Routing to Multi-Instantiated Destinations: Principles and Applications

J.J. Garcia-Luna-Aceves
Palo Alto Research Center, Palo Alto, CA 94304
Computer Engineering Department, University of California at Santa Cruz, CA 95064

***Abstract*—Prior solutions for routing to multi-instantiated destinations (e.g., Internet multicasting and anycasting, and routing in information centric networks) simply adapt existing routing algorithms designed for single-instance destinations, or rely on flooding techniques. As a result, they are unnecessarily complex and incur excessive overhead. A new approach for routing to multi-instantiated destinations is introduced, and MIDR (Multiple Instance Destination Routing) is presented as an example of the approach. MIDR uses only distance information to multi-instantiated destinations, without routers having to establish overlays, know the network topology, use complete paths to destination instances, or know about all the instances of destinations. MIDR enables routers to maintain multiple loop-free routes to the nearest instances of any given destination, as well as to some or all instances of the same destination. It is shown that MIDR provides multiple loop-free paths to destination instances, and that is orders of magnitude more efficient than traditional approaches based on routing to single-instance destinations. MIDR can be used in name-based content routing, IP unicast routing, multicasting, and anycasting.**

I. Introduction

The first routing protocol for packet switching networks can be traced back to Baran's original design of packet switching at the RAND Corporation in the 1960s [5]. The "hot potato heuristic routing doctrine" described by Baran is the first instance of distance-vector routing in which the number of hops traversed by messages originated by destination nodes determine the preferred paths to them over time.

The routing protocols developed for early public data networks and the ARPANET in the 1960s and 70s also assumed that destination nodes advertise either distances to themselves or adjacent links to the rest of the network. The distributed algorithms used as part of these protocols were more efficient than Baran's proposal; they consisted of either a distributed version of the Bellman-Ford algorithm using vectors of distances, or Dijkstra's shortest-path-first algorithm using flooding of link-state updates [46].

The routing protocols developed for the Internet for routing within autonomous systems [35] (e.g., RIP and OSPF) evolved directly from the early routing protocols designed for the ARPANET; and BGP [35] supports routing across autonomous systems by using path information to detect loops. The key differences between Internet routing and routing within a packet switching network is that Internet destinations are network address ranges, and the routers attached to a network advertise a link, path, or distance for that network. While Internet routing protocols must build routing tables regarding destinations that are not routers, the distributed algorithms in which they are based were designed assuming that destinations are nodes of packet-switching networks. This choice seems trivial at first glance, because a router to which a destination is attached can report the presence of the destination. Furthermore, as Section II summarizes, various forms of routing to multi-instantiated destinations have been proposed or implemented to date based on algorithms designed for routing to single-instance destinations. Examples are multicast routing (e.g., [4], [14]), anycasting (e.g., [3], [25], [29], [50], [56]), routing to multi-homed networks in BGP [35], and routing to replicated content in information centric networking (ICN) architectures (e.g., [1], [6], [52]).

/978-1-4799-6204-4/14$31.00

Unfortunately, using algorithms designed for routing to single-instance destinations results in unnecessary signaling overhead and complexity, limited functionality, and in some cases incorrect solutions. This has large implications for the future Internet, given that the nature of Internet destinations is shifting from individual machines to content objects that may be replicated dynamically, services, classes of resources, groups of things or individuals that are distributed over networks, and even networks that may be partitioned.

We argue that traditional notions of unicast, anycast, and multicast routing need to be revised in favor of an integrated framework for routing in which any destination may have multiple instances. Section III presents such a framework. In a nutshell, routing to multi-instantiated destinations entails establishing a lexicographic ordering of distances to destinations in which the identifiers of the routers to which destination instances are attached are part of the attribute set used to define the ordering. Many concrete approaches are possible based on this simple observation.

Section IV presents *Multiple Instance Destination Routing* (MIDR) as an example. This is the first routing algorithm for multi-instantiated destinations and uses only distance information about destinations. MIDR also provides an integrated approach for routing to *some or all* instances of a given destination by building a *Multiple Instance Destination Spanning Tree* (MIDST). MIDR does this without having to rely on flooding or the pre-selection of a node as a core for a group of destinations as it is done in receiver-initiated approaches to multicast routing. MIDR does not require routers to know the network topology, path information, routes to all network sites, or all the instances of any given destination.

Section V shows that MIDR provides multiple paths to destination instances without ever creating a routing-table loop, that it converges to shortest paths to the nearest instances of

destinations, and that it establishes an acyclic graph to reach some or all instances of a given destination. Section VI compares the communication and time complexities of MIDR with traditional routing approaches applied to multi-instantiated destinations. MIDR 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 sites where destination instances are present. Section VII discusses how our unifying framework for routing to multi-instantiated destinations can be applied to improve or redefine IP routing, IP multicasting and anycasting, and name-based content routing.

II. Related Work

A. Routing Algorithms

Many routing algorithms have been developed over the years to ensure that the distributed computations incurred in updating all routing tables in a network with correct entries for each destination terminate within a finite time. Examples include: using sequence numbers to identify the most recent update information (e.g., [7], [18], [34]), using path information to detect or block loops (e.g., [10], [17]), or using diffusing computations to avoid loops (e.g., [16]). In addition, many approaches have been proposed that attain routing to single-instance destinations over multiple paths (e.g., [32], [37], [38], [55]). However, attaining loop-free routing does not solve the problem of establishing valid routes to a destination that can be replicated arbitrarily in a network. In fact, none of the algorithms reported to date for routing to single-instance destinations can be used without change to enforce loop-free routing to multi-instantiated destinations. This is because the ordering they establish with respect to each destination inherently assumes that the destination corresponds to a single node of the graph representing a network.

Consider as an example the loop-freedom condition defined for sequence numbered distances. A destination j is the only node that can increase the sequence number used to validate reported distances to j. A router $i \neq j$ stores the most recent sequence number associated with j, and can select a neighbor k as a next hop to destination j if k reports either (a) a larger sequence number from j than i currently stores, or (b) the same sequence number as i stores and a smaller distance to j than i currently attains. Unfortunately, the ordering attempted by the use of sequence numbers originated by a given instance of j is invalidated by the existence of other instantiations of j issuing sequence numbers independently of one another.

Similar problems exist with algorithms based on diffusing computations or path information. Contrary to what some prior work on anycasting has assumed [25], traditional distance-vector routing algorithms *cannot* ensure correct anycasting. Furthermore, the problem exists even when routing is based on link-state algorithms. In this case, the multiple instances of a given destination j appear to be a single node in a graph, which leads to routers having inconsistent topology maps that results in long-term or permanent loops (see Fig. 5 in [25]).

There are only three possible ways to support routing to multi-instantiated destinations using the algorithms designed to date for routing to single-instance destinations. The type of approach used depends on the amount of information known to the sources of data packets.

If sources do not know which nodes constitute instances of the destination (i.e., a router with one or more receivers of the multicast group attached to it), then the sources must flood the entire network with signaling packets; this approach has been called "sender initiated." The opposite approach to the above case consists of each source node knowing all the nodes to which instances of the intended destination attach. In this case, signaling from each destination instance must reach all potential sources, and sources must have enough information to compute shortest paths to all destination instances.

The alternative to the above two approaches consists of designating a node to serve as the representative (i.e., the address) of the set of destination instances. All nodes maintain routes to the representative node of the multi-instantiated destination and are also capable of learning the mapping from the identifier of the multi-instantiated destination to the identifier of the representative node. Each node with a destination instance establishes a route to the representative node, so that information can flow towards that instance. A source sends its data packets towards the representative node, and either that node or another relay (depending on the specific solution) ensures that the data packets are forwarded to all destination instances based on the routes established previously.

There are three main examples of prior work on routing to multi-instantiated destinations based on the above three approaches: routing to all members of a multicast group denoted by a group identifier; routing to any one member of an anycast group; and name-based routing of content. We summarize the prior work in each of these areas next.

B. Multicast Routing Protocols

McQuillan [33] proposed the first link-state routing approach to support multicasting. In essence, each node floods link-state advertisements (LSA) stating the sate of adjacent links and the existence of receivers for different multicast groups. Given this information, each node can compute shortest routes to all receivers of each multicast group. Multicast OSPF [35] constitutes a more recent example of this approach.

Deering [13] described an approach to multicasting similar to what McQuillan introduced, and also introduced a sender-initiated approach to multicast routing based on distances. Various approaches can be used for flooding signaling packets from each source and for nodes with no receivers of a multicast group to send prune messages towards sources. Examples of this approach are ODMRP [28] and PIM dense mode [14].

The first approach for multicast routing based on representatives was introduced by Ballardi, Francis and Crowcroft [4], and many subsequent examples have been proposed (e.g., [14], [30], [40]). A node serves as the address of a multicast group and is called the core of the group. Nodes maintain routes to all network nodes and hence to all cores, and are able to learn the mapping from the multicast group address to the address of the core. Each receiver of a multicast group sends join requests towards the core of the group to establish a shared multicast tree spanning all the receivers and the core. Sources simply send data packets towards the core, and data packets are sent to all receivers of the multicast group over the multicast tree.

C. Anycast Routing Architectures and Protocols

McQuillan [33] was arguably the first to address routing problems in packet switching networks associated with destinations with multiple instances. He discussed the concepts of logical addressing (which is now called anycasting), broadcast addressing (or intelligent flooding), and group addressing (which is now called multicasting). He also proposed approaches to handle these three addressing methods based on the use of directory nodes supporting indirection services and routing algorithms designed for single-instance destinations.

Interestingly, all the solutions that have been proposed for anycasting over the years are similar in nature to McQuillan's original proposals, assume the use of routing algorithms designed for single-instance destinations, and none of them is able to truly address the scaling problems in anycast routing associated with the handling of anycast addresses. Patridge et al. [41] defined logical addressing in the context of the Internet Protocol (IP) and called it host anycasting service.

In the application-layer anycasting approach [56], resolvers must maintain the list of all the addresses that belong to an anycast group and a metric database of information associated with each anycast group member. Ballani and Francis [3] proposed the use of an overlay of anycast proxies (AP) sharing the same IP address, with each AP required to know all other APs, under the assumption that only a few APs are needed for Internet-scale operation; multiple APs act as intermediaries between clients and anycast targets. The Global IP-Anycast (GIA) framework [25] assumes that anycasting within domains can be supported using traditional intra-domain routing protocols and uses an on-demand query-based inter-domain routing protocol based on broadcast signaling. IPv6 [50] provides limited support for anycasting, in that it allows the use of anycast addresses, but there are no known methods to support anycast routing in a scalable manner. The *i*3 approach [48] to anycasting consists of using an overlay of servers that map identifiers to actual IP addresses, and clients contact target destinations through the overlay; however, no new approaches are introduced for anycast routing.

D. Name-Based Content Routing Protocols

Name resolution and routing of content are essential in all information centric network (ICN) architectures [1], [6], [52], and several approaches have been proposed to support content routing based on the names of named data objects (NDO) that may be replicated in a network. Interestingly, all these approaches are based on algorithms designed for routing to single-instance destinations.

Like sender-initiated approaches for multicasting, some content routing approaches rely on flooding of content requests to cope with the fact that nodes requesting content by name do not know the locations if copies of content. Directed Diffusion [22] 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 [47] uses an approach similar to directed diffusion for named-based content routing in ad hoc wireless networks subject to connectivity disruption. Nodes use opportunistic caching of content and flood interests persistently within and across connected network components.

A number of approaches are based on maintaining routing information to all replicas of content by means of path-vector algorithms or link-state algorithms. Gritter and Cheriton proposed the Name-Based Routing Protocol (NBRP) [20] as an extension of BGP. The CBCB (combined broadcast and content based) routing scheme for content-based networking [9] is an example of content-routing similar to the receiver-initiated approach to multicasting. 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 based on predicates are sent between consumers and producers of content over the tree(s) established in the network.

DONA [27] 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.

The routing approach in the Mobility First project [36] requires using either network addresses or source routing or partial source routing. Several ICN projects have adopted content routing modalities based on the link-state routing approach (e.g., [11], [12], [15], [23], [39], [45]). NLSR [31] is the most recent example of name-based content routing based on complete topology information. Routers flood link-state advertisements (LSA) that describe the state of physical links or the name of prefixes of content for which they have local copies.

Some ICN projects (e.g., [42], [45]) adopt content routing modalities based on distributed hash tables (DHT) running in overlays over the physical infrastructure to accomplish name-based routing. A destination is assigned a home location in the DHT that nodes can determine by using a common hash function from the name space of destinations to the name space of nodes in the DHT. DHT nodes can cache known mappings to improve efficiency, and the DHTs are built using underlying routing protocols that discover the network topology.

III. Routing to Multi-Instantiated Destinations

A multi-instantiated destination is a non-empty set of entities denoted with the same unique identifier consisting of a string of alphanumeric symbols. The identifier can be drawn from a flat or hierarchical naming space and can have a fixed length or variable length depending on the application. An entity can be an information object or collection of objects, a thing or class of things, a process, a service, a network, an end system, or an intermediate system. It may be part of one or more destinations, each denoted with a different identifier.

The objective is to support routing to multi-instantiated destinations in a way that permanent loops are impossible, and without each node having to flood the network, know about all destination instances, or rely on a pre-defined representative node. The services to be provided consist of: (a) reaching the nearest instances of a destination, and (b) reaching a subset of instances or all the instances of a destination.

Supporting loop-free routing to the nearest instances or to all or some instances of multi-instantiated destinations

can be done using distance information in a manner that scales in much the same way as routing to single-instance destinations. Doing so requires three basic functionalities with respect to a given destination. First, using common rules, routers must establish lexicographic orderings with respect to the destination instances that are nearest; different routers may order themselves with respect to different instances. Second, a router must report as its distance to the destination the value of its distance to the nearest instance, and use a common lexicographic ordering to select the nearest instance to use. Third, routers establish a lexicographic ordering among all destination instances to permit the selection of a single destination instance as the root of a routing structure used for communication with all or some instances of the destination.

Let a router that originates an advertisement for a given destination instance that is locally available be called an *anchor* of the destination. The routing functionality summarized above can be attained by: (a) having each router include the identifier of the anchor corresponding to the preferred nearest destination instance as an attribute of the distance it reports for the destination; and (b) using loop-free routing constraints that make use of this information to establish lexicographic orderings to destination instances.

IV. MIDR

We describe MIDR (Multiple Instance Destination Routing) as an example of routing to multi-instantiated destinations. The operation of MIDR assumes that each router is assigned a unique identifier and each destination is assigned a unique identifier. Identifiers may be hierarchical or flat, and they may be user friendly or not. For a given router, the *root anchor* of a destination is the anchor whose identifier is lexicographically smallest among all the anchors it knows for the destination.

MIDR relies on sequence numbers created by the anchors of destinations to determine which updates carry the most recent information about a destination, enforce loop-free routing, and maintain a lexicographic ordering among distances reported by different anchors of the same destination.

The lexicographic value of an identifier i is denoted by $|i|$. The set consisting of router i and all its neighbor routers is denoted by N^i, it can be viewed as the neighborhood node set. The set of next hops of router i for destination j is denoted by S^i_j. If i is an anchor for destination j, then $S^i_j = \{i\}$. 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.

A. Information Stored and Exchanged

Router i maintains four tables: (1) a *link cost table* (LT^i) listing the cost of the link from router i to each of its neighbors; (2) a *neighbor table* (NT^i) stating routing information reported by each neighboring router for each destination; (3) a *routing table* (RT^i) that stores routing information for each destination; and (4) a *multipoint routing table* (MRT^i) that stores routing information created for those destinations requiring multipoint communication support. The entry in LT^i for link (i,k) with $k \in N^i - \{i\}$ consists of the identifier of neighbor k and the cost of the link to it (l^i_k).

The information stored in NT^i for destination j from each neighbor $k \in N^i$ consists of routing information for the nearest anchor and the root anchor of the destination. The routing information for the nearest anchor consists of: the distance from router k to j (d^i_{jk}); the identifier of an anchor (a^i_{jk}) where j is present, which can be i itself; and the sequence number created by a^i_{jk} for j (sn^i_{jk}). The routing information for the root anchor of the destination consists of: the identifier of the root anchor (ra^i_{jk}), which can be i itself; 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 i is an anchor for j, then $d^i_{ji} = 0$ and $a^i_{ji} = i$.

The row for destination j in RT^i specifies: (1) the identifier of j; (2) the routing update information for j (RUI^i_j); (3) the list of neighbors that are valid next hops (S^i_j), which includes a neighbor s^i_j that offers the shortest distance to j; and (4) an anchor list (A^i_j) with a tuple for each different anchor currently reported by neighbors in S^i_j. 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.

The information in RUI^i_j consists of: (1) 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}); (2) the distance from i to j (d^i_j); (3) the anchor of j that has the smallest name among those that offer the shortest distance to j (a^i_j); and (4) the sequence number created by a^i_j for j (sn^i_j).

The entry for destination j in MRT^i states: the identifier of j; the multipoint update information for destination j (MUI^i_j); and the list of neighbor routers that have joined the MIDST for the destination ($MIDST^i_j$). In turn, 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 destination j (rsn^i_j).

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

Router i updates NT^i_{jk} after any input event affecting the information stored for j from neighbor k. It updates NT^i_{jk} with the information reported by k for j only if sn^i_{jk} is the most recent sequence number known from a^i_{jk}.

B. Routing to Nearest Instances of Destinations

MIDR establishes a lexicographic ordering of the distances to any given destination reported by routers based on the distance values, anchor identifiers, and sequence numbers created by the anchors. Instead of remembering the most recent sequence number for a given destination, a router maintains the sequence numbers created by *all* the anchors currently reported by its neighbors. The information about a given anchor of a destination is deleted after a finite time that is long enough to

ensure that up-to-date information about valid anchors of the destination is received, before anchor information is deleted.

A router can select neighbors as next hops to destinations only if they report up-to-date information and offer shorter distances to the destinations than the router itself, or the same distances but have lexicographically smaller identifiers. To address the case in which routers are unable to find viable next hops to some destinations, anchors send updates about their destinations periodically and increment the sequence numbers they assign to their destinations. Let A_j^i be the set of anchors known to router i for destination j. The following condition is sufficient to ensure that no routing-table loops are formed when routers change their next hops.

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

$$k \in \{\ v \mid v \in N^i \wedge [\ \forall m \in A_j^i\ (\ a_{jv}^i \neq m\ \vee\ sn_{jv}^i \geq sn(m))\ ]\ \} \quad (1)$$

$$(\ d_j^i < \infty \wedge\ [\ d_{jk}^i < d_j^i \vee (\ d_{jk}^i = d_j^i\ \wedge\ |k| < |i|\ )\ ]\ ) \vee \quad (2)$$
$$(\ d_j^i = \infty\ \wedge\ d_{jk}^i < d_j^i\ \wedge$$
$$\forall v \in N^i - \{k\}(\ (\ d_{jk}^i + l_k^i < d_{jv}^i + l_v^i\ ) \vee$$
$$(\ d_{jk}^i + l_k^i = d_{jv}^i + l_v^i \wedge |k| < |v|\ )\ )$$

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

For each destination j, router i determines which routers in N^i report valid sequence numbers created by anchors of the destination, and then determines those routers that can be next hops (i.e., belong to set S_j^i) using SOC. If at least one router in N^i is found that satisfies SOC, router i computes $d_j^i = d_{min} = Min\{d_{jm}^i + l_m^i \mid m \in S_j^i\}$, where S_j^i is the set of routers in N^i that satisfy SOC. Router i then sets $a_j^i = a_{jq}^i$ and $sn_j^i = sn_{jq}^i$, where $q \in S_j^i$, $d_{jq}^i + l_q^i = d_{min}$, and $|q| \leq |m|$ for any $m \in S_j^i$ such that $d_{jm}^i + l_m^i = d_{min}$.

Router i schedules an update $U_j^i[ud_j^i = d_j^i,\ ua_j^i = a_j^i,\ usn_j^i = sn_j^i]$ to neighbor k if $d_j^i < \infty$ and $ud_{jk}^i = \infty$, so that neighbor k can satisfy SOC by making i part of S_j^k. Router i schedules an update U_j^i to all its neighbors if it makes any changes to d_j^i, a_j^i, or sn_j^i after an input event. If SOC is not satisfied at router i after an input event, then router i resets $S_j^i = \emptyset$; sets $ud_j^i = d_j^i = \infty$, $ua_j^i = a_j^i = null$, and $usn_j^i = sn_j^i = 0$; and schedules an update U_j^i to all its neighbors.

Fig. 1 illustrates how MIDR routes to the nearest replica of a single destination when three routers (d, o, and u) serve as the anchors of the destination and each link has unit cost. It is assumed that all routers have received the most-recent sequence numbers from the anchors they know (d, o, or u) for the destination. The first tuple listed next to each node indicates the shortest distance from the router to the destination and the anchor with the smallest identifier at that distance. Updates from each router state only the preferred anchor (e.g., the update from node e states d as the anchor and distance 2 to it). Each additional tuple next to a router, if any, states an alternate anchor for the destination and the distance to it. The arrowheads in the links between nodes indicate the next-hop neighbors of nodes, and the arrow to the lexicographically smallest next hop is shown with the color of the anchor.

As Fig. 1 shows, updates from an anchor propagate only as long as they provide routers with shorter paths to destinations. In the example, no routing update about the destination propagates more than four hops, even though the network diameter is eight. In general, independently of how many anchors exist in a network for given destination, a router only has as many active anchors for the destination as it has neighbors, because each router reports only the best anchor it knows for each destination. Even in this small network of just 23 routers, several routers have multiple paths to the destination; all links can be used to forward data packets; very few routers know about all the anchors of the destination; and traversing any possible directed path to the destination in Fig. 1 necessarily terminates at d, o, or u, without traversing a loop.

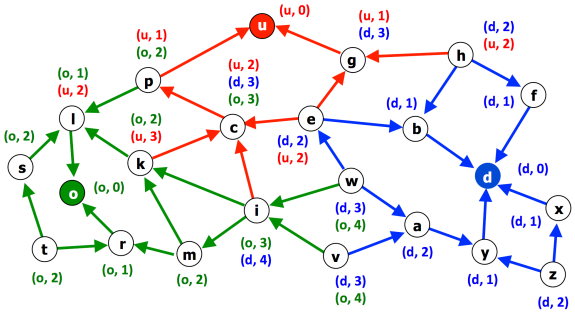


Fig. 1: Routing to the nearest instance of a destination

C. Routing to Some or All Instances of a Destination

MIDR supports routing to *all or some* anchors of the same destination by first connecting all anchors of a given destination by means of a *Multiple Instance Destination Spanning Tree* (MIDST) for the destination.

To build the MIDST for destination j, routers select the *root anchor* of the destination to be that anchor that has the lexicographically smallest identifier among the anchors from whom they currently have valid distances to the destination. Therefore, at each router i, $ra_j^i = Min\{a_{jk}^i, ra_{jk}^i \mid k \in N^i\}$. If $i \neq ra_j^i$ then the distance from router i to its root anchor ra_j^i is $rd_j^i = rd_{js}^i + l_s^i$, where $s \neq i$ is the next hop to ra_j^i. If $i = ra_j^i$ then $rd_j^i = 0$.

The MIDST is established in a distributed manner using the distance updates exchanged among routers. The following Root-Anchor Notification Condition (RNC) is used by router i to determine the neighbors to which it forwards updates about its root anchor for destination j when it also knows multiple other anchors for the same destination. Routing updates are exchanged only by routers located between the root anchor and other anchors. Eq. (3) in RNC states that neighbor k has reported to router i an anchor or root anchor that is different than the root anchor currently assumed by router i. Eq. (4) in

RNC 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 known to router i that is not the root anchor of destination j.

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 $k \in N^i - \{i\}$ such that

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

$$\forall v \in N^i (a^i_{jv} = i) \ \vee \ \forall v \in N^i - \{k\} (\ a^i_{jk} \neq a^i_{jv} \ \vee \tag{4}$$

$$(\ d^i_{jk} + l^i_k < d^i_{jv} + l^i_v \ \vee \ [\ d^i_{jk} + l^i_k = d^i_{jv} + l^i_v \wedge |k| < |v|\ ]\ )\ )$$

Using RNC, a router sends distance updates about its root anchor only 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. Distance updates about the root anchor propagate to all other anchors of the same destination, and only over the preferred paths between the root anchor and other anchors. Router i uses the following condition to select the next hop to ra^i_j.

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

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

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

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

ROC uses the same lexicographic ordering based on sequence numbers proposed in several prior routing protocols based on sequence numbers (e.g., [32]), given that a root anchor is the intended destination and cannot be replicated in the network. Eq. (5) states that router k reports as its root anchor the anchor of destination j known to router i that has the lexicographically smallest identifier, and an up-to-date sequence number from such an anchor. Eq. (6) states that router k must offer the shortest distance to the root anchor, which may be 0 if i is the root anchor. 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.

Fig. 2 shows how routing information regarding the root anchor of a destination is propagated. In the example, router d has the lexicographically smallest identifier among all the anchors of the destination. Routers g, c and i know about d and other anchors with larger identifiers than d, and assume that d should be the root anchor for the destination. Accordingly, each of those routers sends an update about d to the best next hop to each other anchor it knows. For example, g sends an update to u regarding d being the root anchor of the destination, because u is its best next hop to u. Similarly, c sends an update to p and to k regarding d being the root anchor of the destination, because they are its best next hops to u and o, respectively. A router that receives an update about d being the root anchor sends its own update to each best next hop to each other anchor it knows. This way, updates about d reach the other two anchors of the destination (o and u).

The dashed blue lines in Fig. 2 indicate the links of subpaths between anchor o or u and root anchor d. Each of these subpaths was enabled when routers along the subpath propagated updates regarding d being the root anchor of destination j. Compared to Fig. 1, many more routers know about d than would be the case if routers did not need to contact all or some destination instances other than the nearest ones. However, many routers (i.e., m, o, r, s, t, and u) do not participate in the propagation of updates about d as the root anchor of destination j, and many (i.e., m, r, s, and t) do not even receive updates about d being the root anchor of destination j. This contrasts with the traditional approach to building shared multicast trees, in which all routers would have to have a route to d.

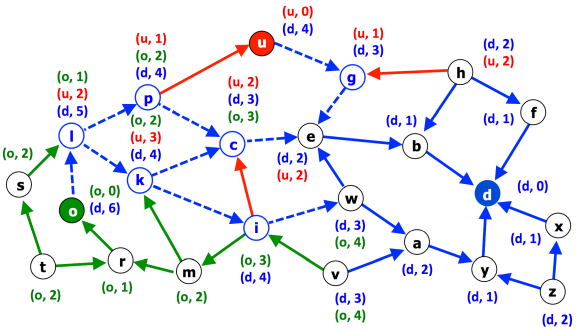


Fig. 2: Maintaining routes to root anchor d

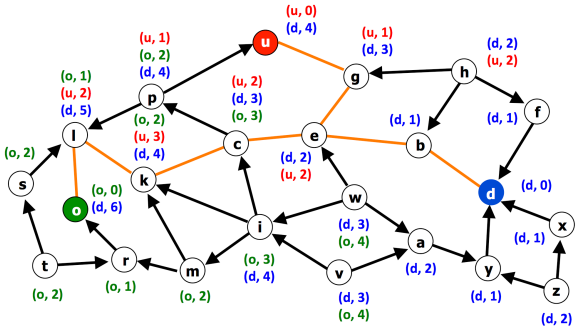


Fig. 3: The MIDST of a destination

When destination j needs a MIDST and an anchor p of j receives updates from any of its neighbors stating that the root anchor of j is a router q such that $|q| < |p|$, then anchor p sends a join request to the neighbor with the lexicographically smallest identifier among all neighbors reporting the shortest distance to root anchor q. Each router forwarding the join request stores an entry for the request for a finite period of time stating the neighbor from which it was received and a unique identifier for the request. The join request traverses the path towards the root anchor q, until it reaches a router x that is already part of the MIDST for destination j. In turn, that router sends a join response over the reverse path traversed by the request. The response makes each relay router processing the request join the MIDST, until the join response reaches the anchor that originated the join request.

Fig. 3 shows the resulting MIDST for the destination in the same example of Fig. 1. Anchors u and o send their join requests towards d to join the MIDST of the destination.

To forward data packets that must be sent to all instances of a destination, routers simply forward the packets towards the nearest anchors of the destination, and the packets are then broadcast over the MIDST of the destination as soon as they reach the first routers that are part of the MIDST. The broadcast mechanism within a MIDST is much the same as in shared-tree

multicast routing protocols; however, we note that no router is required to know all the anchors of the destination to establish the MIDST of a destination, and only routers in the shortest paths between anchors and the root anchor participate in the signaling needed to build the MIDST.

To send data packets to some of the anchors of a destination, a router that is not part of the MIDST simply sends the packets towards the nearest anchor of the destination; the first router in the MIDST that receives the data packets forwards them over the MIDST of the destination using an algorithm aimed at reaching all or some anchors (e.g., [29]).

Specifying when a destination requires the need for a MIDST can be done in a number of ways. One approach is for the name or identifier of the destination 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 destination.

V. MIDR CORRECTNESS

MIDR guarantees that routing-table loops are never formed, even as the instances of destinations and the network topology change. Furthermore, routers running MIDR attain the shortest distances to the nearest instances of each known destination. The following theorems prove that this is the case.

Theorem 1. *No routing-table loops can be formed if routers use SOC to select their next hops to destinations.* □

Proof: The proof is by contradiction. Assume that a routing loop L_j for destination j consisting of h hops is created at time t_L when the routers in the vertex set of L_j change successors according to SOC. Let $L_j = (n_1, n_2, ..., n_h)$, with $n_{i+1} \in S_j^{n_i}$ for $1 \leq i \leq h-1$ and $n_1 \in S_j^{n_h}$.

According to SOC, each hop $n_i \in L_j$ $(1 \leq i \leq h)$ can select its next hops (i.e., $S_j^{n_i}$) in only two ways, depending on whether or not $d_j^{n_i} < \infty$ 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 $d_j^{n_m} = \infty$ when n_m joins L_j by adding n_{m+1} to $S_j^{n_m}$ for each $n_m \in I_j$. By assumption, router n_m uses Eq. (2) in SOC; therefore, $d_{jn_{m+1}}^{n_m} < \infty$ 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 $d_{jq}^{n_m} > d_{jn_{m+1}}^{n_m}$ or $d_{jq}^{n_m} = d_{jn_{m+1}}^{n_m}$ and $|n_m| < |q|$. Therefore, the following relation must hold between $d_{jn_{m+1}}^{n_m}$ and $d_{jn_{m-1}}^{n_m}$ for any $n_m \in I_j$:

$$\begin{aligned} d_{jn_{m-1}}^{n_m} &> d_{jn_{m+1}}^{n_m} \\ \vee\ (\ d_{jn_{m-1}}^{n_m} &= d_{jn_{m+1}}^{n_m} \wedge |n_{m-1}| > |n_{m+1}|\ ) \end{aligned} \tag{8}$$

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

$$\begin{aligned} (\ d_{jn_{m-1}}^{n_m} &= d_{jn_{m+c+1}}^{n_{m+c}} \wedge\ |n_{m-1}| > |n_{m+c}|\ ) \\ \vee\ (\ d_{jn_{m-1}}^{n_m} &> d_{jn_{m+c+1}}^{n_{m+c}}\ )\ \text{ for }\ h \geq c \geq 0. \end{aligned} \tag{9}$$

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

$$d_{jn_i}^{n_i - 1} \geq d_j^{n_i} \tag{10}$$

$$\begin{aligned} d_j^{n_i} &> d_{jn_{i+1}}^{n_i} \geq d_j^{n_{i+1}} \\ \vee\ (d_j^{n_i} &= d_{jn_{i+1}}^{n_i} \geq d_j^{n_{i+1}}\ \wedge\ |n_i| > |n_{i+1}|). \end{aligned} \tag{11}$$

Consider a subset of hops $\{n_l, n_{l+1}, ..., n_{l+k}\} \in L_j - I_j$ that forms a contiguous chain in L_j, where $k \leq h$. It must be the case that either $d_j^{n_{l+i}} > d_{jn_{l+i+1}}^{n_{l+i}} \geq d_j^{n_{l+i}}$ for at least one hop n_{l+i} in the chain, or that $d_j^{n_{l+i}} = d_{jn_{l+i+1}}^{n_{l+i}} \geq d_j^{n_{l+i}}$ and $|n_{l+i}| > |n_{l+i+1}|$ for each hop n_{l+i} in the chain. Accordingly, Eqs. (10) and (11) imply that

$$\begin{aligned} (\ d_{jn_{l+1}}^{n_l} &= d_{jn_{l+k+1}}^{n_{l+k}}\ \wedge\ |n_l| > |n_{l+k}|\ ) \\ \vee\ (\ d_{jn_{l+1}}^{n_l} &> d_{jn_{l+k+1}}^{n_{l+k}}\ )\quad \text{for}\ \ h \geq k \geq 0. \end{aligned} \tag{12}$$

It follows from Eqs. (9) and (12) that using SOC enforces the same lexicography 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 $d_{jn_{i+1}}^{n_i} > d_{jn_{k+1}}^{n_k}$, where $n_k \in L_j$ and $k > i$, then $d_{jn_{m+1}}^{n_m} > d_{jn_{m+1}}^{n_m}$ for any given $m \in \{1, 2, ..., h\}$, which is a contradiction. On the other hand, if $d_{jn_{i+1}}^{n_i} = d_{jn_{k+1}}^{n_k}$ 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 destination j. ■

Assume that MIDR 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 destination instance changes stop taking place after a given time t_T. The following theorem proves that MIDR attains shortest paths to the nearest instances of known destinations within a finite time. To simplify the inductive proof, we assume that the cost of any operational link is 1; however, the same basic approach applies to the case of positive link costs [8].

Theorem 2. *If MIDR is used in network G, the routes to destinations converge to the shortest distances to the nearest anchors of the destinations within a finite time after* t_T. □

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

Basis case: For $k = 1$, consider an arbitrary neighbor of a given anchor α_i of destination 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 $U_j^{\alpha_i}$ from α_i stating $d_j^{\alpha_i} = 0$, $a_j^{\alpha_i} = \alpha_i$, and $sn_j^{\alpha_i} = s(\alpha_i)$ (the most recent sequence number created by α_i) within a finite time after t_T; and it must update $d_{j\alpha_i}^{n_1} = 0$, $a_{j\alpha_i}^{n_1} = \alpha_i$, and $sn_{j\alpha_i}^{n_1} = s(\alpha_i)$.

Because $d_{j\alpha_i}^{n_1} = 0$ and $sn_{j\alpha_i}^{n_1} = s(\alpha_i)$ always satisfy SOC at router n_1 for destination j, it must be the case that $\alpha_i \in S_j^{n_1}$.

Furthermore, any other next hop in $S_j^{n_1}$ must also be an anchor, because the smallest link cost between neighbors equals 1 and hence the smallest value of $d_j^{n_1}$ equals 1. Router n_1 must set $d_j^{n_1} = 1$ and send and update stating that distance, together with the identifier of that anchor and the sequence number it created, after a finite time $t_1 > t_T$. Therefore, the theorem is true for the basis case.

Inductive step: Assume that the theorem is true for any router n_k that is k hops away from its nearest anchors of destination j. It must be true that $d_j^{n_k} = k$ 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 stating $d_j^{n_k}$, $a_j^{n_k}$, and $sn_j^{n_k}$ a finite time after t_k. Accordingly, each neighbor p of n_k must update $d_{jn_k}^p = k$, $a_{jn_k}^p = a_j^{n_k}$, and $sn_{jn_k}^p = sn_j^{n_k}$ a finite time after t_k.

Let router $q \in N^{n_k}$ be such that it is more than k hops away from any anchor of destination j. Because $d_j^{n_k} = k$ is the shortest distance from n_k to destination j after time t_k, router q cannot have any neighbor reporting a distance to j smaller than k after time t_k. Therefore, $d_{jn_k}^q$ must satisfy SOC within a finite time after t_k and router q must make n_k a next hop to destination j a finite time after t_k. Furthermore, any neighbor of q in S_j^q must have reported a distance of k hops to j. Router q selects the anchor in S_j^q with the smallest identifier, and sends an update within a finite time $t_{k+1} > t_k$ stating $d_j^q = k + 1$, together with the identifier 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 destination j must attain a shortest distance of $k + 1$ hops to destination j within a finite time, and the theorem is true. ■

The following theorems address the correctness of MIDR for routing to multiple sites announcing a destination. Theorem 3 shows that routes to root anchors are loop-free. Using the same assumptions introduced for the proof of Theorem 2, Theorems 4 and 5 show that MIDR builds a MIDST for a destination that requires it.

Theorem 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. *If MIDR 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 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 MIDR 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 $|P_{n_2r}| = |P_{mr}| + 1$, 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 5. *If MIDR 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 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 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. ■

VI. Performance Comparison

We compare the performance of MIDR with approaches that require routing information for all instances of each destination. Given that our comparison must apply to any

protocol based on a given approach, we focus on the communication and time complexities of the approaches. Assuming that all transmissions over any given link are successful, the communication complexity (CC) of a routing algorithm is the number of messages that must be transmitted for each router to have correct routing information about all the destinations. The time complexity (TC) of a routing algorithm is the maximum time needed for all routers to have correct routing information for all destinations.

The number of routers in the network is denoted by N and E denotes the number of network links. The number of different multi-instantiated destinations available in the network is denoted by D, the average number of instances of the same destination is denoted by R, the average number of neighbors per router is l, and the network diameter is d.

We assume a network without hierarchical routing, and that a separate control message is sent for any given link-state advertisement (LSA) or distance update. In practice, multiple LSAs and distance updates can be aggregated to conserve bandwidth. However, given that the maximum size of a control message is a constant value independent of the growth of N or D, this aggregation does not change the order size of the overhead incurred by the routing protocols. We consider three approaches for routing to multi-instantiated destinations based on complete information about destination instances.

Link-State Routing (LSR): In this approach, routers send to each other all the information about the network topology and the location of every instance of each destination. A router must transmit an LSA for each adjacent link and each local destination instance, and each LSA must be sent to all the other routers in the network, which may require transmuting the same LSA over E links. Given that a router can be a maximum of d hops from the source of a given LSA, the time and communication complexities of LSR are: $TC_{LSR} = O(d)$; $CC_{LSR} = O(RDE + lNE)$.

Loop-free Distance-Vector Routing (LDVR): In this approach routers maintain loop-free distances to all network nodes and also the location of every destination instance. The traditional distance-vector routing (DVR) approach subject to looping problems (e.g., RIP) cannot be used for routing to multi-instantiated destinations. DVR signaling can traverse long paths and "counting to infinity" can occur, this approach is known to have $O(N)$ time complexity and $O(N^2)$ communication complexity [24].

By contrast, in a loop-free distance-vector algorithm (LDVR), routing updates regarding a destination would take a time proportional to the network diameter to reach all routers, similar to the LSR case [54]. Each router must send a distance update for each local destination instance and for each node to which destination instances are attached. Hence, the time and communication complexities of LDVR are: $TC_{LDVR} = O(d)$; $CC_{LDVR} = O(RDE + NE)$.

Distributed Hash Table (DHT): In this approach, all network routers participate in the DHT and routers must maintain the mapping between a destination name or identifier or a destination instance and the node in the DHT representing the destination or an instance of it. The DHT approach that incurs the least amount of overhead is a virtual DHT with one-hop routing [21], such that routers run the DHT locally and maintain routes to all routers in the network. The communication complexity associated with publishing a destination in the DHT and associating R sites with the destination (to support routing to any or all copies of the destination) is $O(RdD)$ assuming no loops. The communication complexity of maintaining routes to all routers that form the DHT is the same as in the LSR approach, given that link-state routing is typically used. Hence, assuming the fastest possible propagation of routes to all routers, the time and communication complexities of this approach equal: $TC_{DHT} = O(d)$; $CC_{DHT} = O(RDd + lNE)$.

Multiple Instance Destination Routing (MIDR): Independently of the number of instances for a given destination, the information a router communicates for a given destination in MIDR is only its distance to the nearest anchor of the destination, plus the anchor identifier and the latest sequence number created by that anchor.

Given that MIDR does not incur any routing-table loops, any routing information propagates as fast as the shortest path between its origin and the recipient. However, in contrast to routing schemes that require routing updates for each destination instance, as the number of instances of a destination increases, the distance from a router to the nearest replica of the destination (x) decreases, and $x \leq d$. In addition, the number of messages required for all routers to have a correct distance to the nearest instance of a given destination is always $O(E)$, regardless of the number of instances of the destination, because each router simply communicates to each of its neighbors its distance to the nearest instance of a destination and hence updates about a single instance per destination traverse each network link. Given that there are D destinations in the network, the time and communication complexities of MIDR are: $TC_{MIDR} = O(x)$; $CC_{MIDR} = O(DE)$.

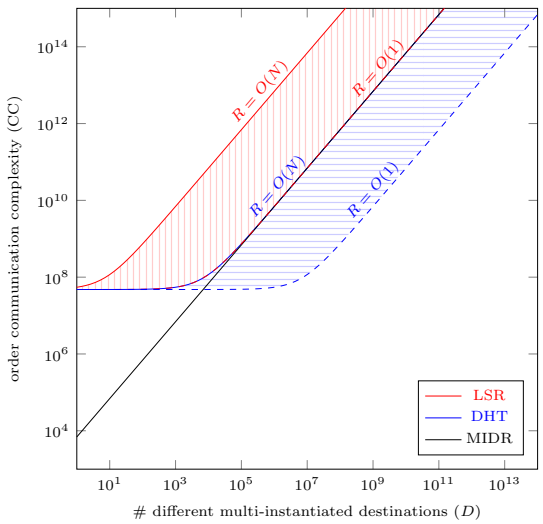


Fig. 4: Signaling overhead as a function of D and R

The performance benefits attained with the proposed framework for routing to multi-instantiated destinations are clear from the results we have stated. As the number of destinations and their instantiations become much larger than the number of nodes, any protocol that requires knowledge of all instances of destinations for correct routing requires order $O(RDE)$ (e.g., LSR) or $O(RDd)$ (e.g., DHT) messages and $O(d)$ steps to

converge. By contrast, MIDR requires order $O(DE)$ messages and $O(x)$ steps (where $x \leq d$) to converge.

Fig. 4 illustrates the signaling overhead of LSR, DHT and MIDR. LDVR is not shown, given that it is very much the same as LSR for large values of D and R. To focus on the effect of D and R, we set $N = 1000$, and assume that l and d are of order $log(N)$, which makes E order $Nlog(N)$. It is clear that MIDR is always orders of magnitude more efficient than LSR. MIDR is more efficient than all the other approaches when either: (a) the number of destinations is of order $O(N)$, which is the case for IP routing; or (b) the number of destinations is much larger than $O(N)$ and destinations are instantiated opportunistically in the network, which is the intended effect in all ICN architectures. As the number of destinations becomes far larger than the number of network nodes and destination instances proliferate, the signaling overhead of MIDR and an ideal DHT approach become the same.

VII. Applications

Changing the routing protocols for the Internet and ICNs to operate based on algorithms designed for multi-instantiated destinations opens up a vast number of opportunities.

A. Autonomic IP Multicasting

Applying MIDR to IP multicast routing is simple. A multicast group denoted by an IP multicast address is a multi-instantiated destination. No changes are needed in the operation of end systems (hosts) or IGMP. A router that learns about the existence of local receivers of a multicast group through IGMP becomes an anchor of the group and starts advertising a 0 distance to the corresponding IP multicast address. A root anchor is elected among all the anchors of the group, and all routers know about the root anchor of a multicast group by using ROC and RNC.

Arguably, the key benefit of MIDR compared to the traditional approaches for IP multicast routing is that it enables autonomic IP multicast routing, in that the construction of multicast routing trees within an autonomous system does not need to pre-define which routers should serve as cores or rendezvous points of multicast groups, and still avoid the need to flood and prune.

In terms of performance, the failure of a router serving as the core of multiple multicast groups constitutes a major disruption if a receiver-initiated multicast routing protocol is used. An alternate core (or rendezvous point) must be pre-assigned or configured, the entire network must be informed of a new core for each multicast group affected, and then the entire multicast tree must be rebuilt for each group affected by the failure. By contrast, a network running MIDR would automatically reconfigure the MIDSTs of the multicast groups affected when anchors stop receiving periodic updates from the root anchor of a multicast group. Each anchor simply applies ROC using a new perceived root anchor and applies RNC to send its updates. As a result, the MIDST of a multicast group may change without the need for external re-configuration. Consider the example shown in Fig. 3, and assume that the multi-instantiated destination represents a multicast group. If anchor d were to fail after the MIDST has been established, then u and o would attempt to become root anchors, because they do not know about the existence of one another. However, according to RNC, routing updates stating o as the root anchor of the multicast group must reach anchor u in a finite time. Thus, anchor u can send a join request towards anchor o to establish the new MIDST consisting of the path (o, l, p, u).

B. More Efficient and Flexible IP Routing and Anycasting

MIDR constitutes the basis for routing within autonomous systems capable of converging as fast as a link-state routing protocol, but incurring fast less signaling overhead. In addition, MIDR supports multi-path Internet routing in which the multi-paths to destinations need not be equal-cost multi-paths, which cannot be attained using any existing intra-domain Internet routing protocol (RIPv2, EIGRP, and OSPF [35]).

With traditional routing to single-instance destinations, a network assigned a given IP address range cannot be physically partitioned, because permanent routing-table loops can occur. By contrast, MIDR allows the same IP address range to be used by multiple physical networks without causing any routing problems. Allowing an IP address or identifier to be multi-homed (i.e., with physical network components or hosts using the same IP address not being connected directly) opens up many new possibilities for hierarchical routing and anycast routing.

MIDR applies equally well when many contiguous routers "speak for" the same identifier or address range. For example, assume that each of the nodes u, o, and d in Fig. 1 are in fact many routers forming a connected network component. Furthermore, MIDR can be used to extend the basic ideas of clusters, which were introduced by Kleinrock and Kamoun [26], and landmarks, which were proposed by Tsuchiya [49]. In essence, a set of clusters or landmarks can share the same identifier, and the rest of the network establishes routing entries to the nearest instances of clusters or landmarks in the set; MIDSTs are then established to connect all clusters or landmarks belonging to the same set (i.e., sharing the same identifier). The approach can be applied recursively using multiple levels of clusters or landmarks, and can be based on distances to identifiers or other information (e.g., [7]).

In contrast to all prior algorithms for single-path and multipath routing, MIDR supports loop-free anycasting, because it is inherent in the computation of loop-free multipaths to destinations. Within small networks, MIDR can support routing to "logical addresses" [33] assigned to sites providing well-known services (e.g., directory services, local printers) or nodes connecting to things of a given class.

Given that MIDR supports routing to different connected components of a network sharing the same identifier, MIDR can support Internet anycasting within an autonomous system (AS) correctly. Furthermore, if information must be shared among two or more network components sharing the same IP address range, or if a specific physical network in the address range must be contacted, then a MIDST can be established to connect all components with the same IP address range.

The scaling problem of Internet anycasting (i.e., having to advertise too many logical addresses across the Internet) can be addressed with MIDR through hierarchical address assignments to anycast groups and the recursive use of MIDSTs.

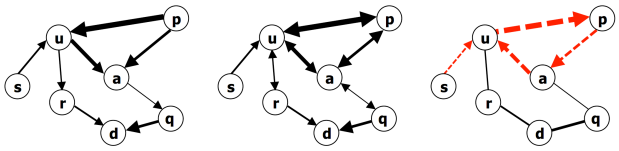
Anycast groups can be part of IP address ranges associated with Internet anycasting in the same AS, and MIDSTs can be established for them. A request for a specific anycast group whose address is in a given address range can be routed to and over the corresponding MIDST, until it reaches a network that has members in the target anycast group. The request can then be routed within that network either over a MIDST defined for a more specific address range, or directly to the nearest anchor of the target anycast group address. Anycasting across autonomous systems involves policy-based routing and is the subject of future work; however, it can also be addressed using routing to multi-instantiated destinations in the context of BGP or alternative routing protocols.

C. *Efficient Name-Based Content Routing*

ICN architectures (e.g., [12], [15], [23], [39], [45]) rely on the control plane to provide multiple routes to the same named data object (NDO). In this regard, the results in Section VI indicate that MIDR constitutes a promising approach for the design of name-based content routing protocols. MIDR is orders of magnitude more efficient than current approaches being used for name-based content routing (e.g., NLSR [31] in NDN [39]) as the number of destinations D (NDOs) and the destination instances R (replicas of NDOs) increase.

In addition, MIDR has a major positive impact in the data plane of an ICN. For example, the NDN project [31], [53] has argued that loop freedom in the control plane is not needed for Interests (content requests) to be forwarded in an ICN, because Interests that carry a content name and a nonce created by the origin of the Interest cannot loop (e.g., see [53], Section 2). We illustrate below that loop-free multipath routing in the control plane is far more desirable than routing approaches (e.g., NLSR) that do not enforce loop-free paths in an ICN operating with a data plane similar to that of NDN and such that: (a) an Interest is identified simply by the name of the content being requested and a nonce; and (b) forwarding of Interests is done on a hop-by-hop basis using that information.

The name and nonce carried in an Interest can be used to *detect* a loop with some probability *after* the Interest traverses a loop and reaches a router that has recorded the same name and nonce of the Interest in its PIT (Pending Interest Table). Interest loop-detection is not certain in NDN, because nonces are not guaranteed to be unique.



(a) MIDR (b) NLSR (c) Interest looping with NLSR

Fig. 5: Effect of multipaths in FIBs on Interest looping

Fig. 5 shows the multipath routing induced by MIDR and NLSR in a seven-node network in which NDO j is announced only by router d, and all links have unit cost. The thickness of a link indicates how congested it is, with thiner lines indicating more congestion. The arrowheads in the figure show the direction in which Interests can propagate over links. Fig. 5(a) shows the multipaths built in the FIBs (Forwarding Information Base) when MIDR is used, which are loop-free because routers coordinate the updates they make to their routing tables using SOC, and FIBs are derived from such routing tables. Fig. 5(b) illustrates the multipaths in the FIBs when NLSR is used, which are not loop-free, because routers compute multiple paths to an NDO independently of other routers. Using the information about the topology and the locations of NDOs, each router running NLSR first computes a shortest path to an NDO, deletes the adjacent link belonging to that path and computes a new path; the process continues until the router has considered all its adjacent links.

Let router s issue an Interest for NDO j. With MIDR, the Interest is bound to reach router d, independently of the Interest forwarding strategy used in the data plane. By contrast, the Interest issued by s may traverse a loop when NLSR is used, depending on the data-plane state, which Fig. 5(c) illustrates with red dashed lines. According to the Interest forwarding strategy in NDN [39], [53], for a given set of possible interfaces listed in the FIB, a router (e.g., u and a) can forward an Interest over one of those interfaces based on their perceived performance. Furthermore, router u must discard the Interest when it loops and u receives it from neighbor a.

Interest forwarding strategies can be designed to cope with Interest-looping problems resulting from inconsistent loop-prone routes due to network dynamics, or the interaction of forwarding strategies with stable FIBs that need not correspond to loop-free routes (as Fig. 5(c) illustrates). Examples include asking a router that detects a loop to reroute the Interest over alternate paths (e.g., u reroutes the Interest for j to r), sending a negative acknowledgment to the source (e.g., u tells s to try again), forwarding an Interest over all possible paths, having the Interest list the path it has traversed, or keeping state about paths traversed for each Interest. However, such approaches may not work and complicate the interaction between data and control planes. Rerouting of Interests without loop-free routing in the control plane need not be successful and takes order $O(N)$ to deliver an Interest because of backtracking due to looping [2]. On the other hand, flooding of Interests induces communication overhead larger than $O(EC)$.

Furthermore, if routing-table loops are allowed in the control plane, then Interest entries must be stored in the PITs for time durations of order $O(N)$ to ensure that no Interest that loops can be recirculated in the same loop. By contrast, with loop-free routing in the control plane, such time durations need only be of order $O(d)$.

VIII. Conclusion

Multi-Instantiated Destination Routing (MIDR) was introduced as an example of routing to destinations that can be arbitrarily multi-instantiated in a network. MIDR provides multiple loop-free paths to the nearest instances of destinations, as well as to some or all destination instances, based solely on distances to destinations. It does not require replicating information about the physical topology of the network or the location of instances of the same destination, or exchanging path information to destination instances.

MIDR was shown to be loop-free at every instant, to converge to the shortest paths to the closets instances of destinations, and to build the MIDST (Multiple Instance Destination Spanning Tree) of a destination when needed.

MIDR was shown to have smaller time and communication complexity than traditional routing approaches applied to multi-instantiated destinations, such as routing based on link states, DHTs, or traditional loop-free distance-vector routing. We argued that much better performance, functionality and simplicity can be attained for IP routing, IP multicasting, and name-based content routing by adopting multi-instantiated destination routing.

The results we have presented are just a beginning, but open up many avenues for future research on routing to multi-instantiated destinations for the Internet and ICNs. Protocols are needed to address specific applications (e.g., [19]), and different techniques can be used to establish lexicographic orderings. Lastly, the proposed approach should be extended to address policy-based routing to multi-instantiated destinations.

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] H. Ballani and P. Francis, "Towards a Global IP Anycast Service," *Proc. ACM SIGCOMM '05*, Aug. 2005.

[4] T. Ballardie, P. Francis, and J. Crowcroft, "Core Based Trees (CBT)," *Proc. ACM SIGCOMM '93*, Oct. 1993.

[5] P. Baran, "On Distributed Communications Series," *RAND Corporation*, Reports RM-3420-PR to RM-3767-PR.

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

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

[8] D. Bertsekas and R. Gallager, *Data Networks*, Second Edition, Prentice-Hall, 1992.

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

[10] C. Cheng et al., "A Loop-Free Extended Bellman-Ford Routing Algorithm without Bouncing Effect," Proc. ACM SIGCOMM '89, 1989.

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

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

[13] S. Deering, "Multicast Routing in Internetworks and Extended LANs," *Proc. ACM SIGCOMM '88*, Aug. 1988.

[14] S. Deering et al., "The PIM architecture for wide-area multicast routing," *IEEE/ACM Trans. on Networking*, Vol. 4, No. 2, April 1996.

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

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

[17] J.J. Garcia-Luna-Aceves and S. Murthy, "A Path-Finding Algorithm for Loop-Free Routing," *IEEE/ACM Trans. Networking*, Feb. 1997.

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

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

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

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

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

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

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

[25] D. Katabi and J. Wroclawski, "A Framework for Scalable Global IP-Anycast (GIA)," *Proc. ACM SIGCOMM '00*, Aug. 2000.

[26] L. Kleinrock and F. Kamoun, "Hierarchical Routing for Large Networks, Performance Evaluation and Optimization," *Computer Networks*, Jan. 1977.

[27] T. Koponen et al., "A Data Oriented (and Beyond) Network Architecture," *Proc. ACM SIGCOMM '07*, Aug. 2007.

[28] S.J. Lee, M. Gerla, and C. Chiang, "On-demand Multicast Routing Protocol in Multihop Wireless Mobile Networks," *Mobile Networks and Applications*, Vol. 7, No. 6, 2002

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

[30] E.L. Madruga and J.J. Garcia-Luna-Aceves, "Scalable Multicasting: The Core-Assisted Mesh Protocol," *Mobile Networks and Applications*, March 2001.

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

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

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

[34] J.M. McQuillan et al., "The New Routing Algorithm for the ARPANET", *IEEE Trans. Commun.*, May 1980.

[35] D. Medhi and K. Ramasamy, *Network Routing: Algorithms, Protocols and Architectures*, Morgan Kaufmann, 2007.

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

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

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

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

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

[41] C. Patridge et al., "Host Anycasting Service," RFC 1546, 1993.

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

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

[44] M.G. Reed et al., "Anonymous Connections and Onion Routing," *IEEE JSAC*, May 1998.

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

[46] M. Schwartz and T.E. Stern, "Routing Techniques Used in Computer Comm. Networks," *IEEE Trans. Comm.*, April 1980.

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

[48] I. Stoica et al., "Internet Indirection Infrastructure," *Proc. ACM SIGCOMM '02*, Aug. 2002.

[49] P. Tsuchiya, "The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks," *Proc. ACM SIGCOMM '88*, Aug. 1988.

[50] S. Weber and L. Cheng, "A Survey of Anycast in IPv6 Networks," *IEEE Comm. Magazine*, Jan. 2004.

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

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

[53] C. Yi et al., "A Case for Stateful Forwarding Plane," *Computer Communications*, 2013.

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

[55] 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 5881243, 1999.

[56] E. Zegura et al., "Application-Layer Anycasting: A Server Selection Architecture and Use in a Replicated Web Service," *IEEE/ACM Trans. Networking*, Aug. 2000.