NDN Distance Vector (ndn-dv)

Varun Patil

Lessons from past routing protocol designs

- Internet: customers attached to the infrastructure
  - Customers announce prefixes to the network
- Routing: establish reachability to customer prefixes
  - Too many customer prefixes □ scaling challenges
  - Does the 2-level routing (router and prefix reachability) help with prefix scalability?

https://github.com/named-data/ndnd

Two types of reachability

- Reachability to routers
  - Scales with number of routers in the network
- Reachability to end-user prefixes
  - (In NDN) Scales with number of applications
- Existing routing protocols do not explicitly or effectively separate the two
  - OSPF: 2 separate LSA types, but both sent together
  - BGP: propagates prefix reachability only

Existing Understanding

- In practice, intra-AS routing separates router and prefix reachability
  - IGP: builds the intra-AS router reachability table.
  - iBGP: builds the mapping from prefixes to next-hop routers.
- Packet forwarding performs two lookups
  - Use the BGP table to determine the exit router for a destination prefix.
  - Use the IGP table to determine how to reach that exit router.
- Locator-Identifier Separation Protocol (RFC 6830)
  - Maps customer prefixes to their network attachment points
  - Endpoint Identifiers (EIDs) and Routing Locators (RLOCs)

Separating reachability concerns in NDN

- ndn-dv routing – establish FIB for reachability to routers
- P2Rsync, a separate prefix-to-router mapping protocol
  - #name prefixes can be orders of magnitude bigger than #routers
  - A name prefix can be reachable through multiple routers
  - A prefix may/not change its attachment point(s) frequently
- Two step Interest forwarding
  - Lookup exit router for the prefix using mapping table
  - Lookup next hop for that router

Why Distance Vector?

- Does not need topological map
  - Every router exchanges distance vector only with neighbors
  - No flooding updates
- Low overhead
  - Fewer updates – changes are only propagated as far as needed
  - Simpler computation

Counting to Infinity

- RIP uses “poison reverse” as mitigation
  - But can still happen in corner cases
- With adequate topological redundancy: count to the next path
- Loops may still exist
  - They are transient
  - Some packets may loop in the data plane
- NDN forwarding breaks these loops
  - PIT / DNL detect looped packets
  - Forwarding strategy can work around them

ndn-dv components

- RIB – distance to each router through each interface
  - Router reachability
- Advertisement – **same** message broadcast to all neighbors
  - Distance Vector
  - Extra information for multi-path reachability
- Prefix Table – mapping routers to prefixes
  - Prefix reachability
- FIB computation – for legacy compatibility
- Security – updates secured like any other NDN application
  - Each router undergoes security bootstrapping
  - LightVerSec trust schema

Destination	Intf (1)	Intf (2)	Intf (3)
/router1	1	3	∞
/router2	2	2	∞
/router4	3	1	∞
/router5	4	2	∞
/router6	∞	∞	1
/router7	4	2	∞

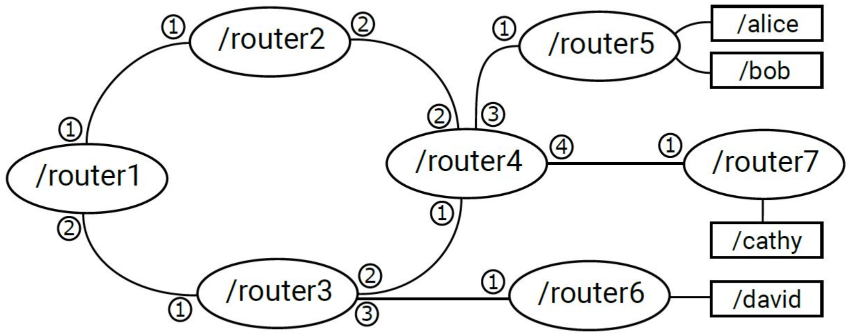
TABLE I: RIB at router 3 in our example.

Destination	Next Hop	Cost	Other
/router1	/router1	1	3
/router2	/router1	2	2
/router3	/router3	0	∞
/router4	/router4	1	3
/router5	/router4	2	4
/router6	/router6	1	∞
/router7	/router4	2	4

TABLE II: Advertisement generated by router 3.

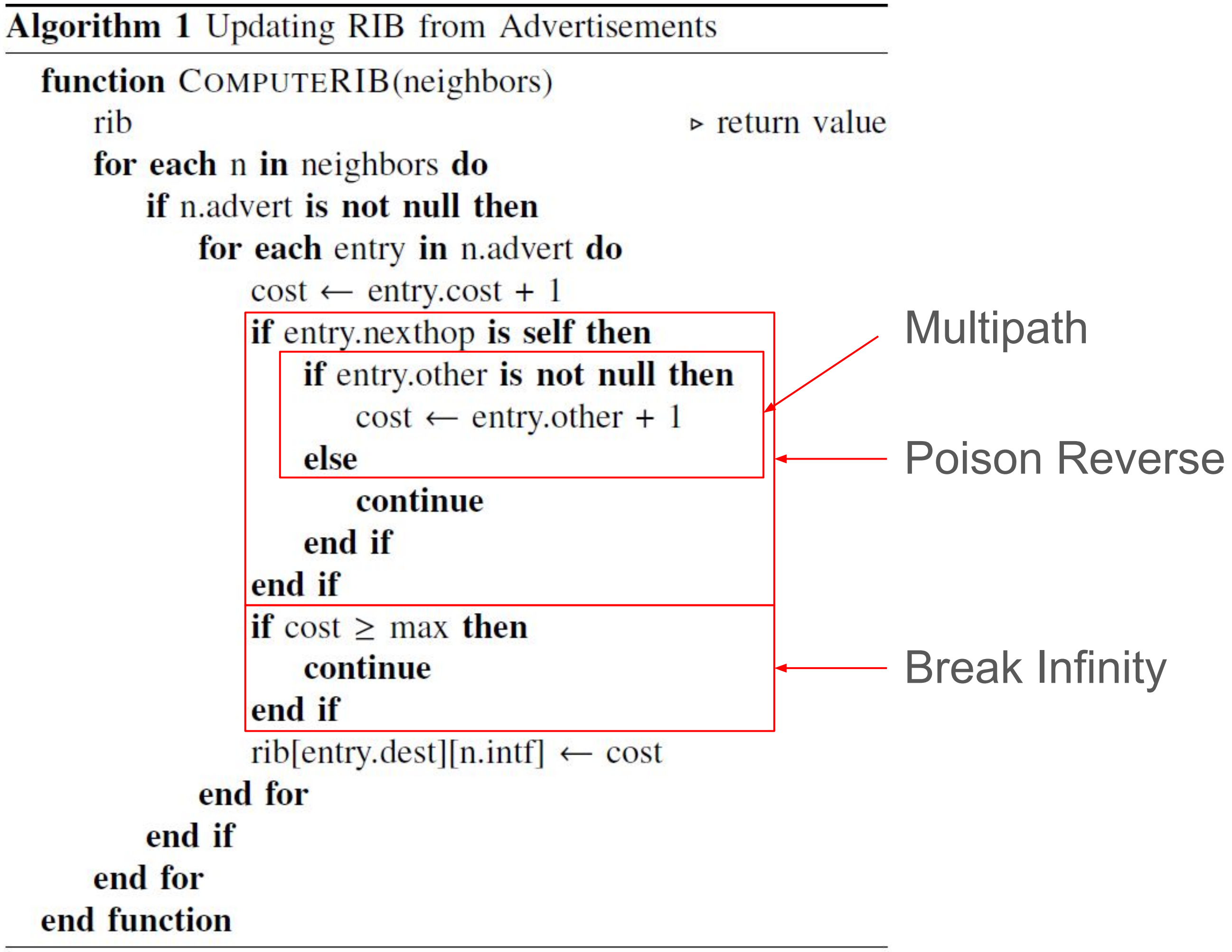
Name Prefix	Exit Router
/alice	/router5
/bob	/router5
/cathy	/router7
/david	/router6

TABLE III: Global prefix table in our example.



Name Prefix	Next Hops
/alice	intf=2 (cost=2), intf=4 (cost=4)
/bob	intf=2 (cost=2), intf=4 (cost=4)
/cathy	intf=2 (cost=2), intf=4 (cost=4)
/david	intf=3 (cost=1)

TABLE IV: FIB at router 3 in our example.

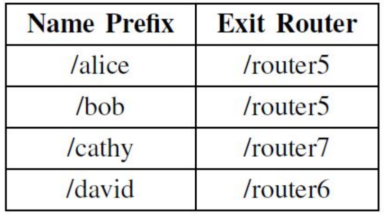

Multipath
Poison Reverse
Break Infinity

Algorithm 1 Updating RIB from Advertisements

```
function COMPUTERIB(neighbors)
    rib                                          ▷ return value
    for each n in neighbors do
        if n.advert is not null then
            for each entry in n.advert do
                cost ← entry.cost + 1
                if entry.nexthop is self then
                    if entry.other is not null then
                        cost ← entry.other + 1
                    else
                        continue
                    end if
                end if
                if cost ≥ max then
                    continue
                end if
                rib[entry.dest][n.intf] ← cost
            end for
        end if
    end for
end function
```

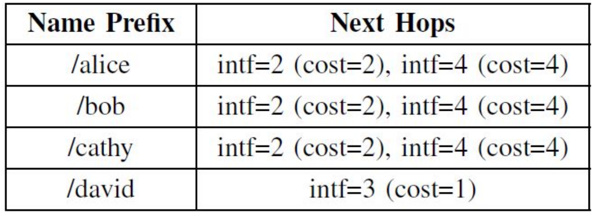
Prefix Table

- Synchronized with SVS-PS globally
- Not affected by topology change
- Ideally – two step forwarding



Name Prefix	Exit Router
/alice	/router5
/bob	/router5
/cathy	/router7
/david	/router6

TABLE III: Global prefix table in our example.



Name Prefix	Next Hops
/alice	intf=2 (cost=2), intf=4 (cost=4)
/bob	intf=2 (cost=2), intf=4 (cost=4)
/cathy	intf=2 (cost=2), intf=4 (cost=4)
/david	intf=3 (cost=1)

TABLE IV: FIB at router 3 in our example.

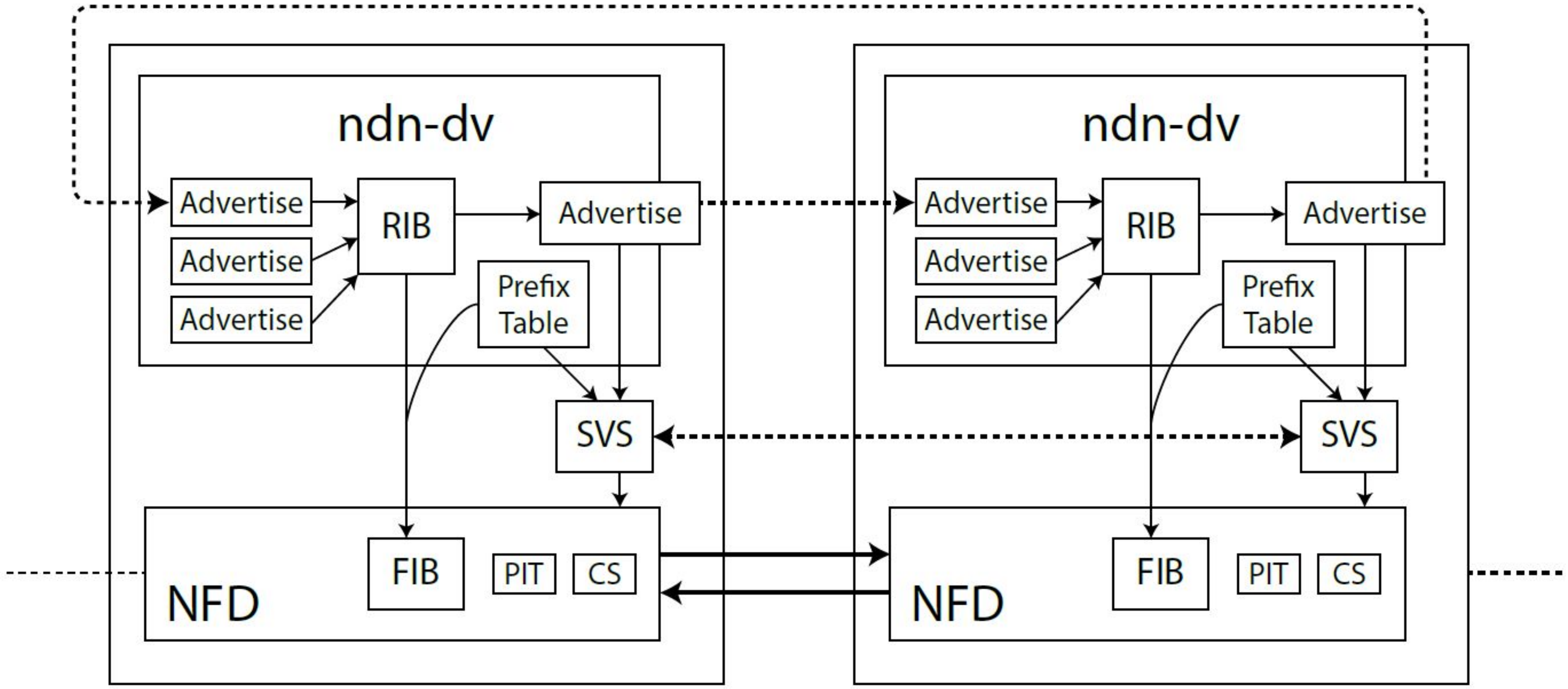


Fig. 1: Design overview of ndn-dv.

Preliminary Evaluation

- 52-node topology, 50ms delay on each link
  - Emulated in MiniNDN
  - https://github.com/pulsejet/eval-ndn-dv
- 80 randomly setup flows, 100 data interest per second
  - Emulate application behavior
- Mean-Time-To-Failure (MTTF) = 4000s → 300s
- Mean-Time-To-Recovery (MTTR) = 120s
- Measure fraction of satisfied Interests

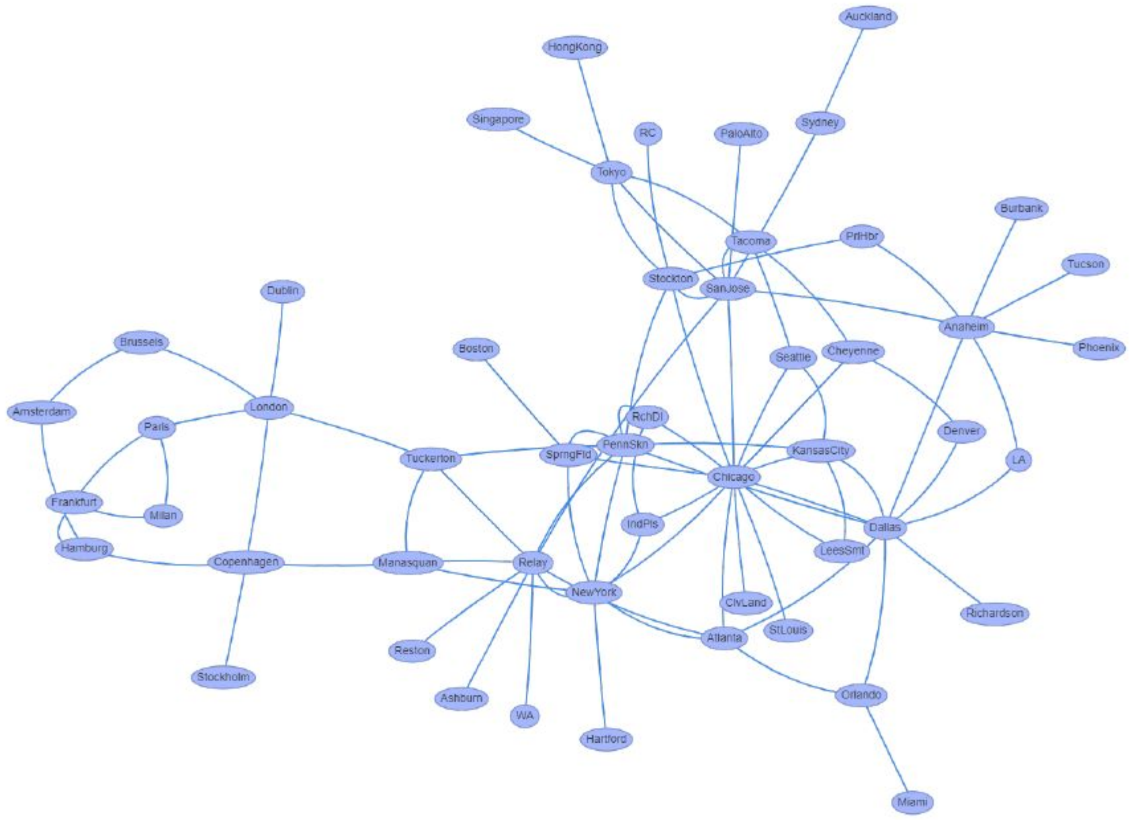
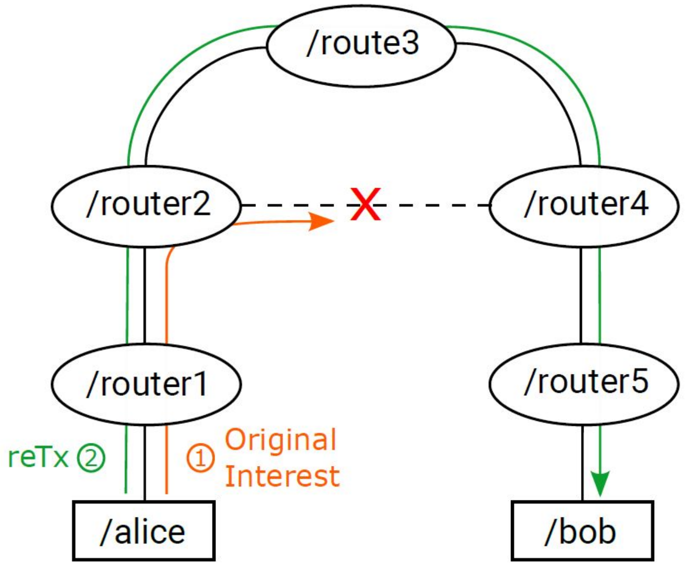


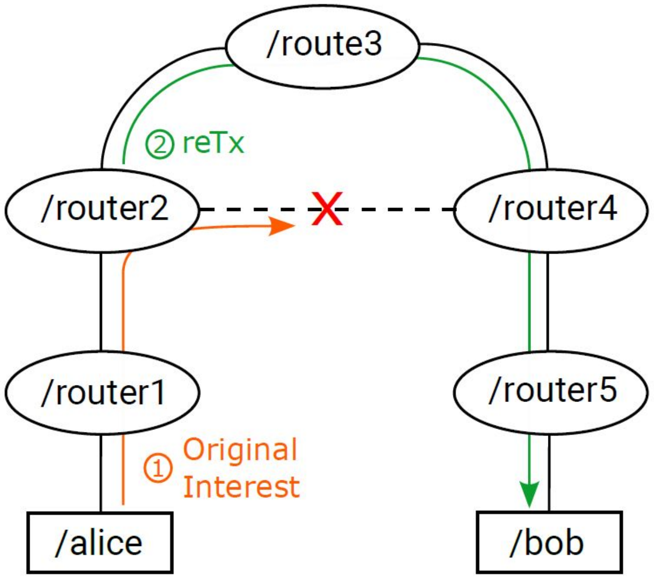
Fig. 3: Sprint PoP topology used for evaluations.

Evaluation w/ Retransmissions

- Baseline – no retransmission



Best route w/ retransmissions



Best two routes strategy (experimental)

Evaluation

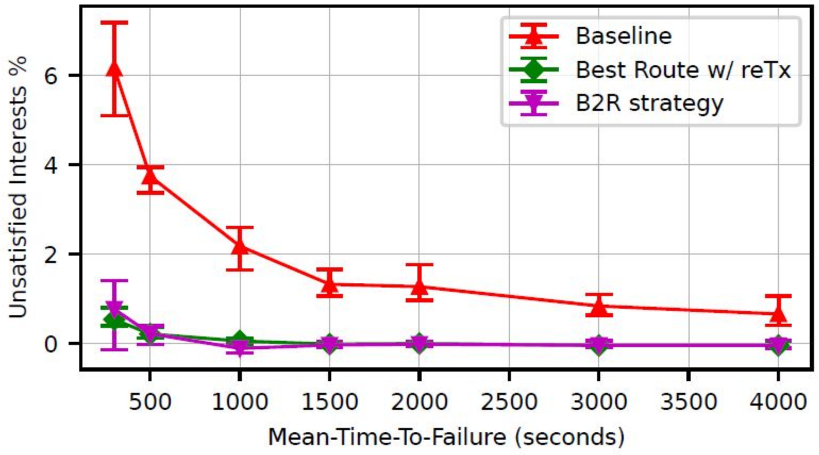


Fig. 7: Fraction of unsatisfied Interests with ndn-dv.

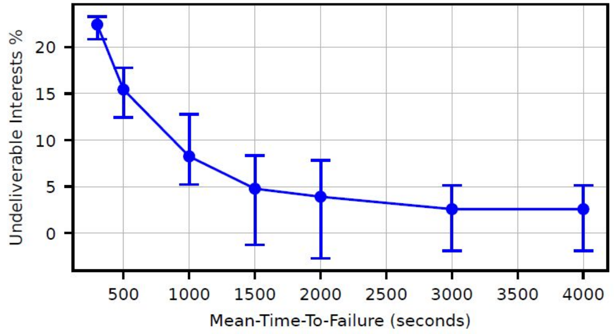


Fig. 6: Unsatisfiable Interests due to network partitions.

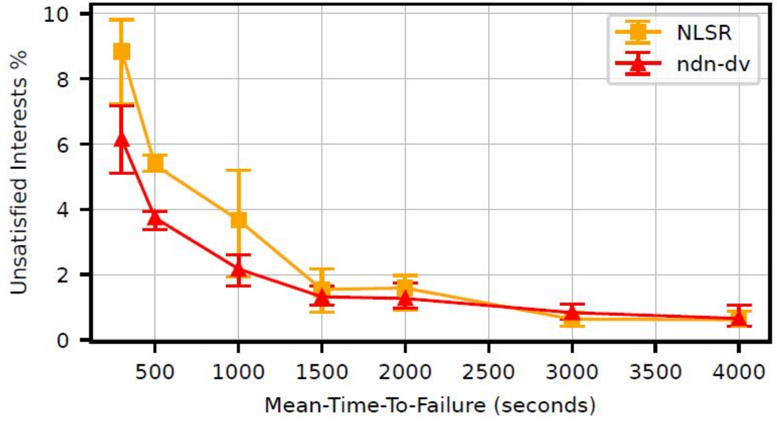


Fig. 8: Baseline comparison of ndn-dv with link-state.

Final Notes

- NDN forwarding breaks loops in the data plane
- NDN can effectively use a simple DV routing protocol
- Scaling by separating router and prefix reachability
  - Prefix to router mapping table, synchronized with NDN Sync
  - **Also usable with any other NDN routing protocols**

- Implementation and specification available in NDNd
  - https://github.com/named-data/ndnd/tree/main/dv
  - https://github.com/named-data/ndnd/blob/main/docs/daemon-example.md
  - https://github.com/named-data/ndnd/blob/main/dv/SPEC.md