to be sent in an update to neighbor k (up^i_{jk}); (b) the current distance from i to j (d^i_j); (c) the anchor of j that has the smallest name among those that offer the shortest distance to j (a^i_j); and (d) the sequence number created by a^i_j for j (sn^i_j).

The entry for prefix j in MRT^i specifies: (a) the name of prefix j; (b) the multipoint update information for prefix j (MUI^i_j); and (c) the list of neighbor routers that have joined the MIDST for the prefix ($MIDST^i_j$). MUI^i_j states the root anchor of j (ra^i_j), the distance to the root anchor (rd^i_j), and the sequence number created by ra^i_j for prefix j (rsn^i_j).

An update message sent by router i to neighbor m consists of the name of router i; a message sequence number (msn^i) used to identify the message; and a list of updates, one for each prefix that needs updating. An update for prefix j sent by router i is denoted by U^i_j and states: the name of the prefix j; the distance to j (ud^i_j); an anchor (ua^i_j); and the sequence number created by ua^i_j for prefix j (usn^i_j).

The entry for prefix j in the update message received from neighbor k by router i is denoted by U^i_{jk}. It states the prefix name j, a distance to it (ud^i_{jk}), an anchor for the prefix (ua^i_{jk}), and the sequence number assigned to the prefix by the anchor (usn^i_{jk}).

Router i sends an update message periodically to each neighbor k containing updates made to RT^i since the last update message that i sent to neighbor k. Router i updates its distance table for $k \in N^i$ after any input event affecting the information stored in N^i_{jk}, including the local availability of prefix j at i. Router i stores new information reported by a neighbor k only when it includes an up-to-date sequence number; otherwise, router i does not trust the update, resets the information from that neighbor for the prefix, and schedules an update to correct the neighbor. Lastly, router i updates the entry for j in MRT^i based on updates received from its neighbors and signaling messages exchanged among routers to join the MIDST of j.

3.2 Routing to Nearest Copies of Prefixes

A router maintains the sequence numbers created by *all* the anchors reported by its neighbors. The information about a given anchor of a prefix is deleted after a finite time that is long enough to ensure that up-to-date information about valid anchors of the prefix is received, before anchor information is deleted.

A router can select neighbors as next hops to prefixes only if they report up-to-date information and offer shorter distances to the prefixes or the same distances but have lexicographically smaller names. Anchors send updates about their prefixes periodically and increment the sequence numbers they assign to prefixes. Let $sn(m)$ denote the sequence number associated with an anchor m in the set of anchors

known to router i for prefix j (A^i_j). The following condition is sufficient to ensure that no routing-table loops are ever created when routers change their next hops.

Successor-Set Ordering Condition (SOC):
Neighbor $k \in N^i$ can become a member of S^i_j (i.e., be a next hop to prefix j) if the following two statements are true:

$$\forall [m, sn(m)] \in A^i_j \; (\; a^i_{jk} \neq m \vee sn^i_{jk} \geq sn(m) \;) \tag{1}$$

$$\begin{aligned} (\; d^i_j < \infty \wedge \; [\; d^i_{jk} < d^i_j \vee (\; d^i_{jk} = d^i_j \; \wedge \; |k| < |i| \;) \;] \;) \vee \\ (\; d^i_j = \infty \; \wedge \; d^i_{jk} < d^i_j \; \wedge \\ \forall v \in N^i - \{k\} (\; [\; d^i_{jk} + l^i_k < d^i_{jv} + l^i_v \;] \vee \\ [\; d^i_{jk} + l^i_k = d^i_{jv} + l^i_v \; \wedge |k| < |v| \;] \;) \end{aligned} \tag{2}$$

Only those neighbors reporting the most recent sequence numbers from the known anchors of prefix j can be considered as next hops (Eq. (1)), and they are ordered lexicographically based on their distances to prefix j and their names (Eq. (2)). If router i has a finite distance to prefix j, then it can select neighbor k as a next hop to j if either k is closer to the prefix than router i or is at the same distance to the prefix but $|k| < |i|$. If router i has no finite distance to prefix j, then it can have k as a next hop to j only if k reports the smallest *finite* distance to j among all neighbors, or it has the smallest identifier among those neighbors reporting the smallest finite distance to j.

Fig. 1 illustrates how DCR routes to the nearest replica of a prefix. The figure shows the routing information used for a single prefix when four routers (d, o, r, and u) serve as the anchors, and each link has unit cost. One ore more tuples are listed in lexicographic order next to each router, with each tuple stating a distance to an anchor of the prefix and the identifier of that anchor. The first tuple in the list states the smallest distance to the prefix and the anchor with the smallest name among all anchors at that same distance. Updates from each router state only the preferred anchor (e.g., the update from node e states r as the anchor and distance 1 to it). Each additional tuple next to a router, if any, states an alternate anchor for the prefix and the distance to it. All routers are assumed to have received the most-recent sequence numbers from any of the anchors of the prefix.

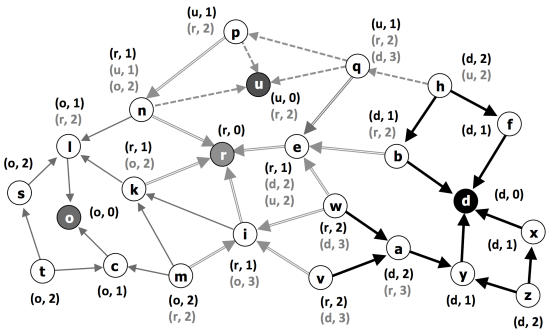


Figure 1. Routing to the nearest copy of a prefix

The updates generated by an anchor propagate only as long as they provide routers with shorter paths to prefixes. In Fig. 1, no routing update about the prefix propagates more than three hops, even though the network diameter is eight. In general, independently of how many anchors exist in a network for a given prefix, a router only has as many active anchors for a prefix as it has neighbors. This is the result of each router reporting only the best anchor it knows for each prefix. The arrowheads in the links between nodes indicate the router that is the next hop towards the prefix, and their shades indicate the anchor to which they point (e.g., the dark arrowheads point to d). Even in this small network of just 24 routers, most routers have multiple paths to the prefix, with at least one being a shortest path; all links can be used to forward requests for content; none of the routers know about all the four anchors of the prefix; and traversing any possible directed path in Fig. 1 necessarily terminates at d, o, r, or u, without traversing a loop.

To illustrate how SOC prevents routing-table loops, assume that router o fails. In this case, routers l and c are unable to find neighbors that satisfy SOC and must send updates stating an infinite distance and a null anchor. After processing the update from routers l and c, routers s and t are unable to find neighbors satisfying SOC and must also send updates stating infinite distances and null anchors. However, after processing the updates from routers l and c, routers n, k and m are able to find neighbors that satisfy SOC and hence respond to l or c with updates stating the tuples $[d^n_j = 1, a^n_j = r]$; $[d^k_j = 1, a^k_j = r]$; and $[d^m_j = 2, a^m_j = r]$. In turn, these updates allow routers c, l, o, s and t to attain routes to prefix j using anchor r within a very short time. Section 4 proves that no routing-table loops are formed while routers change their routing tables in response to network changes or prefixes being replicated dynamically.

Algorithms 1 and 2 illustrate how a router can update its distance and routing tables according to SOC to support routing to the nearest instances of prefixes. It is assumed that, if needed, router i sends a scheduled update message after the two algorithms are executed.

Router i uses Algorithm 1 to update the information reported by its neighbor k regarding prefix j, and to determine whether k has reported valid routing information. The algorithm is executed after i receives an update from neighbor k regarding prefix j or any other input event affecting the information in N^i_{jk}. Router i stores the information reported by neighbor k if it includes an up-to-date sequence number from the reported anchor of prefix j; otherwise, it resets the information from that neighbor for the prefix, and schedules an update to correct the neighbor.

Router i uses Algorithm 2 to determine which neighbors can be next hops for prefix j according to SOC. Using the information reported from these neighbors, it computes its minimum distance, anchor and sequence number for prefix j. The minimum distance to a prefix is computed using only those neighbors satisfying the constraints imposed by SOC for the cases in which $d^i_j < \infty$ and $d^i_j = \infty$.

Algorithm 1 Update Neighbor Entry

INPUT: A^i_j, NT^i_j, U^i_{jk}
$valid = 0$;
for each $[m, sn(m)] \in A^i_j$ **do**
  if $ua^i_{jk} \neq null \ \wedge \ (ua^i_{jk} \neq m \ \vee \ usn^i_{jk} \geq sn(m))$
  then $valid = 1$
end for
if $valid = 0$ **then**
  $up^i_{jk} = 1$ [schedule update U^i_j to neighbor k];
  $d^i_{jk} = \infty$; $a^i_{jk} = null$; $sn^i_{jk} = 0$
else
  $d^i_{jk} = ud^i_{jk}$; $a^i_{jk} = ua^i_{jk}$; $sn^i_{jk} = usn^i_{jk}$
end if
Execute Algorithm 2

Algorithm 2 Update Routing Entry

INPUT: N^i, NRT^i_j, RT^i_j;
$S^i_j = \emptyset$; $d_{min}(j) = \infty$;
if $d^i_j < \infty$ **then**
  for each $k \in N^i - \{i\}$ **do**
    if $d^i_{jk} < d^i_j \ \vee \ (\ d^i_{jk} = d^i_j \wedge |k| < |i| \ )$ **then**
      $S^i_j = S^i_j \cup \{k\}$;
      if $d^i_{jk} + l^i_k < d_{min}(j) \vee (d^i_{jk} + l^i_k = d_{min}(j) \wedge |a^i_{jk}| < |a^i_j|)$ **then**
        $d_{min}(j) = d^i_{jk} + l^i_k$;
        $s(j) = k$; $a(j) = a^i_{jk}$; $num(j) = sn^i_{jk}$
      end if
    end if
  end for
else
  for each $k \in N^i - \{i\}$ **do**
    if $d^i_{jk} + l^i_k < d_{min}(j)$ **then**
      $S^i_j = \{k\}$; $d_{min}(j) = d^i_{jk} + l^i_k$;
      $s(j) = k$; $a(j) = a^i_{jk}$; $num(j) = sn^i_{jk}$
    end if
  end for
end if
if $d_{min}(j) = \infty$ **then** $a(j) = null$; $num(j) = 0$; $s(j) = null$
if $(d^i_j \neq d_{min}(j) \vee a^i_j \neq a(j) \vee sn^i_j \neq num(j))$ **then**
  for each $k \in N^i - \{i\}$ **do** $up^i_{jk} = 1$ [schedule update U^i_j]
end if
$d^i_j = d_{min}(j)$; $s^i_j = s(j)$; $a^i_j = a(j)$; $sn^i_j = num(j)$

3.3 Routing to All or Some Copies of Prefixes

DCR supports routing to *all* or *some* anchors of the same prefix by means of *multi-instantiated destination spanning trees* (MIDST). All the anchors of a given prefix are connected with one another through the MIDST for the prefix, which is rooted at the anchor of the prefix with the smallest name, which we call the *root anchor* of the prefix. The MIDST is established using routing updates exchanged only by routers located between the root anchor and other anchors. To send data packets to all the anchors of a prefix, a router that is not part of the MIDST simply sends the packets towards the nearest anchor of the prefix; the first router in the MIDST that receives the data packets broadcasts them over the MIDST of the prefix.

The distance from router i to the root anchor ra^i_j is $rd^i_j = rd^i_{js} + l^i_s$, where $s \neq i$ is the next hop to ra^i_j selected by router i. If $i = ra^i_j$ then $rd^i_j = 0$. To build the MIDST for prefix j, routers select the root anchor of the prefix to be that anchor of the prefix that has the lexicographically smallest name; therefore, at each router i and for any neighbor $k \in N^i$, $|ra^i_j| \leq |ra^i_{jk}|$ and $|ra^i_j| \leq |a^i_{jk}|$.

The MIDST is established in a distributed manner using the distance updates exchanged among routers. A router that knows about multiple anchors for a prefix other than the anchor it considers to be the root anchor sends updates about the root anchor along the preferred path to each of the other anchors it knows. Routers that receive updates about the root anchor send their own updates to their preferred next hops to each other anchor they know. This way, distance updates about the root anchor propagate to all other anchors of the same prefix. Updates about the root anchor propagate only to those routers in preferred paths between the root anchor and other anchors. If router i changes its routing information for the root anchor of prefix j, it schedules an update about its root anchor to each neighbor that satisfies the following condition.

Root-Anchor Notification Condition (RNC):
Router i sends an update with the tuple $[ra^i_j, rd^i_j, rsn^i_j]$ to each router $k \in N^i - \{i\}$ for whom the following statements are true:

$$|a^i_{jk}| > |ra^i_j| \ \vee \ |ra^i_{jk}| > |ra^i_j| \tag{3}$$

$$\forall v \in N^i (a^i_{jv} = i) \ \vee \ \forall v \in N^i - \{k\} (\ a^i_{jk} \neq a^i_{jv} \ \vee \ (\ d^i_{jk} + l^i_k < d^i_{jv} + l^i_v \ \vee \ [\ d^i_{jk} + l^i_k = d^i_{jv} + l^i_v \wedge |k| < |v| \ ] \ ) \ ) \tag{4}$$

Eq. (3) states that k has not reported as its anchor or root anchor the same root anchor adopted by i. Eq. (4) states that i forwards the update about the root anchor to k if either i is an anchor and all its neighbors report i as their chosen anchor, or k is the lexicographically smallest next hop to an anchor that is not the root anchor.

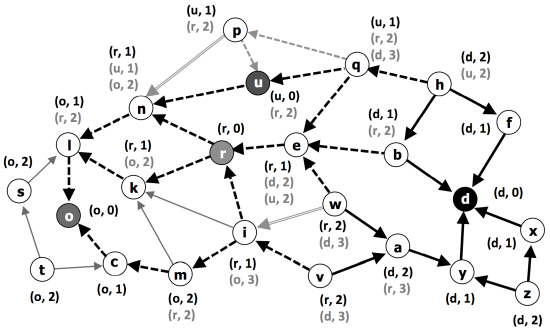


Figure 2. Propagating root anchor information

Fig. 2 shows how information about the root anchor of a prefix is propagated. In the example, router d has the smallest name among all the anchors of the prefix. The dark dashed arrowheads indicate those routers that propagate updates about d being the root anchor to some of their neighbors. For example, router b propagates a root anchor update to e because that neighbor is its best choice towards anchor r; and router g propagates a root anchor update to u because it is the best choice for u itself. Router r propagates an update stating that d is the root anchor to n and k because all its neighbors report r as their anchor, and e and i have already reported d as the root anchor.

Because of RNC, updates about the root anchor of a prefix reach all the other anchors of the prefix [12]. However, we

observe in Fig. 2 that many routers (e.g., o, p, s, t) do not participate in the propagation of updates about d being the root anchor of prefix j, and some (i.e., p, s, and t) do not even receive updates about d being the root anchor of prefix j. Only those routers along shortest paths between two different anchors of the prefix may participate in the signaling. This is much more efficient than the traditional approach to building shared multicast trees [3, 21, 30], in which all routers have to have routes to the root d, which needs to be pre-defined.

To allow anchors and relay routers to join the MIDST of a prefix, routers use the following condition to select their next hops towards the root anchor, and forward their requests to those neighbors. Eq. (5) states that the root anchor reported by k has the smallest name among all anchors of j known to i, and also reports an up-to-date sequence number from such an anchor. Eq. (6) states that k must offer the shortest distance to the root anchor among all neighbors. Eq. (7) orders router i with its selected next hop to the root anchor based on the distance to the anchor and the sequence number created by the anchor.

Root-Anchor Ordering Condition (ROC):
Router i can select neighbor $k \in N^i$ as its next hop to its root anchor for prefix j if the following statements are true:

$$|ra^i_{jk}| \leq |ra^i_j| \ \wedge \ rsn^i_{jk} \geq rsn^i_j \tag{5}$$

$$\forall m \in N^i \ (\ rd^i_{jk} + l^i_k \leq rd^i_{jm} + l^i_m \ ) \tag{6}$$

$$rsn^i_j < rsn^i_{jk} \ \vee \ [\ rsn^i_j = rsn^i_{jk} \ \wedge \ rd^i_{jk} < rd^i_j \ ] \tag{7}$$

Each anchor of a prefix originates a join request and sends it to its its lexicographically smallest next hop to the root anchor. The join request can be identified by the prefix, the prefix and a nonce, or the anchor name and the prefix. Each router receiving and forwarding a join request stores an entry for the request denoting the neighbor from which it was received for a finite period of time. The join request traverses the path towards the root anchor of the prefix, until it reaches the root anchor or a router x that is already part of the MIDST of prefix j. The response to the request traverses the reverse path of the join request and makes each router processing the response become part of the MIDST.

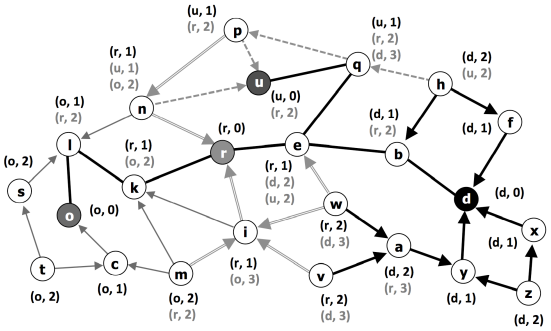


Figure 3. The MIDST of a prefix

Fig. 3 shows the resulting MIDST for the prefix in the same example of Fig. 2. Anchors u, r, and o send their join requests towards d to join the MIDST of the prefix. The links that constitute the MIDST are indicated by solid lines in the figure, and multipoint data traffic for the prefix can flow in both directions of those links.

To forward a content request to all anchors of a prefix, a router simply forwards the request to one of the nearest anchors of the prefix, and the request is then broadcast over the MIDST of the prefix as soon as it reaches the first router that has joined the MIDST. This approach is much the same as that used in shared-tree multicast routing.

Different approaches can be implemented to forward a content request to some of the anchors of a prefix. If enough nearest anchors are known, the request can be forwarded directly to them. Else, the request can be sent to a nearest anchor, who can then forward the request over the MIDST. Some of the anchors can be reached over the MIDST either by specifying a maximum number of hops that a request should traverse on the MIDST, or by means of a labeling scheme over the MIDST that denotes how many anchors can be reached through each branch of the MIDST at a given router in the MIDST(e.g., see [19]).

Specifying when a prefix requires routing to all its replicas, and hence the need for a MIDST, can be done in a number of ways. One approach is for the name of the prefix to denote the need for a MIDST; this is the equivalent of a multicast address for the case of traditional routing. An alternative approach is for a special signaling message to request the creation of a MIDST for a given prefix.

4. DCR Correctness

The following theorems prove that DCR guarantees that routing-table loops are never formed, even as the locations of content and the network topology change. Furthermore, routers running DCR attain the shortest distances to the nearest instances of each known prefix.

Theorem 4.1. *No routing-table loops can be formed if SOC is used to select the next hops to prefixes at each router.* □

Proof. The proof is by contradiction. Assume that a routing loop L_j for prefix j consisting of h hops is created at time t_L when the routers in L_j change successors according to SOC. Let $L_j = (n_1, n_2, ..., n_h)$, with $n_{i+1} \in S^{n_i}_j$ for $1 \leq i \leq h-1$ and $n_1 \in S^{n_h}_j$. According to SOC, each hop $n_i \in L_j$ $(1 \leq i \leq h)$ can select its next hops (i.e., $S^{n_i}_j$) in only two ways, depending on whether or not $d^{n_i}_j < \infty$ when $n_i \in L_j$ selects its next hops before or at time t_L when L_j is formed.

Assume that there is a subset of hops $I_j \subset L_j$ such that $d^{n_m}_j = \infty$ when n_m joins L_j by adding n_{m+1} to $S^{n_m}_j$ for each $n_m \in I_j$. By assumption, router n_m uses Eq. (2) in SOC; therefore, $d^{n_m}_{jn_{m+1}} < \infty$ and router n_{m+1} must report to n_m either an anchor that router n_m did not know before, or a more recent sequence number created by an anchor known to n_m before the update from n_{m+1}. Furthermore, for all $q \in N^{n_m}$, it must be

true that $d^{n_m}_{jq} > d^{n_m}_{jn_{m+1}}$ or $d^{n_m}_{jq} = d^{n_m}_{jn_{m+1}}$ and $|n_m| < |q|$. Therefore, the following relation must hold between $d^{n_m}_{jn_{m+1}}$ and $d^{n_m}_{jn_{m-1}}$ for any $n_m \in I_j$:

$$d^{n_m}_{jn_{m-1}} > d^{n_m}_{jn_{m+1}} \quad (8)$$
$$\vee \left(d^{n_m}_{jn_{m-1}} = d^{n_m}_{jn_{m+1}} \wedge |n_{m-1}| > |n_{m+1}| \right)$$

Consider a subset of hops $\{n_m, n_{m+1}, ..., n_{m+c}\} \in I_j$ that forms a contiguous chain in L_j, where $c \leq h$. It follows from Eq. (8) that

$$\left(d^{n_m}_{jn_{m-1}} = d^{n_{m+c}}_{jn_{m+c+1}} \wedge |n_{m-1}| > |n_{m+c}| \right) \quad (9)$$
$$\vee \left(d^{n_m}_{jn_{m-1}} > d^{n_{m+c}}_{jn_{m+c+1}} \right) \text{ for } h \geq c \geq 0.$$

On the other hand, by assumption, every hop $n_i \in L_j - I_j$ must have $d^{n_i}_j < \infty$ when it uses SOC to select its next hops and hence join L_j. Therefore, the following two equations must be satisfied for any $n_i \in L_j - I_j$:

$$d^{n_i - 1}_{jn_i} \geq d^{n_i}_j \quad (10)$$
$$d^{n_i}_j > d^{n_i}_{jn_{i+1}} \geq d^{n_{i+1}}_j \quad (11)$$
$$\vee \left(d^{n_i}_j = d^{n_i}_{jn_{i+1}} \geq d^{n_{i+1}}_j \wedge |n_i| > |n_{i+1}| \right).$$

Consider a subset of hops $\{n_l, n_{l+1}, ..., n_{l+k}\} \in L_j - I_j$ that forms a chain in L_j, where $k \leq h$. Then either $d^{n_{l+i}}_j > d^{n_{l+i}}_{jn_{l+i+1}} \geq d^{n_{l+i}}_j$ for at least one hop n_{l+i} in the chain, or $d^{n_{l+i}}_j = d^{n_{l+i}}_{jn_{l+i+1}} \geq d^{n_{l+i}}_j$ and $|n_{l+i}| > |n_{l+i+1}|$ for each hop n_{l+i} in the chain. Accordingly, it follows from Eqs. (10) and (11) that

$$\left(d^{n_l}_{jn_{l+1}} = d^{n_{l+k}}_{jn_{l+k+1}} \wedge |n_l| > |n_{l+k}| \right) \quad (12)$$
$$\vee \left(d^{n_l}_{jn_{l+1}} > d^{n_{l+k}}_{jn_{l+k+1}} \right) \text{ for } h \geq k \geq 0.$$

It follows from Eqs. (9) and (12) that using SOC enforces the same lexicographical ordering among the hops of L_j for any given combination of chains of nodes in L_j that belong to I_j or $L_j - I_j$ and use SOC to select their next hops when they join L_j. Accordingly, it must be true that, if at least one hop in $n_i \in L_j$ is such that $d^{n_i}_{jn_{i+1}} > d^{n_k}_{jn_{k+1}}$, where $n_k \in L_j$ and $k > i$, then $d^{n_m}_{jn_{m+1}} > d^{n_m}_{jn_{m+1}}$ for any given $m \in \{1, 2, ..., h\}$, which is a contradiction. On the other hand, if $d^{n_i}_{jn_{i+1}} = d^{n_k}_{jn_{k+1}}$ for any n_i and n_k in L_j, then $|n_m| > |n_m|$ for any given $m \in \{1, 2, ..., h\}$, which is also a contradiction. Therefore, L_j cannot be formed when routers use SOC to select their next hops to prefix j. □

Assume that DCR is executed in a connected finite network G, that a router is able to detect within a finite time who its neighbor routers are, and that any signaling message sent over a working link between two routers is delivered correctly within a finite time. Further assume that topological changes and name prefix changes stop taking place after a given time t_T. The following theorem proves that DCR attains shortest paths to the nearest replicas of known prefixes within a finite time assuming for simplicity that the cost of any operational link is one unit.

Theorem 4.2. *If DCR is used in network G, routes to prefixes converge to the shortest distances to the nearest anchors of the prefixes within a finite time after t_T.* □

Proof. Without loss of generality, we focus on a specific prefix j. The proof is by simple induction on the number of hops (k) that routers are away from the nearest anchors of prefix j. Let the set of anchors in the network for prefix j be $A = \{\alpha_1, \alpha_2, ..., \alpha_r\}$, where r is at most equal to the number of routers in the network.

Basis case: For $k = 1$, consider an arbitrary neighbor n_1 of a given anchor α_i of prefix j, with $1 \leq i \leq r$. Given that the signaling between neighbors is reliable and no links fail after time t_T, router n_1 must receive an update $U^{\alpha_i}_j$ from α_i stating $d^{\alpha_i}_j = 0$, $a^{\alpha_i}_j = \alpha_i$, and $sn^{\alpha_i}_j = s(\alpha_i)$ (the most recent sequence number created by α_i) within a finite time after t_T; and it must update $d^{n_1}_{j\alpha_i} = 0$, $a^{n_1}_{j\alpha_i} = \alpha_i$, and $sn^{n_1}_{j\alpha_i} = s(\alpha_i)$.

Because $d^{n_1}_{j\alpha_i} = 0$ and $sn^{n_1}_{j\alpha_i} = s(\alpha_i)$ always satisfy SOC at router n_1 for prefix j, it must be the case that $\alpha_i \in S^{n_1}_j$. Furthermore, any other next hop in $S^{n_1}_j$ must also be an anchor, because the smallest link cost between neighbors equals 1 and hence the smallest value of $d^{n_1}_j$ equals 1. Router n_1 must set $d^{n_1}_j = 1$ and send and update stating that distance, together with the name of that anchor and the sequence number it created, after a finite time $t_1 > t_T$. Therefore, the basis case is true.

Inductive step: Assume that the theorem is true for an arbitrary router n_k that is k hops away from its nearest anchors of prefix j. It must then be true that $d^{n_k}_j = k$ after a finite time $t_k > t_1$. By assumption, the signaling between neighbors is reliable and no links fail after time $t_T < t_k$; therefore, each neighbor of n_k must receive updates from n_k a finite time after t_k stating $d^{n_k}_j$, $a^{n_k}_j$, and $sn^{n_k}_j$. Each neighbor $p \in N^{n_k}$ must update $d^p_{jn_k} = k$, $a^p_{jn_k} = a^{n_k}_j$, and $sn^p_{jn_k} = sn^{n_k}_j$ a finite time after t_k.

Let router $q \in N^{n_k}$ be more than k hops away from any anchor of prefix j. Because $d^{n_k}_j = k$ is the shortest distance from n_k to prefix j after time t_k, router q cannot have any neighbor reporting a distance to j smaller than k after time t_k. Therefore, $d^q_{jn_k}$ must satisfy Eqs. (1) or (2) of SOC within a finite time after t_k and router q must make n_k a next hop to prefix j a finite time after t_k. Furthermore, any neighbor of q in S^q_j must have also reported a distance of k hops to j. Router q selects the anchor in S^q_j with the smallest name, and sends an update within a finite time $t_{k+1} > t_k$ stating $d^q_j = k+1$, together with the name of its chosen nearest anchor and the most recent sequence number created by that anchor. Therefore, router q and hence any router $k+1$ hops away from the nearest anchors of prefix j must attain a shortest distance of $k+1$ hops to prefix j within a finite time and the theorem is true. □

The following theorems address the correctness of DCR for routing to multiple sites announcing a destination. Theorem 4.3 shows that routes to root anchors are loop-free. Using the same assumptions introduced for the proof of Theorem

4.2, Theorems 4.4 and 4.5 show that DCR builds a MIDST for a destination that requires it.

Theorem 4.3. *No routing-table loops can be formed if ROC is used to select next hops to the root anchors of destinations.* □

Proof. The proof is by contradiction. Assume that a routing loop L_r for root anchor r consisting of h hops is created when nodes in L_r change successors according to ROC. Let $L_r = (n_1, n_2, ..., n_h)$, with n_{i+1} being the next hop of n_i for $1 \leq i \leq h-1$ and n_1 being the next hop of n_h.

According to ROC, it must be true that $rsn_r^{n_i} \leq rsn_r^{n_{i+1}}$ for $1 \leq i \leq h-1$ and $rsn_r^{n_h} \leq rsn_r^{n_1}$ for each hop $n_i \in L_r$ $(1 \leq i \leq h)$. This result implies that $rsn_r^{n_i} = rsn_r^{n_{i+1}}$ for $1 \leq i \leq h-1$ and $rsn_r^{n_h} = rsn_r^{n_1}$. Because of ROC, it follows from this result that $rd_r^{n_i} < rd_r^{n_{i+1}}$ for $1 \leq i \leq h-1$ and $rd_r^{n_h} < rd_r^{n_1}$, which implies that $rd_r^{n_i} < rd_r^{n_i}$ for $1 \leq i \leq h$. This is a contradiction and hence the theorem is true. □

Theorem 4.4. *If DCR is used in network G, each anchor of destination j attains a shortest distance to the root anchor of the same destination within a finite time after t_T.* □

Proof. It follows from Theorem 4.2 that all routers must have a route to their nearest instances of destination j after a finite time $t_R \geq t_T$. Let r be the root anchor of destination j, then $|r| < |u|$ for any router $u \neq r$ that is an anchor of destination j. Let A be the set of anchors of destination j other than r, and let $a \in A$ be such that shortest physical paths between a and r in G do not include any other anchor. The length of any such path P_{ar} must be odd or even.

Assume that $|P_{ar}| = 2h$, where h is an integer. Because DCR converges to correct routing information for the nearest anchors, there is a router $m \in P_{ar}$ with $d_j^m = rd_j^m = h$, $a_j^m = ra_j^m = r$, and two neighbors n_1 and n_2 in $N^m \cap P_{ar}$. Let $|P_{n_2 r}| = |P_{mr}| + 1$, then $|P_{an_2}| = h-1$ hops. Given that $P_{an_2} \subset P_{ar}$, any nearest anchor to n_2 must be $h-1$ hops away. Without loss of generality, let $a_j^{n_2} = a$. According to SOC, n_2 must report $d_j^{n_2} = h-1$ and $a_j^{n_2} = a$ to m. Without loss of generality, assume that $|n_2| \leq |v|$ with $v \in N^m$, $d_{jv}^m = h-1$ and $a_{jv}^m = a$. According to RNC, given that $a \neq r$ and $P_{am} \subset P_{ar}$ is a shortest path, router m must send an update stating $ra_j^m = r$ and $rd_j^m = h$ to neighbor n_2. Using a similar argument, it can be shown that n_2 and every router in the subpath from n_2 to a in P_{ar} must send an update with its own distance to r to the next hop in P_{ar} towards a. Hence, given that P_{ar} is a shortest path, it must be true that a obtains a shortest distance to r in a finite time if $|P_{ar}|$ is even.

Assume that $|P_{ar}| = 2h+1$. There must be a router $n_1 \in P_{ar}$ with $d_j^{n_1} = h$ and $a_j^{n_1} = r$ and a router $n_2 \in N^{n_1} \cap P_{ar}$ with $d_j^{n_2} = h$ and $a_j^{n_2} = a$. According to SOC, they inform each other of their distances and anchors for j. Using an argument similar to the one above, it can be shown that, because of RNC, router n_1 must inform n_2 of its distance to r, and every router between n_2 and a in P_{ar} must send an update with its own distance to r to the next hop in P_{ar} towards a. Therefore, anchor a must have a shortest distance to r in a finite time if $|P_{ar}|$ is odd. It then follows that $a \in A$ must attain a shortest distance to r in a finite time.

Consider now an anchor $b \in A$ such that a physical shortest path P_{br} between b and r includes a sequence of $k \geq 1$ other anchors $(a_1, a_2, ..., a_k)$, with a_1 being the closest to r. It follows from the previous argument that a_1 must attain a shortest route to r. By induction on k, it also follows that a_k must also attain a shortest distance to r in a finite time. Accordingly, it follows from the argument made for anchor a that anchor b must obtain a shortest distance to r in a finite time. Therefore, the theorem is true. □

Theorem 4.5. *If DCR is used in network G, a MIDST of destination j is established that includes all anchors of destination j within a finite time.* □

Proof. If follows from Theorem 4.4 that every anchor other than r must have at least one next hop to r. To join the MIDST, each anchor $a \neq r$ sends a join request along its smallest next hop to r. Because routes to r are loop free (Theorem 4.3), the join request must reach either r or a router in the MIDST, which in turn sends a response that makes the routers in the path to a join the MIDST. Therefore, the theorem is true. □

5. Name-based Signaling for DCR

DCR can be implemented using different name-based signaling approaches, depending on the ICN architecture in which it is used. The main decisions to be made are: (a) how to name routers, (b) the syntax of update messages, and (c) how to exchange such messages between neighboring routers. The naming of routers can be done using a hierarchical name space like the one proposed for NLSR [22]. With this naming scheme, the name of a router would be "$/ < network > / < site > / < router >$", and the name of the DCR daemon running in it would be "$/ < network > / < site > / < router > /DCR$."

The semantics of the update process in DCR consists of a router sending incremental routing-table updates to its neighbors. An update message is identified by the name of the router, the protocol (DCR), the type of message, and a sequence number incremented by the sending router. The update messages can be sent periodically and serve as an indication that the router is operational. The syntax of update messages and exactly how messages are exchanged between routers depend on the basic signaling defined in the ICN architecture in which DCR operates.

The signaling among routers can be receiver-initiated or sender-initiated. With receiver-initiated signaling, data follow an Interest stated by the receiver. The DCR process in a router using NDN or CCN must periodically send a "Routing Interest" (RI) to elicit routing updates from its neighbors. In contrast to NLSR, the state of each neighbor router is different and hence DCR signaling cannot be based on CCNx Sync as it is done in NLSR [22]. A router must send an RI stating "$/ <$

$network > / < site > / < router > /DCR/update\ /seq\_no$" to each neighbor requesting routing information. A router receiving the RI responds with a content object corresponding to a "Routing Update" (RU) with the updates made to its routing table since the last RI.

Sender-initiated signaling consists of each router sending RUs to all its neighbors, without having to be asked explicitly by any one neighbor. Implementing this signaling approach is much more efficient but requires adding a "push" mechanism in the data plane of some ICN architectures (CCN and NDN) or changing the semantics of Interests. Hello messages stating the presence of a router can be sent as long-term RIs to which multiple RUs can be sent by the same router as needed. Alternatively, an RU can be sent as an Interest containing updates as a payload. The RIs or RUs sent from a router inform its neighbors that the router is alive.

If DCR were used in CCN or NDN, a DCR update would be digitally signed by the anchor that originates the update, and the update would state information that can be used in signature verification [16]. Hence, the problem of ensuring in DCR that an anchor of a prefix is valid is the same as the problem of ensuring in NLSR that an LSA regarding local content is valid. Security mechanisms similar to those proposed in NLSR [22] could be implemented in DCR. However, the design, verification, and performance of efficient security mechanisms for DCR and other name-based content routing protocols is outside the scope of this paper.

6. Control-Plane Efficiency

To compare the performance of DCR with other name-based content routing approaches, we focus on the time, communication, and storage complexities of the approaches in the control plane. The number of routers in the network is denoted by N and E denotes the number of network links. The number of different name prefixes available in the network is denoted by C, the average number of replicas for a given name prefix is R, the average number of neighbors per router is l, and the network diameter is d.

We assume that a separate control message is sent for any given LSA or distance update. In practice, multiple LSAs and distance updates can be aggregated to conserve bandwidth. In fact, aggregating distance updates for multiple prefixes is easier than aggregating LSAs from multiple sources. However, given that the maximum size of a control message is a constant value independent of the growth of N or C, this aggregation does not change the order size of the overhead incurred by the routing protocols.

The communication complexity (CC) of a routing protocol is the number of messages that must be transmitted successfully for each router to have correct routing information about all the C prefixes. The time complexity (TC) of a routing protocol is the maximum time needed for all routers to have correct routing information for all prefixes when all messages are transmitted successfully. The storage complexity (SC) of a routing protocol is the maximum number of entries in the routing table of an arbitrary router. Given that the difference in the number of messages exchanged between neighbors with receiver-initiated or sender-initiated signaling is independent of N, C and R, our results apply to both types of signaling.

Link-State Routing (LSR):
This approach is used in NLSR and OSPFN. The LSA originated by a router regarding a link or a name prefix must be sent to all the other routers in the network, a router must transmit an LSA for each adjacent link and each prefix that is stored locally and each LSA must be flooded in the network, and each router must store a record for each link and prefix copy in the network. Accordingly, the time, communication, and storage complexities of LSR are:

$$TC_{LSR} = O(d);\ CC_{LSR} = O(ERC + lEN);\ SC_{LSR} = O(RC + E) \tag{13}$$

Distributed Hash Table (DHT):
The most efficient DHT scheme is a virtual DHT with one-hop routing [14], such that routers run the DHT locally and maintain routes to all routers in the network. The communication complexity associated with publishing a prefix in the DHT and associating r sites with the prefix (to support routing to any or all copies of the prefix) is $O(dRC)$ assuming no loops. The communication complexity of maintaining routes to all routers is $O(lNE)$, given that link-state routing is typically used. The fastest possible propagation of routes to all routers is order $O(d)$, and each router must store a record for as many prefixes as there are in the network. It follows that the complexity of the DHT approach is:

$$TC_{DHT} = O(d);\ CC_{DHT} = O(dRC + lEN);\ SC_{DHT} = O(C + E) \tag{14}$$

Traditional Distance-Vector Routing (DVR):
Because DVR signaling can traverse long paths, and long-term loops and "counting to infinity" can occur, the basic distance-vector approach is known to have $O(N)$ time complexity and $O(N^2)$ communication complexity [17]. Furthermore, each router must store and communicate distance information about all prefix replicas and destination nodes in the network. Accordingly, the complexity of DVR is:

$$TC_{DVR} = O(N);\ CC_{DVR} = O(N^2RC);\ SC_{DVR} = O(RC + N) \tag{15}$$

This is much worse than the previous two approaches and explains why name-based routing using distance vectors has not been considered in the past.

Distance-based Content Routing (DCR):
Independently of the number of anchors for a given prefix or routers in the network, the information a router stores and communicates for a given prefix in DCR is only its distance to the nearest anchor of the prefix, plus the anchor name and the latest sequence number created by that anchor. As the number of replicas increases, the distances from a router to the nearest replica of a prefix decreases, and it is always the

case that the number of hops from any router to the nearest replica of a prefix (x) is at most d hops. Furthermore, DCR does not incur any routing-table loops. This means that: (a) any routing information propagates as fast as the shortest path between its origin and the recipient; and (b) the number of messages required for all routers to have a correct distance to a given prefix is $O(E)$, regardless of the number of times R the prefix is replicated. Given that there are C prefixes in the network, the complexity DCR is:

$$TC_{DCR} = O(x);\ CC_{DCR} = O(EC);\ SC_{DCR} = O(C) \tag{16}$$

It is clear from Eqs. 13 to 16 that DCR has far smaller storage complexity than LSR or DVR, because a router only stores one entry for a prefix, rather than entries for prefix replicas, and does not store any topology information. As R becomes $O(N)$, DCR requires orders of magnitude less storage overhead than LSR and DVR in large networks. DCR is also more efficient in terms of storage than the DHT approach in large networks, because a DHT requires routers to maintain network topology information. DVR allows routers to attain correct routing tables much faster than with DVR, LSR or DHT, because routers only exchange updates about nearest prefix copies, rather than all copies, and such updates need to traverse paths that become much shorter than the network diameter as content replicas proliferate.

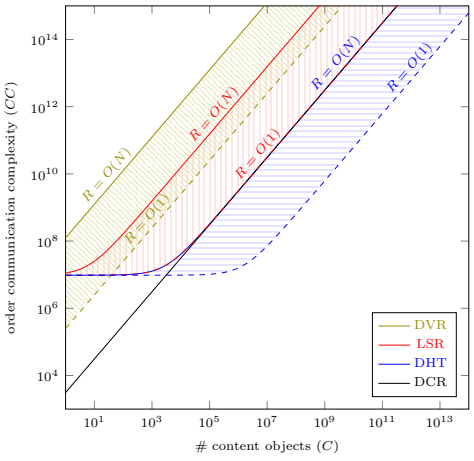


Figure 4. Impact of C and R on overhead

Fig. 4 illustrates the combined effect of the values of C and R on the communication complexity (signaling overhead) of DVR, LSR, DHT and DCR. To focus on the effect of C and R, we set $N = 500$, and assume that l and d are of order $O(log(N))$, which makes E order $O(Nlog(N))$. The number of replicas per objet is varied from order $O(1)$ to order $O(N)$. It is clear that the DVR and LSR approaches are orders of magnitude less efficient than the DHT and DCR approaches when content replicas proliferate. It is also worth noting that, as the number of content objects becomes far larger than the number of network nodes and the copies of such content objects proliferate, the signaling overhead of DCR and an ideal DHT approach are the same.

7. Simulation Experiment

A simple simulation experiment is used to contrast the signaling overhead incurred by DCR and the LSR approach exemplified by NLSR and OSPFN. QualNet [35] (version 5.0) is the discrete event simulator used. The implementation of DCR was based on modifications of the Distributed Bellman Ford implementation in Qualnet to support SOC, and the implementation of the LSR approach was based on the OSPF implementation by adding LSAs to advertise content the way NLSR and OSPFN do. In both cases RUs are sent without RIs that request them. LSR and DCR use the same time period to refresh their routing structures, and each simulation ran for 10 different seed values. We used static networks of 100 nodes, with nodes being uniformly distributed in the network to avoid disconnected nodes. The transmission of an update to all neighbors of a node is counted as a single transmission, and an update message in DCR and LSR carries all the updated distances or link states, respectively. The number of nodes requesting NDOs is increased from 10% to 40% of the 100 nodes of the network. Each node originally publishes two NDOs, and a node that caches an NDO that it requested advertises the NDO.

Table I shows the average number of control packets generated per node with DCR and LSR. As the rate of content requests increases and NDO replicas proliferate, the signaling incurred by LSR becomes excessive. On the other hand, with DCR, all updates fit in a single message and routers advertise only their distances to the nearest instances of NDOs. As a result, the signaling overhead of DCR remains constant independently of the number copies per NDO.

Table 1. Control-plane overhead

Approach	10%	20%	40%
DCR	100	100	100
LSR	699	1442	2269

8. Data-Plane Efficiency

DCR operates independently of the data-plane mechanisms used for forwarding content and content requests in an ICN. However, the multipath loop-free routing provided by DCR has a major impact on the efficiency of the data plane in an ICN.

For example, in CCN [7] and more recently NDN [39, 29], each router maintains a forwarding table (FIB), a pending interest table (PIT), and a content store. The PIT has an entry for each Interest that has been forwarded and waiting for data to return. For a given Interest properly identified, the PIT states the "faces" (interfaces) over which the Interest has been received and the faces over which the interest has been forwarded. At each router, the FIB is populated from the routing table maintained by a routing protocol and states one or multiple next hops to a prefix. To obtain content in CCN and NDN, Interests are forwarded over the routes defined in the FIBs, and data packets are sent back along the reverse

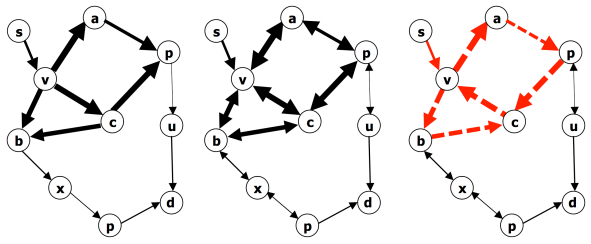
paths traversed by Interests.

It has been argued that, independently of the routing protocol used in the control plane, an Interest cannot loop if it is identified using the name of the content being requested and a nonce (e.g., see [39], Section 2.1). However, we show below that this need not be the case even if a nonce were to denote an Interest uniquely. Interest looping is a function of the way in which forwarding strategies for Interests interact with the FIB entries populated by the routing protocol.

Fig. 5 shows an example network of ten routers in which all links have unit cost and router d has advertised a given NDO j. For simplicity, the thickness of a link indicates its available bandwidth for data in both directions; hence, a thin link indicates that the link is perceived as congested. An arrowhead in the figure indicates the direction in which Interests can traverse a link based on the FIB entries in the routers induced by DCR and NLSR.

As Fig. 5(a) illustrates, the FIB entries obtained with DCR consist of next hops to prefix j that must belong to loop-free paths to the prefix, because routers coordinate their routing updates based on SOC. Interests flow over loop-free paths towards prefix j regardless of the forwarding strategy used in the data plane, the congestion state over links, forwarding decisions made at each router, and the length of time allowed for an Interest entry to be kept in the PIT.

With NLSR [22], each router uses the information it knows about the topology and the locations of a prefix to compute multiple paths to the prefix sequentially as described in Section 2. Fig. 5(b) shows the directions in which Interests can be forwarded when NLSR is used to build the FIBs. The FIB entries stored at routers need not define loop-free paths, because routers compute their paths to prefix j without any coordination with one another.



(a) DCR (b) NLSR (c) Interest loops

Figure 5. Multipaths in FIBs and Interest looping

Fig. 5(c) shows a scenario with router s issuing an Interest for j to router v. As a result of forwarding decisions made by b, p and c based on the perceived congestion to their neighbors, the Interest can traverse loop (v, b, c, v) or loop (v, a, p, c, v) as shown, or loop (v, a, p, c, b, v) not shown.

If v simply drops the Interest after receiving it again from c, which is the basic strategy in NDN, then router s must retransmit after a timeout expires or a negative acknowledgment is received. Because of looping, this may incur delays of order $O(N)$, where N is the number of network routers. On the other hand, if v attempts to reroute the Interest, there is no guarantee that the Interest will not loop again. The Interest from s may take eight or nine hops to be delivered after traversing one of the loops, or it may be dropped if its loop traversal leaves no rerouting alternatives. In general, rerouting of Interests in an ICN without loop-free routing in the control plane constitutes an on-demand depth-first search strategy in the data plane, which need not be successful and takes order $O(N)$ to deliver an Interest because of backtracking due to looping [2, 23].

If routing-table loops are allowed in the control plane, then Interest entries should be stored in the PITs long enough to ensure that no Interest that loops can be recirculated in the same loop. This means that Interest entries in the PITs must remain for time durations of order $O(N)$.

Interest flooding in the data plane can be used to ensure the delivery of Interests in order $O(d)$, where d is the network diameter and $d = O(log(N))$ (e.g., see [36]). However, this approach induces communication overhead larger than $O(EC)$, with E and C being the number of links and prefixes, respectively.

We have discussed Interest looping problems resulting from forwarding strategies in the data plane using stable FIBs populated with a routing protocol that is not loop-free. However, Interest looping can also occur with inconsistent FIBs that do not correspond to loop-free paths. Many approaches for loop-free multipath routing have been proposed [28, 37, 41] that work much better than single-path routing or equal-cost multipath routing. However, DCR is the first to augment the desired loop-free multipath routing functionality to the case of ICNs in which content can be replicated arbitrarily. DCR offers efficiency gains in the data plane that are inherent in using loop-free multipath routing, and does this while attaining high efficiency in the control plane.

Using DCR in the control plane reduces content-delivery delays by reducing the time that Interests take to reach routers that can answer them, and also allows PIT timers to be of order $O(d)$. This is the case even in the presence of topology changes or the relocation of prefixes in the network. Because DCR guarantees that routing tables and FIBs are always loop-free, routers can continue to forward Interests over remaining paths as stated in the FIBs while new routes are computed in the control plane. Furthermore, the time incurred by DCR to recompute loop-free paths to prefixes is $O(x)$, where $x \leq d$. This is much shorter than the time complexity of $O(N)$ that can be incurred in the rerouting of Interests in the data plane when FIB entries are not part of loop-free paths.

9. Conclusions

We introduced *Distance-based Content Routing* (DCR), the first approach for name-based content routing in ICNs based solely on distances to NDOs or name prefixes. DCR does not require any routing information about the physical topology of the network or information about all the replicas of the same content to provide multiple shortest paths to the nearest replica of content, as well as to all or some instances of an NDO or

name prefix. DCR was shown to be loop-free at every instant and to converge to the shortest paths to the closets replicas of content.

DCR has smaller time complexity and orders of magnitude smaller communication and storage complexities than name-based routing approaches that require information about all content replicas. It has the same communication complexity of an ideal DHT approach when content replicas proliferate, and has smaller time and storage complexity than a DHT approach. A simulation experiment was used to illustrate the substantial savings in signaling overhead derived from having to communicate updates about the nearest copies of content, rather than all copies. In addition, DCR was shown to enable efficient and simple data planes, because it provides multiple loop-free paths over which content requests can propagate.

A detailed characterization of the performance of DCR and other name-based routing approaches is needed to address: the interaction of the control and data planes; the end-to-end delays incurred in obtaining NDOs; the impact of network size, number of NDOs, and naming hierarchy on the performance of the protocols; and the efficiency with which multi-point communication is supported.

Name-based routing approaches with communication and storage complexities smaller than order $O(EC)$ and $O(C)$ (with E and C being the number of links and name prefixes in the ICN, respectively) should be investigated.

References

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

[2] B. Awerbuch, "A New Distributed Depth-First-Search Algorithm," *Information Processing Letters*, pp. 147?150, 1985.

[3] T. Ballardie, P. Francis, and J. Crowcroft, "Core Based Trees (CBT)," *Proc. ACM SIGCOMM 93*, Oct. 1993.

[4] M.F. Bari et al., "A Survey of Naming and Routing in Information-Centric Networks," *IEEE Commun. Magazine*, July 2012, pp. 44–53.

[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., "A Routing Scheme for Content-Based Networking," *Proc. IEEE Infocom '04*, March 2004.

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

[8] Content Mediator Architecture for Content-aware Networks (COMET) Project [online]. http://www.comet-project.org/

[9] A. Detti et al., "CONET: A Content-Centric Inter-networking Architecture," *Proc. ACM ICN '12*, 2012.

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

[11] 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, 2007.

[12] J.J. Garcia-Luna-Aceves, "Routing to Multi-Instantiated Destinations: Principles and Applications," *Proc. IEEE ICNP 2014*, Oct. 2014.

[13] M. Gritter and D. Cheriton, "An Architecture for Content Routing Support in The Internet," *Proc. USENIX Symposium on Internet Technologies and Systems*, Sept. 2001.

[14] A. Gupta, B. Liskov and R. Rodrigues, "Efficient Routing for Perr-to-Peer Overlays," *Proc. ACM NDSI '04*, March 2004.

[15] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proc. ACM MobiCom '00*, 2000.

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

[17] M.J. Johnson, "Updating Routing Tables after Resource Failure in a Distributed Computer Network," *Networks*, Vol. 14, No. 3, 1984.

[18] T. Koponen et al., "A Data Oriented (and Beyond) Network Architecture," *Proc. ACM SIGCOMM 07*, 2007.

[19] B.N. Levine and J.J. Garcia-Luna-Aceves, "Improving Internet Multicast Using Routing Labels," *Proc. IEEE ICNP '97*,1997.

[20] Q. Li and J.J. Garcia-Luna-Aceves, "Efficient Content Routing in MANETs Using Distances to Directories," *Proc. 2014 IEEE INFOCOM Workshop on Name-Oriented Mobility*, May 2 2014.

[21] E.L. Madruga and J.J. Garcia-Luna-Aceves, "Core Assisted Mesh Protocol for Multicast Routing in Ad-Hoc Networks," U.S. Patent 6,917,985, 12 July 2005.

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

[23] S. Makki and G. Havas, "Distributed Algorithms for Constructing a Depth First Search Tree," *Proc. ICPP '94*, Aug. 1994.

[24] M. Marina and S. Das, "On-Demand Multipath Distance Vector Routing in Ad Hoc Networks," *Proc. IEEE ICNP '01*, 2001.

[25] J.M. McQuillan, "Enhanced Message Addressing Capabilities for Computer Networks", *Proceedings of the IEEE* , Nov. 1978.

[26] Mobility First project [online]. http://mobilityfirst.winlab.rutgers.edu/

[27] M. Mosko and J.J. Garcia-Luna-Aceves, "Multipath Routing in Wireless Mesh Networks," *Proc. IEEE WiMesh '05*, Sept. 2005.

[28] S. Murthy and J.J. Garcia-Luna-Aceves, "Congestion-Oriented Shortest Multipath Routing," *IEEE Infocom '96*, March 1996.

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

[30] M. Parsa and J.J. Garcia-Luna-Aceves, "A Protocol for Scalable Loop-free Multicast Routing," *IEEE JSAC*, April 1997.

[31] Publish Subscribe Internet Technology (PURSUIT) Project [online]. http://www.fp7-pursuit.eu/PursuitWeb/

[32] J. Rajahalme et al., "On Name-Based Inter-Domain Routing," *Computer Networks*, pp. 975-985, 2011.

[33] 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 23, 2009

[34] Scalable and Adaptive Internet Solutions (SAIL) Project [online]. http://www.sail-project.eu/

[35] Scalable Network Technologies, *Qualnet*, http://web.scalable-networks.com/content/qualnet.

[36] I. Solis and J.J. Garcia-Luna-Aceves, "Robust Content Dissemination in Disrupted Environments," *Proc. ACM CHANTS 08*, Sept. 2008.

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

[38] L. Wang et al., "OSPFN: An OSPF Based Routing Protocol for Named Data Networking," Technical Report NDN-0003, 2012.

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

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

[41] W. 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 5881243, 1999.