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

J.J. Garcia-Luna-Aceves

Palo Alto Research Center
UC Santa Cruz

jj@soe.ucsc.edu
Origins of Routing for Packet Switching


- Packet headers, store-and-forward operation, statistical multiplexing of links.
- “Hot-potato heuristic routing doctrine”: Primordial distance-vector routing.
- Addresses for sender and destination of packet.
- Routing of packets is based on destination-based routing tables using node addresses.
- Destinations are routers that originate routing updates.
State of The Art in Shortest-Path Routing Algorithms

- **Problem:** Compute the path of minimum length from each router to each destination node in the network
- **G(N, E) is the network of |N| nodes and |E| links**

Each destination node has a single instance in the graph

**Distances:** Node starts by sending distance to itself

**Link-state:** Node shares the state of adjacent links
The Brave New World of Information Centric Networks (ICN)

- Many papers have stated that routing in ICNs is inherently different than routing in the “old” IP Internet.
  - Content objects are copied opportunistically in the network.
  - Multiple instances of the same destination.

- However, existing proposals for routing in ICNs assume routing algorithms designed for single-instance destinations.
Remembering Multi-Homing and John McQuillan

- McQuillan worked on the “old” and the “new” ARPANET routing protocols:

- He and others also studied multi-homed destinations:

- Same routing algorithms for single-instance destinations.

- A directory is used to map identifier of “group” to identifier of each instance.
How Can We Route to Multiple Instances of The Same Thing with Today’s Routing Algorithms?

- Source does not know all instances
- Source knows all instances
- Something in between
How Can We Route to Multiple Instances of The Same Thing with Today’s Routing Algorithms?

- **Source does not know all instances**
  - Flood the network towards instances and prune as needed

- **Source knows all instances**
  - Have each node know the topology and the location of each instance. Instances flood the network.
  - Source can compute source trees to instances

- **Something in between**
  - Designate a representative node between sources and destination instances
  - Compute a routing tree rooted at the representative
Prior Results in ICN Routing

- **No information:**
  - Intanagonwiwat et al., [Directed Diffusion (Trans. Networking 2002)]; many DTN routing schemes.

- **All information:**
  - Mahmudul-Hoque et al., [NLSR (ACM ICN ‘13)].
  - DHT approaches need an underlay to build the DHT.

- **Representative:**
  - Carzaniga, Rosenblum, and Wolf [Content-Based Networking (Infocom 04)].
Distance-based Content Routing (DCR): Basic Approach

- Establish a lexicographic ordering of distances to instances of destination.
- The name of a router “speaking for” a destination instance (called anchor) is an attribute that must be used in the ordering.
- Routers choose what information to share with their peers to preserve ordering (e.g., “the best distance to any instance of content”).
- Lexicographic ordering among all instances determines which instance can be the root of a DAG spanning all instances.
DCR

Routing to nearest instances of destination:

- \( N^i \) = set of neighbors of node \( i \)
- \( S^i \) = set of neighbors of node \( i \) that are closer to destination (next hops).
- \( A^i \) = set of known anchors of destination at node \( i \).

An update from \( i \) about a destination states \([d^i, a^i, sn^i]\):
  - distance \( (d^i) \), an anchor \( (a^i) \), a seq. number created by the anchor \( (sn^i) \).

- \( d^i = \text{Min}\{d^i_k + l^i_k | k \in S^i\} \)
  - \( d^i_k \) = distance reported by \( k \); \( l^i_k \) = cost of link to \( k \)
DCR

Routing to nearest instances of destination:

- Successor-Set Ordering Condition (SOC) used by node $i$ to select a next hop $k$ for destination:

  ✓ Node $k$ reports up-to-date information
  
  \[
  ( \forall m \in A^i \ ( a^i_k \neq m \lor s^i_k \geq s^i(m) ) )
  \]

  ✓ If node $i$ has a finite distance:
    
    Node $k$ is closer ($d^i_k < d^i$) or is at the same distance and has a smaller name ($d^i_k = d^i \land |k| < |i|$)

  ✓ If node $i$ does not have a finite distance:
    
    Node $k$ offers the smallest distance or has smallest name among neighbors offering the same smallest distance
    
    \[
    ( \forall q \in N^i - \{k\} \ ( d^i_k < d^i_q \lor ( d^i_k = d^i_q \land |k| < |i| ) ) )
    \]
• Three instances (at d, o, and u)
• Lexicographic ordering based on: hop count to instance, ID of instance’s anchor, and sequence number from anchor.
• Route to nearest instance w/o knowing all.
DCR
Routing to some or all instances of destination:

1. Select the anchor with the smallest identifier as the root anchor (ra) of the destination.
2. Select neighbor \( k \) as next hop to root anchor if node \( k \):
   - Reports up-to-date anchor information
   - Has the smallest sequence number for \( ra \) or is closest to \( ra \) among all neighbors with a valid sequence number for \( ra \).
3. Ensure that root anchor information is known by all anchors of destination:
   - Node \( i \) sends update about \( ra \) to neighbor \( q \) if \( i \) is an anchor or \( q \) is the lexicographically smallest way to reach another anchor:
     \[
     a^i_q \neq ra^i_j \land |ra^i_q| > |ra^i| \land \\
     \forall k \in N^i - \{i\} \ [ a^i_k = i \lor \\
     (a^i_k \neq ra^i_q \lor (d^i_q < d^i_k \lor [d^i_q = d^i_k \land |q| < |k|)]))]
     \]
• One instance is “best” and hence is known by other instances ($d < o < u$).
• Updates about $d$ are sent only along the best paths between $d$ and $o$ and between $d$ and $u$. 
The MIDST of a Destination: Multiple Instance Destination Spanning Tree

- Only Anchors send joins towards root anchor to form MIDST.
- Messages sent towards nearest instance, broadcast over MIDST to reach all or some instances.
Why Is This Important?

Attain orders of magnitude improvement in the control plane

\[ CC_{LSR} = O(ERC + lNE) \]
\[ SC_{LSR} = O(RC + E) \]
\[ CC_{DHT} = O(dRC + lNE) \]
\[ CC_{DHT} = O(C + E) \]
\[ CC_{DCR} = O(EC); \]
\[ SC_{DCR} = O(C) \]

- \( C \): # destinations
- \( R \): # replicas
- \( E \): # edges; \( N \): # nodes
- \( d \): diameter; \( l \): node degree

\( N = 500 \)
\( l \) and \( d \) are \( O(\log N) \)
**Why Is This Important?**

**Loop-Free FIBs**  
**Loop-Prone FIBs**  
**Interest looping**

**More efficient data plane!**

Looping with multi-path routing leads to searching for paths in $O(N)$ steps w/o guarantees (DFS) and Interest timers of order $O(N)$ to avoid loops not being detected!

Loop-free FIBs allow searching for paths in $O(x)$ steps and Interest timers of order $O(d)$, with $x < d << N$
Summary

- **Just a start!** Many more results are needed:
  - DCR simulations in ns3 soon
  - Real implementation after that
  - Multipoint applications (e.g., multicast)
  - QoS, policies, other types of termination detection.

- Change the way we think about destinations for *all* types of routing; assume any destination can have multiple instances.

- Integrate unicast, multicast, and anycast routing.

- Need solutions that scale *better* than $O(C)$. *It can be done using distances.*
THANK YOU!

ANY QUESTIONS?