Adaptive Forwarding in Named Data Networking

Cheng Yi
University of Arizona
yic@cs.arizona.edu

Alexander Afanasyev
UCLA
afanasev@cs.ucla.edu

Lan Wang
University of Memphis
lanwang@memphis.edu

Beichuan Zhang
University of Arizona
bzhang@arizona.edu

Lixia Zhang
UCLA
lixia@cs.ucla.edu

This article* is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT

In Named Data Networking (NDN) architecture, packets carry data names rather than source or destination addresses. This change of paradigm leads to a new data plane: data consumers send out *Interest* packets, routers forward them and maintain the state of pending Interests, which is used to guide *Data* packets back to the consumers. NDN routers' *forwarding* process is able to detect network problems by observing the two-way traffic of Interest and Data packets, and explore multiple alternative paths without loops. This is in sharp contrast to today's IP forwarding process which follows a single path chosen by the routing process, with no adaptability of its own. In this paper we outline the design of NDN's adaptive forwarding, articulate its potential benefits, and identify open research issues.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Packet-switching networks

Keywords

NDN, data plane, adaptive forwarding

1. INTRODUCTION

A network's architecture design determines the shape and form of its forwarding mechanism. Today's IP Internet accomplishes packet delivery in two phases. At the routing plane, routers exchange routing updates and select the best routes to make up the forwarding table (FIB). At the data plane, routers forward packets strictly following the FIB. Thus, routing is stateful and adaptive, while forwarding is stateless and has no adaptability. This *smart routing, dumb forwarding* design places the responsibility of robust packet delivery solely on the routing system.

As a newly proposed Internet architecture, Named Data Networking (NDN) inherits the hourglass shape of the IP architecture, but replaces the host-to-host data delivery model at the hourglass' thin waist by a data retrieval model [2, 5]. NDN packets carry data names rather than source or destination addresses. Data consumers express Interests in the form of desired data names, without specifying where the data may be located. Routers satisfy the Interests by retrieving the data, which are bound to the names with cryptographic signatures, from their own caches, intermediate repositories, or the data producers. While routing in an NDN network serves the same purpose as in an IP network, i.e., computing routing tables to be used in forwarding Interest packets, the data plane in an NDN network is split to a two-step process: consumers first send out *Interest* packets, then *Data* packets flow back along the same path in the reverse direction. Routers keep the state of pending Interests in order to guide Data packets back to consumers.

Obvious benefits of NDN's data plane include built-in network caching and multicast support. A less obvious but equally important benefit is its *adaptive forwarding* enabled by the state maintained at routers. By recording pending Interests and observing Data packets coming back, individual NDN routers can measure packet delivery performance (e.g., round-trip time and throughput), detect packet losses, and utilize multiple alternative paths to bypass problematic areas. With such an intelligent and adaptive data plane, the routing plane in an NDN network only needs to disseminate long-term changes in topology and policy, without having to deal with short-term churns.

The seminal paper by Jacobson et al. [2] sketched out the blueprint of the overall NDN architecture, however the operations of its data plane are not fully explained and the design specifics remain to be filled in. Our main goal in this paper is to describe how NDN's adaptive forwarding works and identify its main advantages as well as the design tradeoffs. We first outline the design of an adaptive forwarding mechanism for NDN, illustrate its benefits using a few case studies, then identify key open research issues. We have carried out a preliminary evaluation of the NDN data plane performance through simulation and the results are reported in a longer version of the paper [9].

2. OVERVIEW OF NDN'S DATA PLANE

In this section we briefly introduce NDN with a focus on its stateful data plane. NDN is a receiver-driven, data-

*The material in this article is based upon the work supported by the National Science Foundation under Grants No. 1039615, 1040036, and 1040868; and performed in a renovated collaboratory by the National Science Foundation under Grant No. 0963183, which is an award funded under the American Recovery and Reinvestment Act of 2009 (ARRA). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

centric communication protocol. All communication in NDN is performed using two distinct types of packets: *Interests* and *Data*, both of which carry a *name*, which uniquely identifies a piece of data. A consumer puts the name into an Interest packet and sends it to the network. Routers use this name to forward the Interest towards the data producer, and the Data whose name provides the best match to the Interest is returned to the consumer. All data packets carry a signature that securely binds the name to the data.

Similar to IP packet delivery, an NDN network performs best effort data retrieval. An Interest or Data packet can be lost, and it is the end consumer's responsibility to retransmit the Interest if it does not receive the desired data after the expected round trip time.However, unlike IP's location-centric approach to data delivery, NDN packets carry data names instead of source or destination addresses. This basic difference in design leads to two profound differences in operations. First, although the name in an Interest packet is used to guide its forwarding, similar to how a destination address guides the forwarding of an IP packet, the Interest may cross a copy of the requested Data at an intermediate router or data repository and bring the Data back, while an IP packet always reaches the destination (if not dropped). Second, NDN consumers have neither addresses nor names to be used for Data packet delivery. Instead, NDN routers keep track of incoming interfaces for each forwarded Interest (a pending Interest) and use this information to bring matched Data packets back to consumers.

2.1 Forwarding Process

Each NDN router maintains three data structures: a *Content Store* for temporary caching of received Data, a *Pending Interest Table (PIT)*, and a *forwarding table (FIB)* (see Fig. 1). By its name, each PIT entry records an Interest packet that has been forwarded, waiting for the Data packet to return. The entry records the name, the incoming interface(s) of the Interest, and the outgoing interface(s) to which the Interest has been forwarded. An NDN router's FIB is similar to the FIB in an IP router except that it contains name prefixes instead of IP address prefixes, and it may show multiple interfaces for a given name prefix (see Section 3.3). In addition, each NDN router has a *strategy module* that makes forwarding decisions for each Interest packet (see Section 3.4).

When a router receives an Interest packet, it first checks whether there is a matching Data in its Content Store. If a match is found, the Data is sent back to the incoming interface of the Interest packet. If not, the Interest name is checked against the entries in the PIT. If the name exists in the PIT already, it means an Interest from another consumer for the same name has been received and forwarded earlier, and the router simply adds the incoming interface of this newly received Interest to the existing PIT entry. If the name does not exist in the PIT, the Interest is added into the PIT and further forwarded.

In addition to the data name, each Interest packet also carries a random nonce generated by the consumer. A router remembers both the name and nonce of each received Interest, so it can tell whether a newly arrived Interest is indeed a new one or an old one that looped back (and drops it). Thus, Interest packets cannot loop. Because Data packets follow the reverse path of the corresponding Interest packets, they do not loop either.

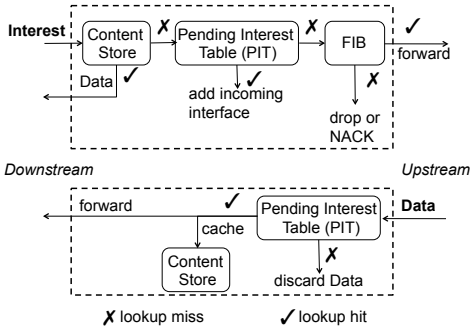

Figure 1: Interest and Data processing in NDN

When a Data packet is received, its name is used to look up the PIT. If a matching PIT entry is found, the router sends the Data packet to the interfaces from which the Interest was received, caches the data, and removes the PIT entry. Otherwise, the Data packet is unsolicited and discarded. Each Interest also has an associated lifetime; the PIT entry is removed when the lifetime expires.

2.2 Datagram State

An NDN router maintains an entry in its PIT for *every* pending Interest packet, thus we say the router contains "datagram state." This state leads to a closed-loop two-way symmetric packet flow: over each link, every Interest packet pulls back exactly one Data packet, maintaining one-on-one flow balance, except in (rare) cases where packets get lost or matching data does not exist.

We note that NDN's datagram state differs in fundamental ways from the virtual circuit state for ATM or MPLS. First, a virtual circuit sets up a single path between an ingress-egress router pair; when it breaks, the state has to be recovered for the entire path. Second, a virtual circuit pins down the path to be used for packet forwarding; if some link along the path gets overloaded due to traffic dynamics, packets on the same virtual circuit cannot be diverted to adapt to the load changes. In contrast, NDN's datagram state is per-Interest, per-hop. At each hop, the router makes its own decision on where to forward an Interest. When a router crashes, only the Interest state in that router is lost; the previous hop routers can quickly detect the problem and forward future Interests around the failure.

3. ADAPTIVE FORWARDING

In this section we describe an initial design that utilizes NDN routers' datagram state to build an intelligent and adaptive data plane. The main goal is to retrieve data via the best performing path(s), detect any packet delivery problems quickly and recover from them.

3.1 Interest NACK

In the original sketch of NDN [2], after a router forwards an Interest, it starts a timer based on estimated RTT. When the expected Data packet comes back before the timer expires, RTT is updated; otherwise there may be potential problems on the path. However, the timer-based problem detection can take time. Worse yet, when the data cannot be found along certain path, the unsatisfied Interest (which we call *dangling state*) remains in the network until its lifetime expires, potentially blocking other consumers' Interests for the same data, since the PIT entry already exists and the routers simply wait for the Data to return.

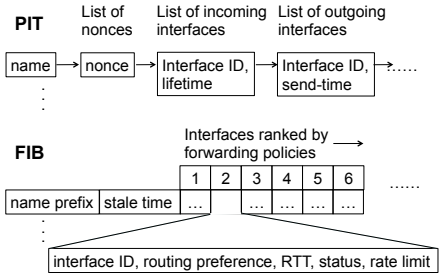

Figure 2: Forwarding State in PIT and FIB

In this paper we introduce *Interest NACK* to address these issues. When an NDN node can neither satisfy nor forward an Interest (e.g., there is no interface available for the requested name), it sends an Interest NACK back to the downstream node. A NACK carries the same name as the corresponding Interest packet, plus an error code explaining why the NACK is generated (e.g., *Congestion*, *No Path*, etc.). If the downstream node has exhausted all its own forwarding options, it will send a NACK further downstream.

In the absence of packet losses, every pending Interest is consumed by either a returned Data packet or a NACK. A NACK notifies the downstream node of network problems quickly, which can then take proper actions based on the error code, and delete the unsatisfiable Interest from PIT. Note that Interest NACKs are different from ICMP messages: a NACK is sent to the previous hop while an ICMP message is sent to the source host. Thus, their effects are entirely different.

3.2 PIT

PIT maintains datagram forwarding state (Figure 2). A PIT entry is created for each requested name. It contains a list of incoming interfaces from which the Interests for that name have been received, a list of nonces that have been seen for that name, as well as a list of outgoing interfaces to which the Interest has been forwarded. Within a PIT entry, each incoming interface records the longest Interest lifetime it has received; when the lifetime expires the incoming interface is removed from the PIT entry. Each outgoing interface records the time when the Interest is forwarded via this interface, so that when Data packet returns the RTT can be computed.

3.3 FIB

NDN FIB differs from IP FIB in two fundamental ways. First, an IP FIB entry usually contains a single best next-hop, while an NDN FIB entry contains *a ranked list of multiple interfaces*. Second, an IP FIB entry contains nothing but the next-hop information, while an NDN FIB entry records information from both routing and data planes, providing input to adaptive forwarding decisions (see Figure 2).

3.3.1 Routing Information

FIB entries record all the name prefixes announced in routing. When a name prefix is withdrawn by routing, it is not immediately removed, but kept for a *stale time* period (or longer if Interests for the corresponding prefix continue to be satisfied). This minimizes transient unreachability during routing convergence, when some reachable prefixes are temporarily withdrawn.

Each FIB entry lists all policy-compliant interfaces together with their routing preference for reaching the given name prefix. That is, an interface is included, unless it is forbidden to serve the prefix by routing policy. Routing preference is the outcome of applying routing policy and metrics to paths computed by routing. It is one of the inputs that we use to rank the interfaces.

3.3.2 Forwarding Performance Information

A FIB entry also records for each interface the data retrieval status. Exactly what is the best way to represent this status is an open research question; we are currently experimenting with a simple coloring scheme:

- Green: the interface is working.
- Yellow: the interface may or may not work.
- Red: the interface does not work.

When an interface comes up online or a new FIB entry is created, the interface status is Yellow. It turns Green if Data is currently flowing back from that interface. A Green interface turns Yellow when a pending Interest times out, or after it is unused for a while. An interface is marked Red if it has failed, or the router has received a "No Path" NACK from the upstream.

A FIB entry also maintains an RTT estimate for data retrieval via each interface. It is a moving average of RTT samples taken every time a Data packet is received. This RTT estimate provides an input to interface ranking; it is also used to set up an *exploration timer* based on the expected Data packet return time, as we explain in Section 3.4.

3.3.3 Interface Ranking

All the Interfaces in a FIB entry are ranked in order to help forwarding strategy choose which interface(s) to use. For each prefix, the ranking of its interfaces is based on routing preference, observed forwarding performance, as well as the *forwarding policy* set by operators. A wide variety of forwarding policies can be supported in an NDN network. For example, if the policy is "the sooner the better," then interfaces with smaller RTTs will be ranked higher; if the policy is performance stability, then the current working path is ranked higher. Yet another example is a higher preference for a particular neighbor, which leads to forwarding a higher percentage of Interests to that interface than other equally available ones. Note the differences between routing policy and forwarding policy: the former determines which routes are available to the data plane, while the latter determines which routes may be used and in which order.

3.3.4 Rate Limit

The one-to-one flow balance between Interests and Data offers NDN networks effective means of congestion control. By pacing Interests sent to the upstream direction of a link (towards producer), one can prevent congestion caused by Data packets in the downstream direction of the link.

We experimented with a simple calculation to set the Interest rate limit over an interface: $L_i = \alpha \times C_i/\bar{S}_i$, where L_i is the Interest rate limit of interface i, C_i is the upstream link capacity of i, $\bar{S}_i$ is an estimate of the size of the Data packets that have been received over i, and α is a configurable parameter. The ratio $C_i/\bar{S}_i$ is the maximum data rate that is allowed from upstream measured in packets per second (pps), which should be the same as the maximum Interest rate going upstream. The coefficient α is used to compensate for errors in the calculations (e.g., imprecise data size estimate, link and network layer overheads).

When L_i is reached, a node cannot forward more Interests out to Interface i. If the node does not have other choices to forward an Interest, it sends a NACK with error code of congestion back to the downstream. The downstream node then explores alternative paths to forward the Interest.

Each NDN node also maintains another rate limit, $L_{i,n}$, for interface i and name prefix n, and stores it in the corresponding FIB entry as shown in Figure 2. When a congestion NACK is received from interface i and for name under prefix n, $L_{i,n}$ is reduced; when a matching Data packet is received, $L_{i,n}$ is increased. The specific adjustment algorithm is an area of our current research. One option is to use algorithms similar to TCP's slow start and AIMD.

When neither L_i nor $L_{i,n}$ is reached, interface i is said to be *available* for forwarding to name prefix n, otherwise *unavailable*. Interests are only forwarded to available interfaces.

3.4 Forwarding Strategy

Given the information stored in PIT and FIB, the strategy module determines which interface to use to forward an Interest, making forwarding decisions adaptive to network conditions. Our initial design includes the handling of new and retransmitted Interests, Interest NACKs, and performing proactive interface probing.

New Interest: When a router does not find a match in its PIT for a newly arrived Interest, it creates a new PIT entry and forwards the Interest using the highest ranked available Green interface for the name prefix. If no such interface exists, the highest ranked available Yellow interface is used.

Retransmitted Interest: If an Interest matches an existing PIT entry but its nonce does not exist in the nonce list, this Interest is regarded as a retransmission. When a router receives a retransmitted Interest before the exploration timer expires, the Interest will not be forwarded. Otherwise, the router will try a next best Green or Yellow available interface to forward the retransmitted Interest.

Interest NACK: When a router receives an Interest NACK, it will send the corresponding Interest to the next highest ranked available interface. Ideally, the router should try a few alternative paths but not for too long (the application may have moved on without the missing Data). Thus, the router uses the exploration timer to limit how long this "path exploration" should take. The timer starts when a new or retransmitted Interest is forwarded for the first time, with a timeout value set to the expected RTT (plus variance). A router explores alternative interfaces whenever a NACK is received until it succeeds or until the timer expires. After the timer expires, router will stop trying alternative interfaces unless it receives a retransmitted Interest (by consumer host).

Interface Probing: Although Interest packets should be forwarded to Green interfaces when they are available, it is also important to periodically probe Yellow interfaces in order to discover other working paths or paths with better performance, e.g., a good path becomes available after failure recovery or a path to a cache that is closer than the producer. A router proactively probes Yellow interfaces by forwarding a copy of an Interest to it. Probing provides availability and performance information for alternative paths, but also retrieves duplicate Data. One can control this overhead by limiting the probing frequency.

In all the above situations, if a router has no available interfaces to forward an Interest, it will send a NACK with a proper error code back to the downstream node. Though routers try their best to explore alternative paths to get around network problems, consumers are ultimately responsible for retransmitting Interests if they still want the data.

4. CASE STUDIES

In this section, we use three problem scenarios, link failure, congestion, and prefix hijack, to demonstrate the advantages of NDN's stateful data plane. Our simulation evaluation has also confirmed that NDN provides superior packet delivery performance than IP in each of these scenarios [9].

Link Failure.

If link layers can detect failures quickly and inform the network layer, failure detection time can be very short in both NDN and IP. In other cases where lower-level detection is unavailable or inadequate, IP relies on routing protocols' periodic keep-alive messages to detect failures, which usually takes seconds or even tens of seconds, while NDN relies on observing two-way packet flows and can detect failures typically in the order of tens or at most hundreds of milliseconds.

In our proposed NDN FIB design, if an interface failure is detected or Interests sent to the interface do not bring back Data, a router labels the interface Red or Yellow respectively and explore alternative paths; in case no alternative path is available, the router returns Interest NACKs, which trigger downstream routers to explore other paths. If a consumer does not receive data within the estimated RTT, it may re-express the Interest, triggering consumer-side routers to search for a working path. As soon as a working path is found, i.e., a path that can bring back valid Data packets, it is assigned a Green status and used to forward future Interests for the same name prefix until another failure occurs, or a better path is found.

In an IP network a failure detection will trigger routing announcements being sent out and the routing system goes through a convergence process, during which inconsistent paths among routers may lead to packet loops or dead ends. As long as IP routers forward packets strictly following the routing table, the network may suffer from routing convergence delays and even significant packet losses during this period. The problem is especially prominent when the routing convergence takes long time, which can be seconds in regular OSPF and tens of seconds or even minutes in BGP.

Although an NDN network also uses routing protocols to propagate prefix reachability information throughout the network, its data plane does not solely rely on routing to forward packets. Instead, NDN routers use both routing information and feedback from the data plane to guide forwarding decision. Therefore, NDN routers can quickly adapt failures and provide uninterrupted data delivery.

Congestion.

When an NDN router detects that a link has reached its load limit, it will automatically try other available links to forward the Interests. If all the available links are congested, the router will return NACKs to downstream routers, which will try their alternative paths. Consequently, traffic in NDN can automatically split among multiple parallel paths as needed, leading to better network utilization and better application performance.

This feature of NDN is in sharp contrast to today's IP routing, which in general does not take congestion into consideration due to concerns of route oscillation and frequent routing updates. When traffic flows experience congestion, the only option is to slow down the sending rate. If keep-alive messages between routers are lost due to congestion, IP routers will consider the link down and start routing convergence process, switching all traffic away from the overloaded link. Thus, IP routers either do not detect the congestion or misdiagnose the problem.

Another benefit of NDN's way to deal with congestion is accurate knowledge and control of available resources. When excess Interests trigger NACK returns from upstream routers, a router can dynamically adjust its rate limit based on the percentage of Interests returned. Therefore, a downstream router can match its sending rate to what upstream router can support. If the network reaches its capacity, the Interest NACKs will eventually be returned all the way back to the consumer and cause the application or transport layer to adjust the sending rate. The adjustment of Interest sending rate is done before excessive Data packets being pulled into the network, a more effective control than reacting to congestion after it occurs. Moreover, when a router needs to return Interest NACKs due to congestion, it can return Interests selectively to achieve certain policy goals. For example, to enforce fairness rules, it can return Interest NACKs to downstream routers in proportion with the number of Interests received from each.

Overall, NDN enables hop-by-hop adaptive congestion control mechanism, which reacts to congestion quickly and utilizes multiple paths as needed, and is able to accommodate administrative policies.

Prefix Hijack.

Assuming a prefix hijack attack leads to packets falling into black holes, NDN can easily detect such attacks because they disrupt the normal two-way packet flows. The attack, when a hijacker announces a victim's name prefix and drops all Interests going to the name prefix, can be mitigated in the following ways. First, some routers may see that a previously unused interface (Yellow) gets ranked higher by the routing protocols than the current face (Green) in use. The routers will continue to forward Interests through the current working interface (which leads to the legitimate prefix origin), and will probe the higher ranked Yellow interface by sending copies of the Interests to it at the same time. Since this new interface leads to the hijacker and does not return Data packets back, it will remain Yellow and unused. Second, for the routers whose current path goes through the hijacker's router even before the hijack happens, they will notice that Data packets stop coming back. This will result in the current interfaces being labeled Yellow and routers switch to better paths if they exist; the consumer will also eventually start retransmitting failed Interests to trigger the exploration of alternative paths.

The above built-in and effective mitigations against black holing attacks are again in contrast to the difficulties in dealing with the same problem in an IP network. When the routing table is polluted, either accidentally or intentionally, the routing system cannot detect the problem itself. Traffic will be drawn to the false origin until the hijack is stopped by operator interventions.

If the hijacker returns bogus Data packets instead of black holing, NDN routers need to be able to detect bogus packets, so that they can mark the face Yellow and try alternative paths. This detection can be accomplished either by signature verification over randomly selected Data packets, or by notifications received from end consumers. Gasti et al. [7] provide a comprehensive analysis of DoS threats to NDN networks and mitigation strategies.

5. RESEARCH ISSUES

In previous sections we have argued that adaptive forwarding with datagram state can achieve robust packet delivery as well as simplified routing. There are also a few important design choices and research challenges that we would like to discuss briefly.

5.1 Forwarding State

NDN's per-packet datagram state brings with it a significant cost, both in router storage and in packet processing overhead. More specifically, since an Interest stays in the PIT of each router along the path until the corresponding Data packet returns, the number of PIT entries in a router is roughly on the order of Bandwidth×RTT/P, where P is the average size of Data packets. For a 10 Gbps link, we need about 100 k PIT entries assuming $RTT = 100$ ms and $P =$ 1000 bytes. If a router has 10 such interfaces, then its PIT needs to hold 1 M entries. Although today's core routers can handle more than 1 M entries in IP routing tables, a PIT entry is likely to be larger in size than an IP routing entry. As the routers get more interfaces and networks go faster over time, the PIT table will also grow. Therefore, one open research issue is how to reduce the size of PIT so that it can be stored efficiently on the routers.

Another research question is how to efficiently lookup and operate on the PIT. For every incoming packet, either Interest or Data, a router has to perform a lookup on the PIT using the name. PIT entries need to be added when a new Interest is received, deleted when a pending Interest is satisfied or expired, and updated when a retransmitted Interest or an Interest NACK is received. All these lookup and operations need to be performed at line speed. There are already several research efforts looking into this issue (e.g., [8]).

5.2 Forwarding Strategies

In Section 3, we presented a simple forwarding strategy design, which, according to our evaluation [9], works reasonably well in handling the network problems described in Section 4. However, it remains an open question whether we can design even better forwarding strategies to satisfy different needs of the network operators and users. Below we discuss three main issues.

How to discover a working path for a new or retransmitted Interest? There is a spectrum of strategies between trying a single interface each time (our current strategy) and flooding to all interfaces, with different trade-offs between the overhead and delay to retrieve data. We can also apply different strategies to first transmission, retransmission, and NACK. For example, the retransmissions can be forwarded to several interfaces simultaneously while the first transmission is forwarded to only a single interface.

How to use multiple paths? For Interests matching the same name prefix, our current approach is to use a single best path as long as it is able to handle all the traffic. That is, only after a problem occurs to that path, such as con-

gestion and packet loss, will a router divert excess traffic to other paths. Another approach is to proactively split traffic along multiple paths. This way, a router can keep getting feedback on data plane performance from multiple paths, and a failure may affect a smaller portion of the traffic. The two approaches are not exclusive of each other. We are currently investigating which one NDN needs, or whether NDN needs both.

How to do interface probing? Routers periodically send Interests to previously failed or unused interfaces to search for better paths. There are two questions associated with probing: when to perform probing and which interface to probe. For the first question, probing can be triggered every N seconds or every M packets. The exact numbers of N and M depend on the overhead that the network operators are willing to tolerate. As to which interface to probe, one approach is to explore a higher-ranked yellow interface with a higher probability because it has a greater chance of leading to a better path.

6. DISCUSSION

Datagram is the basic unit in packet switched networks, just like atom is the basic unit of all material. Therefore, controlling Interest forwarding using per-datagram, per-hop state offers a network the flexibility to support a wide variety of functions. While semantics of the per-datagram state can differ (i.e., what kind of information is remembered in the state and how this information is used), it is the granularity of NDN's data plane state that allows (1) loop-free (multipath) data retrieval, (2) native support of temporal and spatial multicast (i.e., servicing requests from different users that are sent either at the same or different time), (3) efficient recovery from losses of packets in transition, (4) effective flow balancing (congestion avoidance), (5) robust recovery from network problems, such as link failures and hijacks, as illustrated in Section 4, and many other important network functions.

Many attempts have been made over the years to add the above mentioned functions into IP networks, with each solution installing its own state into the network that cannot be used to solve other problems. An NDN network can use the same datagram state to provide all the functions at once, and the fundamental reason is because per-packet state is of the finest control granularity in a packet switched network.

It is conceivable that one may set up state of coarser granularity, e.g., per-connection or per-destination-prefix, for control purposes in IP or some other network architectures. The trade-off in choosing state granularity is between the functionality to be supported versus the amount of resource it consumes. When a coarser (than datagram) granularity of state is used for control purposes, it can be adequate to support a specific solution, but is unlikely to be able to support other uses, simply because different control purposes require certain state information that is incompatible with the chosen granularity. For example, IP multicast requires state information associated with {host, multicast group} pair, which is incompatible with the state information needed by XCP [4] to control congestion. Similarly, the state information needed by XCP is different from that needed by PushBack [1], a solution to mitigate DDoS attacks. Other piecemeal solutions include Pretty Good BGP [3] to mitigate route hijacks, Failure-Carrying Packets [6] to deliver packets under failures. Each of them may solve one problem well by adding its own state or mechanisms tailored to the specific problem.

NDN's datagram state does incur significant cost, which is perceived by many as infeasible based on today's technologies. For example, today's router hardware may not be able to hold a PIT or operate at wire speed. We consider these challenges as part of the research issues in realizing NDN, as discussed in Section 5.1.

7. CONCLUSION

NDN's communication model of retrieving data by names leads to a data plane design that keeps datagram state at every router. Because datagram is the basic unit in packet switched networks, this datagram state provides the flexibility to solve a host of existing problems that have resisted effective solutions up to now. In this paper we described a specific design on how to utilize this datagram state to provide high performance and resilience in an NDN network.

At the same time, we are fully aware that installing datagram forwarding state at routers brings largely open issues in terms of both technical feasibility and economical viability. The history of IP development shows that, when a new architecture solution provides significant functional advantages as well as new application opportunities, even though its overhead may seem higher and its initial implementation offers inferior performance compared to the highly engineered implementation of the incumbent architecture, research and technology advancements would eventually catch up to close the gap and even go further. Thus, we remain modestly optimistic about the future of NDN and its stateful data plane, and this paper serves as our invitation to the research community to further examine this new direction for building resilient networks.

8. REFERENCES

[1] J. Ioannidis and S. M. Bellovin. Router-based defense against DDoS attacks. In *Proc. of NDSS Symposium*, 2002.

[2] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In *Proceedings of ACM CoNEXT*, 2009.

[3] J. Karlin, S. Forrest, and J. Rexford. Pretty good BGP: Improving BGP by cautiously adopting routes. In *Proceedings of IEEE ICNP*, 2006.

[4] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *Proc. of SIGCOMM*, 2002.

[5] L. Zhang et al. Named data networking (NDN) project. Technical Report NDN-0001, PARC, October 2010.

[6] K. Lakshminarayanan, M. Caesar, M. Rangan, T. Anderson, S. Shenker, and I. Stoica. Achieving convergence-free routing using failure-carrying packets. In *Proceedings of ACM SIGCOMM*, 2007.

[7] Paolo Gasti, Gene Tsudik, Ersin Uzun, Lixia Zhang. DoS & DDoS in named-data networking, May 2012. Under submission.

[8] Y. Wang, K. He, H. Dai, W. Meng, J. Jiang, B. Liu, and Y. Chen. Scalable name lookup in NDN using effective name component encoding. In *Proceedings of IEEE ICDCS*, 2012.

[9] Y. Cheng et al. Smart forwarding: A case for stateful data plane. Technical Report NDN-0002, May 2012.