NDN Construction for Big Science: Lessons Learned from Establishing a Testbed

Huhnkuk Lim, *Nonmember*, IEEE, Alexander Ni, *Nonmember,* IEEE, Dabin Kim, *Nonmember*, IEEE, Young-Bae Ko, *Member*, IEEE, Susmit Shannigrahi, *Nonmember*, IEEE, Christos Papadopoulos, *Senior Member*, IEEE

Abstract—*Named Data Networking (NDN) is one instance of Information Centric Networking (ICN), which is a clean-slate approach that promises to reduce inefficiencies in the current Internet. NDN provides intelligent data retrieval using the principles of name-based symmetrical forwarding of Interest/Data packets and in-network caching. The continually increasing demand for the rapid dissemination of large-scale scientific data is driving the use of NDN in big science experiments. In this article, we establish the first intercontinental NDN testbed to offer complete insight into NDN construction for big science. In the testbed, an NDN-based application that targets climate science as an example big-science application is designed and implemented with differentiated features compared to previous works on NDN-based application design for big science. We first attempt to systematically address detailed analysis of why or how NDN benefits fit in big science and issues that must be resolved to improve each advantage, mostly based on lessons learned from establishing the NDN testbed for climate science. We extensively justify the needs of using NDN for large-scale scientific data in the intercontinental network, through experimental performance comparisons between classical deliveries and NDN-based climate data delivery, and detailed analysis of why or how NDN benefits fit in big science.*

Index Terms—Named Data Networking, Information Centric Networking, big science, intercontinental NDN testbed, NDN-based application, climate science, large-scale scientific data.

I. Introduction

The current Internet follows a host-centric networking paradigm and requires source/destination identifiers (e.g., IP addresses) for end-to-end sessions. As a result, the current Internet is unable to maintain pace with rapidly growing user demands in Quality of Service (QoS), mobility, multicasting, and security [1]. To resolve these architectural problems in a clean-slate manner, a new network paradigm, Information-Centric Networking (ICN) has been proposed. ICN focuses on data itself, not its location. Named Data Networking (NDN) is a promising design of ICN for realizing future internet architecture [2].

NDN uses globally unique and hierarchical names for data fetching. NDN changes the communication paradigm from the data location to the name of the data. To retrieve data, an Interest packet with a consumer's desired data name is delivered to NDN networks using name-based forwarding. The targeted Data packet from the NDN networks is returned to the consumer in the opposite direction along the same path used by the Interest packet and is cached in intermediate NDN routers [2]. The NDN platform has been implemented by NDN research communities in order to explore various applications based on the NDN architecture [2].

As the data volumes and complexity rapidly increase, big science needs to investigate new techniques for intelligent storage and data distribution over networks. Recently, NDN and big science communities have inspired innovative changes in distribution and management for big-science experimental data [3–11].

For instance, previous works of [3–4] on NDN-based application design for big science have been addressed in order to enable NDN-based climate science data searching/fetching. With regard to NDN-based applications, the authors addressed a name translator for climate science and the federating of distributed catalog systems that store and manage NDN names in order to increase the speed of discovery of desired data. The work of [5] by the co-authors of this article designed an NDN-based climate science application with a differentiated feature of providing metadata information for climate data files, while the work of [6] attempted a preliminary study on an overlay-based NDN testbed implementation using the NDN application. The work of [7] presented a preliminary study addressing opportunities and challenges to enable NDN-based intelligent data retrieval in networks for High Energy Physics (HEP). A substantial reduction in the average request delay could be achieved through the use of caching in NDN with Virtual Interest Packet (VIP) forwarding. The work of [8] showed the dependence of cache size on traffic volume and traffic patterns, and even a 1-GB cache at the edge could significantly improve data distribution. The forwarding strategy in the work guaranteed the nearest Interest forwarding path from a consumer to a producer.

However, none of these studies represent a comprehensive and thorough work on NDN construction for big science (i.e., to present fully an application design, NDN testbed implementation, various experiments, and detailed analysis to justify the needs of using NDN for big science altogether).

To provide overall insight into NDN for big science, we attempt to establish the first overlay-based intercontinental NDN testbed for climate science. The testbed is established using the NDN platform and NDN-based application for climate science, with differentiated features of providing NDN-based metadata information for climate data files and an exceptional name conversion rule for erroneous climate data files, compared to previous works of [3–4] on NDN-based application design for big science.

In establishing the overlay-based intercontinental NDN testbed, we first prove that NDN over TCP rather than UDP has better throughput performance owing to an existing problem in UDP for large-scale scientific data. Compared to the related NDN works of [3–11] on big science, an NDN congestion control scheme is first adopted into the testbed for parallel fetching experiments of climate science data.

We consider one scenario in big science to reflect NDN-based searching/fetching of large-scale climate data produced and collected from globally distributed experiments. Through first experimental performance comparisons of NDN-based delivery and classical delivery techniques for climate science, we confirm performance improvement (especially for throughput) owing to in-network caching and name-based symmetrical forwarding in NDN, which representatively justifies the need of using NDN for large-scale scientific data in an intercontinental network. The effect of pipeline size and packet size on throughput is explored in the overlay-based NDN network for climate science.

In another effort to justify the needs of using NDN for big science, we first present detail analysis of why or how NDN benefits fit in big science and issues that must be resolved to improve each advantage, mostly inspired by our experience and research. We prove that the number of files at the producer and cache replacement policy are new factors that affect the cache size. We first explore how the caching performance in NDN for climate data scales with the number of consumers and requested data files.

The remainder of this article is organized as follows. First, related works on NDN construction for big science are presented. Next, we describe NDN architecture and a common NDN platform that implements it in order to provide basic information on NDN. Third, we present the NDN testbed for a big science application and describe our experiments in detail. Fourth, detail analysis of why or how NDN benefits fit in big science and issues that must be resolved to improve each advantage are systematically addressed. These are mostly inspired by lessons learned from establishing the NDN testbed for climate science. Last, we provide conclusions.

II. Related Works

Previous works of [3–4] on NDN-based climate science application design have been addressed. While they do not cover the feature of providing NDN-based metadata information for climate data files, they have the strength of federating distributed catalog systems that store and manage NDN names in order to increase the speed of discovery of desired data in multiple domains.

The work of [5] by the co-authors of this article presented a differentiated NDN-based application for providing metadata information for climate data files, while the work of [6] attempted a preliminary study on an overlay-based NDN testbed implementation and a partial experiment. They, however, need to explore various experiments and a detailed analysis of why or how NDN fits in big science in order to justify the needs of using NDN for big science.

Through a discussion of similarities and differences of NDN in climate science and High Energy Physics (HEP), [7] addressed a preliminary work for designing an NDN-based Large Hadron Collider (LHC) network in the US. This work showed that NDN with VIP forwarding and caching at both the core and tier-2 edge nodes can contribute to achieving a 71 % reduction in the average delay per chunk compared to a no-caching case in the simulated LHC network. A substantial reduction in the average request delay could also be achieved through the use of caching in NDN, compared to NDN with VIP and no caching. The design of a practical NDN testbed and an NDN-based HEP application in a current LHC network, and experimental performance comparisons with classical delivery techniques, are still required to fully explore NDN construction for HEP.

In [8], a case study that focused on caching and aggregation effects in NDN and a forwarding strategy for improving large-scale climate data distribution in many replica data producers was addressed. This showed the dependence of the cache size on traffic volume and traffic patterns. Even a 1-GB cache at the edge can significantly improve data distribution. The forwarding strategy guaranteed the nearest Interest forwarding path from a consumer to a producer so that it distributes the load on a single producer and reduces the latency of fetching a climate data file. The authors still need to prove why or how other NDN features such as named-data driven, security for data itself, mobility, and no perceptible transport fit in climate science.

The work of [9] presented a design of the deadline-based intelligent data transfer protocol for NDN, which creates a reserved path and bandwidth up to a cache. The authors still need to evaluate its performance through experiments. The authors analytically investigated the possible bandwidth savings in HEP using NDN request aggregation. If 50% of the duplicate requests were aggregated, the bandwidth reduction came down to 79%.

The work of [10] developed an NDN consumer and producer following the CERN Virtual Machine File System (CVMFS) in order to serve HEP data based on NDN. This is a distributed file system by the HEP community that provides content addressable names that are useful for NDN. Architectures for the consumer and producer need to be presented, together with experiments using them.

Last, the SDN-Assisted NDN for Data Intensive Experiments (SANDIE) project has aimed to develop an NDN-based scientific data distribution architecture for data-intensive science such as HEP since 2017 [11]. This provides optimization of NDN features such as caching, forwarding, and congestion control. The project also plans to implement an HEP data naming scheme and improve the NDN system scalability. It requires more progressed to show actual results.

As a common weak point in these related works, they do not address a comprehensive and thorough work on NDN construction for big science. To cover overall insight into NDN construction for big science, we fully address an NDN-based climate science application with differentiated features compared to previous works [3–4], the first intercontinental NDN testbed implementation, and various experiments to justify the needs of using NDN for climate science altogether. In another effort to justify the needs of using NDN for big science, we systematically present detailed analysis of why or how NDN benefits fit in big science and issues that must be resolved to improve each advantage, mostly inspired by our experience and research.

III. NDN Architecture and Platform

A. NDN Architecture

Data fetching is achieved using two kinds of packets: an Interest and a Data packet. To fetch data, a consumer puts the unique and hierarchical name of the targeted data into an Interest packet, and the Interest packet is forwarded to a router (or producer) based on the name carried in it. A producer or router in the NDN network returns data corresponding to the name in the Interest packet, with a signature by the producer's key.

To forward the Interest packet, the NDN router uses a Pending Interest Table (PIT), a Forwarding Information Base (FIB), and a Content Store (CS), as shown in Fig. 1. The PIT entries record data names in Interest packets that an NDN router has forwarded but not yet answered, and their incoming interfaces. When an Interest packet reaches an NDN router, the CS checks whether the data name carried in the Interest packet exists. If so, the router delivers the Data packet to the incoming interface of the Interest packet. Otherwise, the router checks for the presence of a matching PIT entry corresponding to the data name. In the case of a lookup hit for the matching PIT entry, the router adds the input interface of that Interest packet to the PIT entry. In the case of a lookup miss in the PIT, the router forwards the Interest packet to an outgoing interface based on the FIB. In the case of a lookup miss in the FIB, the router drops the Interest packet [2].

To forward the Data packet, the NDN router uses a PIT and a CS, as shown in Fig. 1. The Data packet is delivered to the requesting consumer using the PIT that is set up by each Interest packet at each hop [2]. The CS is used to temporarily store Data packets that the router has accepted in order to answer future Interest packets. When a Data packet arrives at a router, it looks up the matching PIT entry for the data name. In the case of a lookup hit in the PIT, the router multicasts the Data packet to all downstream interfaces described in the matching PIT entry. It then eliminates that PIT entry and stores the Data packet in the CS. In the case of a lookup miss for the matching PIT entry, the router discards the Data packet.

In Fig. 1, a climate data fetching scenario based on Interest and Data packet forwarding in an NDN network is shown. Two consumers request the climate data of interest through the NDN network. The Interest and Data packet forwarding procedure is indicated by sequence numbers on the Interest and Data packet arrows. NDN routers contain the forwarding prefix (*/csu/atmos/*) with an assigned outgoing face in each FIB, which guides the Interest packets to the next hops. Interest/Data packet exchange in reverse directions on the same path removes the need for source and destination addresses, in contrast to the current Internet. Consumer 1 fetches the climate data segment of */csu/atmos/hur_Amon.../v1/s2* from the producer, while consumer 2 fetches the climate data segment of */csu/atmos/hur_Amon.../v1/s1* that has been already stored in the CS of the middle NDN router. The two data segments are cached by the routers on the reverse paths to the consumers that requested them. To obtain a large-scale scientific data file that consists of many packets, the consumer issues an Interest packet with a sequence number (for example, *s1* and *s2* in Fig. 1), and the producer (or router) then returns a data packet corresponding to a sequence number.

B. Common NDN Platform

The NDN platform has been improved for NDN architecture research and development by NDN research communities [2]. The NDN Forwarding Daemon (NFD) has been implemented to accomplish the Interest and Data packet forwarding procedures as a common and main segment of the NDN platform. NFD contains the following modules: Forwarding, Faces, Tables, Management including the Routing Information Base (RIB) management, and Common Service (ndn-cxx library, Core, NFD tools and NDN essential tools).

The Forwarding module implements basic packet processing pathways for Interest/Data forwarding, which interacts with Faces, Tables, and Strategies. It provides a framework to support different forwarding decisions in the Interest/Data forwarding procedure. Packet sending and receiving occur via a face that is the same as the physical network layer. Tables are data structures needed for FIB, CS, PIT, and Strategy Choice in order to serve modules in the NFD.

The Management module can be classified as a set of managers (i.e., FIB, Face, Strategy Choice, and Forwarding status managers). Using the FIB manager, a user can add or erase FIB entries. The RIB management is used for routing information registration and management with name prefixes that were created by a system administrator or individuals. An ndn-cxx library, a Core service, NFD tools, and NDN essential tools offer various common services shared between NFD modules.

IV. NDN Testbed for a Big Science Application

We describe an intercontinental NDN testbed for climate modeling science as an example big science application. We demonstrate climate modeling data fetching, specifically of Coupled Model Intercomparison Project 5 (CMIP5) data [12], across continents on the NDN testbed. The performance is compared to those of classical delivery techniques in a climate modeling application in order to verify the need of using NDN for a climate science application. The testbed represents the first overlay-based intercontinental NDN network for big science.

A. NDN Testbed for Climate Science Application

An NDN testbed was established to explore name-based climate data delivery across continents, as shown in Fig. 2.

The upper part of the figure presents physical connections for the testbed. The GLObal RIng network for advanced Applications Development (GLORIAD) research network with 10 Gbps links was used to provide direct connections between continents and to support a much greater BW than a consumer/producer speed (1 Gbps) between them [13]. A TCP tunnel between the NDN gateway routers on both continents was established to separate NDN traffic logically from other traffic in legacy IP routers and to support overlay-based NDN communication in the intercontinental network.

In establishing the overlay-based intercontinental NDN testbed, we attempted to apply TCP and UDP in overlay IP networks and first proved that NDN over TCP rather than UDP has better performance for large-scale scientific data. TCP in the current NDN platform (NDN over TCP) can handle large NDN packets through packet fragmentation/reassembly, and results in proper performance under overlay connection. UDP in the current NDN platform (NDN over UDP) does not properly support large NDN packets (more than 64 kB) yet, and requires an additional function of packet fragmentation. Although UDP that has no congestion control and packet retransmission is suitable for NDN from the aspect of transport architecture, we found that UDP for large-scale scientific data resulted in unsatisfying throughput performance. The loss of a single UDP segment that results in the retransmission of a large NDN data packet consisting of many UDP segments, led to the degraded throughput performance. Therefore, we adopted TCP in the overlay-based NDN testbed for large-scale scientific data.

Congestion control in NDN is an open research area. There are few proposals but none of them has been completely adopted as an application in the NDN platform yet. We first attempted to adopt an NDN congestion control scheme (i.e., the NDN-Additive Increase and Multiplicative Decrease (AIMD) algorithm) [14] in the overlay-based intercontinental NDN tested for big science. In parallel fetching experiments without any caching effect, we could improve throughput by adopting it in NDN networks.

The lower part of the figure describes the NDN testbed in terms of platform architecture. The architectures for both a consumer and producer are implemented using the NDN platform and our NDN-based climate science application (i.e., a front-end system for a consumer and a back-end system for a producer), as shown in Fig. 2 [5]. Our NDN-based climate science application has differentiated features for providing NDN-based metadata information for climate data files and an exceptional name conversion rule for erroneous CMIP5 files, compared to the previous works of [3–4] on NDN-based application design for big science. The previous works mainly had the strength of federating of distributed catalog systems that manage NDN names in order to increase the speed of discovery of desired data. The architecture for a router can be implemented using only the NDN platform.

1) Application systems for consumers and producers

For consumers, NDN.JS and a Firefox add-on (i.e., the open sources developed by the NDN community) are used to support NDN-based data searching and fetching using a conventional web browser. For producers, we improved repo-ng (the open-source NDN repository developed by the NDN community) to establish an NDN repository with functions needed for climate science. We developed and improved an NDN-based climate science application for both consumers and producers [5].

Using the front-end system, consumers search potential CMIP5 files and fetch the desired file with the data name contained in an Interest packet from the NDN testbed. Using the back-end system, producers publish unique NDN data by first converting a flat CMIP file name to a hierarchical NDN name. They then establish a repository to store and manage original CMIP5 files and their NDN name lists and metadata sets.

2) Design concept for NDN-based climate science application

The main design concept for our NDN-based climate modeling application is focused on two features. First, the front-end and back-end systems are designed to support NDN-based searching/fetching of CMIP5 files using name-based data management and publication. Second, users do not need to be concerned about an underlying network architecture (NDN or traditional IP network) while they are interested in user interfaces for data searching and fetching and those follows the existing climate modeling application workflow (i.e., the ESGF system [12]). Brief overviews of the front-end and back-end systems follow.

3) Front-end system for consumers

The front-end system consists of 1) a Front-end NDN Engine (FNE) that processes search results returned from the NDN producer application after requesting an Interest packet to search potential CMIP5 files, and 2) a user interface that allows web-based access to the NDN testbed and supports climate modeling data searching and fetching based on an NDN name [5].

After an Interest packet to search potential CMIP5 files is sent by a consumer, the FNE then receives datasets that include NDN name lists and their corresponding metadata sets, which are the detailed descriptions of real climate data files and are used to determine which data files are necessary. The FNE next separates individual pairs of an NDN name and its metadata information from each received dataset.

The FNE reconstructs the received NDN name to an original CMIP5 file name in order to let a user know the complete CMIP5 data file name. The FNE then inserts the retrieved CMIP5 file name lists into the browser. This enables the user to select and fetch the desired CMIP5 files. The FNE is also used for metadata browsing: when a user selects a desired file name in a web browser, the FNE extracts metadata information corresponding to the name and launches it in the web browser. The metadata browsing is a function needed for providing users with metadata for climate data files.

In addition, the FNE attaches sequence numbers to the end of NDN names, and consequently the NDN name with a sequence number in an Interest packet is carried into the NDN testbed.

All CMIP5 file names found by a search Interest packet are indicated in the UI screenshot, as shown in Fig. 3 (a). To provide more detailed information for a CMIP5 file, the metadata for the file is shown, which is a differentiated user-friendly function compared to previously introduced application systems [3–4]. Their UI had the strength of name auto-completion in the search box as a user-friendly function. To fetch a desired CMIP5 file, Interest packets with sequence numbers are carried into the NDN testbed. A producer or router sequentially returns a set of segmented data packets that correspond to the CMIP5 file's NDN name to the requesting consumer. The UI screenshot for fetching a desired CMIP5 file is shown in Fig. 3 (b). Application systems for consumers can be implemented in a proxy server so that classical IP hosts interact with the proxy for NDN-based searching/fetching and the proxy communicates with NDN networks.

4) Back-end system for producers

The back-end system consists of an NDN name translator/metadata manager, an NDN repository composed of a data container and a name/metadata container, and an NDN producer application [5]. The back-end system basically follows the conventional ESGF workflow [12], which returns the searched file name lists and corresponding metadata sets and transfers a desired climate data file by user request.

The name translator converts the original CMIP5 file name to a hierarchical NDN name format for all climate data files, as shown in Fig. 3 (c). It allows for an exceptional conversion rule for erroneous CMIP5 files, while the previous works of [3–4] addressed a translator to convert CMIP5 file names without any exceptional conversion rule. Converted NDN names and their metadata sets as well as original CMIP5 files, are stored in the NDN repository in order to provide more detailed information for CMIP5 files. The syntax of the original CMIP5 file name follows the Data Reference Syntax (DRS) standardized by ESGF [12]. The metadata manager extracts the metadata sets from each CMIP5 file and manages them in order to provide detailed information for CMIP5 files.

The NDN repository is used to contain CMIP5 files and their datasets and allows for data-centric access and file fetching. The NDN repository is divided into two parts: a data container to store CMIP5 files and to support data fetching, and a name/metadata container to store converted NDN names and their metadata sets separately.

The NDN producer application processes Interest packets issued by consumers. When receiving an Interest packet carried to search potential CMIP5 files, the NDN producer application attempts to identify the data name in the Interest packet among NDN name lists in the name/metadata container. In the case of a lookup hit, the corresponding NDN names and their metadata sets are returned to the requesting consumer.

When the back-end system starts, it parses all of the original CMIP5 file name attributes to obtain and produce NDN name components. During name translation, several components that cannot be acquired from the original name such as the *product* value can be obtained from its metadata information. As a result, the back-end system prepares all name components that are required to convert an original CMIP5 data file name to an NDN name. Consequently, it produces hierarchical NDN names using the required name components.

B. Demonstration of Climate Modeling Data Delivery based on the NDN Testbed

The ESGF was constructed to distribute and manage efficiently climate modeling data in global networks [12]. Its architecture is based on a peer-to-peer federated infrastructure for data management in current IP networks. It provides users with easy access to distributed climate data centers. Owing to its end-to-end transport control mechanism in the current Internet, it faces potential problems of high latency and corrupted ratios [5].

Through first performance comparisons of classical delivery techniques (the ESGF system and FTP/HTTP) and NDN-based climate modeling CMIP5 data delivery in serial fetching experiments, we confirm the throughput improvement owing to in-network caching and name-based symmetrical forwarding in our NDN testbed for climate science. These are shown in Fig. 4 and Fig. 5, which justify the need of using NDN for disseminating large-scale climate data in an intercontinental network. In both figures, the effect of pipeline size and packet size on throughput is explored in the overlay-based NDN network for climate science.

Without TCP tuning on the NDN gateway routers, we attempt to investigate an optimum pipeline size in the consumer, in order to fetch a climate science CMIP5 file with the highest throughput. The pipeline size is the number of Interest packets that a consumer can successively send without waiting for responding Data packets [6]. Fig. 4 (a) shows throughputs measured for different pipeline sizes as a function of packet size (i.e., data granularity). Throughput results were based on both remote data retrieval from the producer with a 1-Gbps interface to the one consumer and local data fetching from caches to the other consumers in the testbed. A 4-GB CMIP5 file (i.e., *cl_Amon_CESM1-CAM5_rcp60_r1i1p1 _205001 210012.nc*) was fetched by four consumers in serial order. As the packet size increased, the throughput increased accordingly. This was due to the reduction of the signing/verification cost. For the climate modeling application, the throughput no longer improved after a pipeline size greater than 6. Each throughput result owing to fetching from the remote producer and local distributed caches is shown in Fig. 4 (b) and Fig. 4 (c), respectively.

With the optimum pipeline size of 6, as shown in Fig. 4 (a), the same CMIP5 file was sequentially fetched by the consumers across the NDN testbed without caching. The NDN throughput without caching improved with increasing packet size and saturated at 140 Mbps over the 1.5-MB packet size, as shown in Fig. 4 (d). Using classical delivery techniques (HTTP and FTP), the identical CMIP5 file was consecutively fetched by the consumers along the NDN testbed routing path. HTTP- and FTP-based deliveries had throughputs of 140–150 Mbps with a fixed packet size of 1.5 kB. Lastly, the same file at the NCAR node (i.e., the National Center for Atmospheric Research node, which is the closest ESGF data center node from the producer) in the state of Colorado (USA) was fetched by the consumers along the routing path configured for the ESGF infrastructure. Delivery using HTTP in the ESGF system resulted in the lowest throughput (i.e., 100 Mbps) with a fixed packet size of 1.5 kB. For the NDN-based delivery with caching, intelligent data retrieval from the distributed caches in local intermediate NDN routers leveraged approximately 4.5–6.5 times the throughput improvement over the 1.5-MB packet size compared to conventional delivery techniques (FTP/HTTP/ESGF), as shown in Fig. 4 (d).

The principle of location-independent data access in NDN leveraged the reduction of redundant traffic in the network, which resulted in changing traffic flow patterns in the network. For the NDN-based delivery without caching and HTTP/FTP-based deliveries from the remote data source, lower end-to-end throughputs (i.e., 140–150 Mbps) compared to the data source speed of 1 Gbps derived from transport control between the gateway routers without TCP tuning.

With TCP tuning on the NDN gateway routers, the throughput of NDN-based delivery saturated at 900 Mbps over a pipeline size of 20 and a packet size of 3 MB, as shown in Fig. 5 (a). With the optimum pipeline size of 20, as shown in Fig. 5 (a), the same CMIP5 file was sequentially fetched by the consumers across the NDN testbed without caching. The NDN throughput without caching improved with increasing packet size and saturated at 850 Mbps over the 3-MB packet size, as shown in Fig. 5 (b). Using classical delivery techniques (HTTP and FTP), the identical CMIP5 file was consecutively fetched by the consumers along the NDN testbed routing path. HTTP- and FTP-based deliveries had a saturated throughput of 850 Mbps with a fixed packet size of 1.5 kB, as shown in Fig. 5 (b). For the NDN-based delivery with caching, intelligent data retrieval from the local distributed caches leveraged approximately 40 Mbps the throughput improvement over the 3-MB packet size compared to classical delivery techniques (FTP/HTTP), as shown in Fig. 5 (b).

To explore the NDN caching effect on latency, first, we attempted to measure the average latency per packet, which is needed for fetching a climate data file (4 GB) sequentially by four consumers in the established testbed. The average latency per packet for NDN-based delivery with caching is derived from the latency per packet on fetching from the remote producer by one consumer and fetching from locally distributed caches by other three consumers. The average latency per packet for other delivery techniques reflects the delivery from the remote producer only. Fig. 6 (a) and Fig. 6 (b) show the average latency per packet measured as a function of packet size for NDN-based delivery and classical delivery techniques (HTTP and FTP) for cases with and without TCP tuning, respectively.

For a larger NDN packet arrival, a consumer needs to wait for the arrivals of more fragmented MTU packets in the overlay-based NDN network (e.g., for a 1-MB NDN packet, a consumer will wait for the arrivals of all fragmented MTU packets of which it consists). Therefore, the average latency per packet for NDN-based delivery in both figures accordingly increased as the packet size increased. NDN-based delivery with caching for the no TCP tuning case, resulted in the smallest latency on the optimum packet size of 1.5 MB depicted in Fig. 4 (a), while that for the TCP tuning case resulted in nearly the same latency as those of HTTP and FTP on the optimum packet size of 3 MB depicted in Fig. 5 (a). The average latency per packet for both HTTP and FTP with a much smaller packet size was dominantly derived from the propagation delay. Big science applications dealing with large-scale scientific data are loss-sensitive. They focus on fetching scientific data files fast without packet loss.

Second, we attempted to measure the average user latency for NDN-based delivery and classical delivery techniques, which means the average delivery time required to serially fetch a climate data file of 4 GB by four consumers in the testbed. The TCP tuning case resulted in much better user latency improvements for all delivery techniques compared to the no TCP tuning case, as shown in Fig. 6 (c) and (d). For both cases, better delivery times for NDN-based delivery with caching could be achieved because duplicated data file requests could be satisfied from local caches.

The delivery time from the remote producer for all delivery techniques with TCP tuning was much shorter than that of the no TCP tuning case. This led to a much smaller user latency improvement due to caching in NDN for the TCP tuning case compared to the no TCP tuning case.

With TCP tuning to allow for larger dynamic window sizes in the IP network, better end-to-end throughputs could be achieved due to throughput improvement in the IP network of the overlay-based NDN network. Significant end-to-end throughput improvement due to TCP tuning for all delivery techniques led to smaller performance improvements between NDN-based delivery with caching and classical delivery techniques compared to the case without TCP tuning. We first explored the NDN caching effect on the average latency per packet and the user latency in the testbed for climate science data. The intelligent data retrieval from locally distributed caches led to the latency improvement for both NDN with no TCP tuning and NDN with TCP tuning.

V. NDN Benefits and Issues for Big Science

We address detailed analysis of why or how NDN benefits fit in big science, and issues that must be resolved to improve each advantage, mostly inspired by our experience and research.

A. *Caching*

The NDN router temporally stores data chunks and returns them to the requesting consumer when it has contents to satisfy Interest packets in the CS. Using this function, NDN can substantially reduce overall traffic volume and user latency in the network. In-network caching is known to be more efficient for static rather than dynamic content such as video streaming or voice [5]. Large-scale scientific data such as CMIP5 data in climate science and Large Hadron Collider (LHC) data in HEP is relatively static. They are thus expected to perform well in terms of cache hit ratio and utilization [5, 7]. The impact improves delivery time and throughput, which increases the data access speed to permanent storage, with the help of saved bandwidth and changed data transfer patterns in networks.

Where and how to cache and manage large-scale scientific data in distributed memories with limited size forms a variety of research issues [8, 15]. Among these issues, a simulation study that showed the caching and aggregation effects on NDN for large-scale climate data was presented [8]. The study manifested the dependence of cache size on traffic volume and traffic patterns. One result in the work shows that even a 1-GB cache size in edge NDN nodes can provide a significant reduction in server hits and network traffic. Since cache size depended on the traffic volume and traffic pattern, the results for the optimal cache size were different in different week traces [8].

It is already known that the content popularity distribution follows a Zipf-like distribution [8]. The α parameter of the Zipf law determines the skewness of distribution and is related to the user's request behavior. A higher value of α determines that the requests are more concentrated on a specific content, which means some specific content is more popular. Different applications have different skewness values depending on the nature of the application. For climate science application, a skewness parameter value of $\alpha = 1.15$ best fits the user request pattern [8].

To prove another new factor that affects cache size, we simulated cache hit ratio as a function of cache size for different numbers of climate data files (catalog size) at the producer in our NDN testbed, as shown in Fig. 7 (a). We had 10 consumers requesting 5 files of size 1 GB, 1.1 GB, 1.2GB, 1.3 GB, and 1.4 GB (catalog size: 5 files) at the producer, and 2 files of size 1.2 GB and 1.3 GB (catalog size: 2 files), respectively. A cache at the NDN gateway router for the consumers and a skewness parameter value of $\alpha = 1.15$ were considered. Fig. 7 (a) shows that the cache hit ratio is saturated at a smaller cache size for a catalog size of two files, while it is saturated at a larger cache size for a catalog size of five files. The increased cache size for larger catalog size was owing to relatively various contents in a higher number of files and a larger volume at the producer.

In Fig. 7 (b) and Fig. 7 (c), we first attempt to explore how the caching performance in NDN scales with the number of consumers and requested data files. All of the files are of the same size of 1.3 GB (i.e., the most dominant size in CMIP5 climate data files) and range from 10 files to 40 files. We also vary the number of consumers from 10 to 30 to investigate its effect on cache hit ratio. The size of the cache at the NDN gateway router for consumers is fixed throughout the simulation (40,000 chunks).

In Fig. 7 (b), the cache hit ratio increased as the number of consumers increased, because increasing the number of consumers increases the probability of requesting the same file by more consumers. Thus, the cache hit ratio increases. In Fig. 7 (c), increasing the number of requested files decreases the probability of requesting the same file by consumers. The contents in the cache were replaced very quickly, which results in reduction of the cache hit ratio.

The cache replacement policy was also a new factor that affected cache size. Least Frequently Used (LFU), which considers the content popularity, guarantees the highest cache hit ratio with the smallest cache size, compared to other well-known cache replacement policies, i.e., Least Recently Used (LRU), FIFO, and Random.

B. *Named data-driven*

Big science applications including climate science and HEP are known to be highly data-driven [4, 7, 12]. In the current climate modeling application that uses the ESGF system, a content owner's location is a major factor. The system should manage information of data center locations as well as CMIP5 file names [5]. The named data-driven NDN requires content name information only and is thus expected to serve big science applications more efficiently by using location-independent data access.

Each application should be able to support a hierarchical naming rule in order to leverage the name-based forwarding mechanism of NDN. The climate research community provides a well-structured naming syntax, the Data Reference Syntax (DRS), to manage efficiently distributed large-scale data with globally flat/unique names [12]. In addition, the HEP research community currently supports flat/unique names for all LHC data.

An NDN-based application should be designed for consumers and producers, in order to support name-based data searching/fetching and to convert flat/unique names to hierarchical/unique ones. The HEP community is currently developing an NDN-based application following the workflow of the Xrootd system (i.e., a data distribution and management system developed by the HEP community) [3, 7].

For continually increasing scientific data, the effort to speed up discovery of a target data name is a crucial issue. Federating distributed catalog systems that store and manage NDN names can be an alternative to increasing the speed of discovery of desired data in multiple domains [3]. Each catalog process is simply configured with the naming schema for the community it wishes to support. Multiple catalogs can even coexist on the same hardware [3].

C. *Security*

Large-scale scientific data volumes and complexity are continually increasing in data centers. Security for scientific data itself is essentially needed for big science. In NDN, security is not focused on communication channels (i.e., hosts/servers) but on the content data itself [2]. NDN provides security for each large-scale scientific data with the help of publisher signatures. This mitigates existing DDoS attacks for a specific source from malicious users because it contains no information about its location.

Recently, the Interest flooding attack that targets specific content in NDN (which is similar to a DDoS attack in the current Internet) was addressed [16]. In case of flooding the network with interests for denial of service attack in NDN, only one interest hits the server for a particular content name, while the remaining interests are aggregated at intermediate routers and are unable to inundate the content producer (server). Likewise, the following interests for the same content would be satisfied by the in-network cache, making NDN more resistant to such types of denial of service attacks. Whereas legacy IP-based big science networks consisting of numerous connected sites are prone to denial of service attack with lots of scenarios. In short, Interest aggregation and in-network caching make NDN resistant to many scenarios of interest flooding attacks, thus significantly narrowing the number of interest flooding attack scenarios. This feature justifies why we have to establish NDN-based big science networks with more reliability and robustness for denial of service attack, instead of classical networks.

The only effective and possible Interest flooding attack scenario in NDN is to send a flood of interests each with different non-existing random or dynamic content names to a victim namespace from a different location. Each interest will create a new PIT entry resulting in PIT exhaustion at routers, thus causing a denial of service for legitimate interests. Several techniques in the literature to mitigate this type of Interest flooding attack have been introduced [16].

D. *Forwarding*

NDN forwarding essentially allows multicasting from an intermediate router rather than a producer for multiple requests of popular scientific data. This prevents data explosion resulting from redundant large-scale scientific data traffic in whole networks.

Large-scale scientific data should be segmented into an optimum data size (i.e., data granularity or packet size) by the NDN producer and transferred to NDN networks. When a data size over 1.5 kB is applied to an overlay-based NDN network, fragmentation to MTU size (1.5 kB) for the IP network and reassembly to a controlled data size for NDN networks occur at the NDN gateway routers in the overlay-based NDN network. An application can use an optimum data size in NDN networks, besides having MTU size constraint in IP networks only.

An optimum pipeline size should be determined by the NDN consumer. Optimum pipeline size was dependent on the achievable throughput in an overlay-based NDN network. To obtain better throughput, controlling the data size and pipeline size is an issue in forwarding large-scale scientific data in NDN networks of an overlay-based NDN network.

The procedures depicted in Fig. 4 (a) and Fig. 5 (a) can be used to determine an optimum pipeline size and an optimum packet size for other big science applications. For congestion cases in a network, dynamic control of pipeline size by an NDN congestion control scheme can be more beneficial to obtain better throughput. The investigation of an analytic model for the optimum pipeline size is required to reduce controlling costs in the overlay-based NDN network for large-scale scientific data.

The NFD performance in the NDN platform must be further improved to support greater bandwidth (e.g., 10 Gbps) [4]. Distributing the load on a single producer and reducing the user latency on fetching large-scale scientific data are required. A forwarding strategy in many replica data producers that guarantees the nearest latency path from a consumer to a data producer and forwards the Interest packet to the nearest producer has been addressed [8]. Dynamic forwarding and caching using the Virtual Interest Packet (VIP) can be supported for load balancing in multiple domains [15].

E. *Mobility*

NDN originally guarantees consumer-side mobility because the nature of its communication is data-centric rather than host-centric, as in the current Internet [17]. For consumer-side mobility, a mobile consumer reissues an Interest packet to the NDN network and can continually retrieve desirable large-scale scientific data.

If a future data production workflow enables mobile data sources to create large-scale scientific data directly with potential data sensing, computing, or processing abilities, data source movement events can occur. In such a case, a robust and seamless producer mobility scheme in NDN is required to support such a challenging producer mobility scenario in big science. The work of [17] suggests mobility schemes that can be supported for such a challenging producer mobility scenario.

F. *No perceptible transport*

Low end-to-end throughput for a source speed frequently occurs as a result of the end-to-end transport control mechanism in classical networks. When fetching large-scale scientific data across the intercontinental NDN over TCP networks, a significantly improved end-to-end throughput can be achieved with TCP tuning on the gateway routers. NDN routers manage the Interest forwarding rate in a hop-by-hop manner such that NDN removes the dependency on the end hosts to accomplish transport control [2, 18]. No perceptible transport in NDN is expected to result in an improved end-to-end throughput for large-scale scientific data compared to classical networks, if the NDN platform becomes more mature.

The NDN congestion control mechanisms can be divided into three categories: receiver-driven, hop-by-hop control, and hybrid [14, 18]. NDN congestion control algorithms in them are still evolving. For the hop-by-hop based congestion control, Nack packets (Duplicate, No data, and Congestion) are used to inform downstream NDN nodes of status information on an upstream NDN node [14, 18]. In case a downstream node receives a Nack-Congestion, the downstream node/consumer reduces its Interest rate using the AIMD algorithm. When a downstream node receives a Data packet, the downstream node/consumer increases its Interest rate using the AIMD algorithm [14, 18]. NDN-AIMD adopted in our testbed could work well together with TCP to improve throughput under congestion situations in parallel fetching experiments, even though an overlap between the two congestion control logics was created. NDN congestion control schemes have been mostly designed for pure NDN networks. A practical NDN congestion control scheme to consider overlay links together should be joined to transfer large-scale scientific data more rapidly across overlay-based NDN networks in a global network [14].

NDN over UDP for large-scale scientific data did not guarantee good performance. To obtain a satisfactory throughput through NDN over UDP for large-scale scientific data, a large NDN packet is also needed. The loss of a single UDP segment led to the retransmission of a large NDN data packet consisting of many UDP segments. This resulted in poor performance in NDN over UDP delivery. In order to overcome the existing problem in UDP for large-scale scientific data, application needs to be intelligent to achieve an acceptable throughput by supporting a congestion control mechanism. The work of [19] suggests UDP congestion control schemes in application.

VI. Conclusion

To offer complete insight into accelerating NDN construction without pitfalls in big-science applications, we established an overlay-based intercontinental NDN testbed for climate science. NDN-based climate science application design was essential to support name-based climate data searching/fetching. Our application had differentiated features compared to those of previous works, mainly with regard to name-based user-friendly searching. Our testbed represented the first overlay-based intercontinental NDN network for big science. The local distributed caching in NDN-based delivery resulted in throughput improvement compared to classical delivery techniques. TCP tuning led to a better end-to-end throughput in the overlay-based NDN network, due to the allowance of larger dynamic window size in the IP network. Lastly, we addressed detailed analysis of why or how NDN benefits fit in big science, and issues required for improving each advantage. Based on various experiments in the testbed and detailed analysis of why or how NDN benefits fit in big science, we extensively verified the justification of using NDN for large-scale scientific data in an intercontinental network. An intelligent UDP congestion control in application is required to fetch large-scale scientific data through the NDN over UDP.

References

[1] B. Ahlgren *et al.*, “A Survey of Information-Centric Networking,” *IEEE Commun. Mag.*, vol. 50, pp. 26–36, July 2012.

[2] L. Zhang *et al.*, “Named Data Networking,” *ACM SIGCOMM Computer Commun. Rev.*, vol. 44, pp. 66–73, July 2014.

[3] C. Fan *et al.*, “Managing Scientific Data with Named Data Networking”, in *Proc. 5th Int. Workshop Network-Aware Data Manage.*, pp. 1–7, Nov. 2015.

[4] C. Olschanowsky, S. Shannigrahi, and C. Papadopoulos, “Supporting Climate Research using Named Data Networking”, in *20th IEEE Int. Workshop Local and Metropolitan Area Networks*, pp. 1–6, May 2014.

[5] D. Kim *et al.*, “Implementation of a Front-end and Back-end NDN System for Climate Modeling Application”, in *Inform. and Commun. Technol. Convergence 2015 Int. Conf.*, pp. 554–559.

[6] H. Lim, A. Ni, D. Kim, and Y.-B. Ko, "Named Data Networking Testbed for Scientific Data,” *2nd International Conference on Computer and Communication Systems*, pp. 65-69, July 2017.

[7] S. Shannigrahi, *et al.*, “Named Data Networking in Climate Research and HEP Applications”, in *21st Int. Conf. on Computing in High Energy and Nuclear Physics, IOP Publishing. Journal of Physics: Conference Series,* vol. 664, pp. 1–8, 2015.

[8] S. Shannigrahi, C. Fan, and C. Papadopoulos, “Request Aggregation, Caching, and Forwarding Strategies for Improving Large Climate Data Distribution with NDN: A Case Study,” *2017 Proceedings of Information Centric Networking*, pp. 54-65, 2017.

[9] S. Shannigrahi and C. Fan, and C. Papadopoulos, “Named Data Networking Strategies for Improving Large Scientific Data Transfers”, *in IEEE Int. Workshop on Information Centric Networking Solutions for Real World Applications*, pp. 1-6, May 2018.

[10] M. Alhowaidi, B. Ramamurthy, B. Bockelman, and D. Swanson, “The Case for Using Content-Centric Networking for Distributing High Energy Physics Software”, *in 37th IEEE Int. Conf. on Distributed Computing Systems*, pp. 2571-2572, June 2017.

[11] https://www.nsf.gov/awardsearch/showAward?AWD_ID=1659403

[12] L. Cinquini *et al.*, “The Earth System Grid Federation (ESGF): An open infrastructure for access to distributed geospatial data,” *Future Generation Comput. Syst.*, vol. 36, pp. 400–417, July 2014.

[13] http://www.gloriad.org/

[14] Y. Ren *et al.*, “Congestion Control in Named Data Networking – A Survey,” *Computer Communications*, vol. 86, pp. 1-11, Apr. 2016.

[15] E. Yeh *et al.*, “VIP: a Framework for Joint Dynamic Forwarding and Caching in Named Data Networks”, in *Proc. 1st Int. Conf. Information-Centric Networking,* pp. 117–126.

[16] L. Wang *et al.*, "Economic Levers for Mitigating Interest Flooding Attack in Named Data Networking,” *Mathematical Problems in Engineering*, vol. 2017, pp. 1-12, June 2017.

[17] D.-H. Kim *et al.*, "End-to-End Mobility Support in Content Centric Networks,” *International Journal of Communication Systems*, vol. 28, Iss. 6, pp. 1151-1167, Apr. 2015.

[18] C. Yi *et al.*, “A Case for Stateful Forwarding Plane,” *Computer Communications*, vol. 36, pp. 779-791, Apr. 2013.

[19] M.-H. Wang, L.-W. Chen, P.-W. Chi, and C.-L. Lei, “SDUDP: A Reliable UDP-Based Transmission Protocol Over SDN”, *IEEE Access*, vol. 5, pp. 5904-5916, May 2017.

Biographies

Huhnkuk Lim received his Ph.D. in Information and Communications from the Gwangju Institute of Science and Technology (GIST), South Korea, in 2006. He joined the Korea Institute of Science and Technology Information (KISTI) in 2006 as a principal researcher. His recent research area cover NDN for large-scale scientific data and edge computing. He has published more than 60 refereed papers in the areas of Information Centric Networking (ICN), cloud computing, and user-driven network/computing resource control in archival journals and conference proceedings. He served as an associate editor of the Journal of Internet Computing and Services from 2007 to 2013.

Alexander Ni received his B.S. in Information Systems from the International Information Technologies University, Kazakhstan in 2014. Since 2015, he has been pursuing an M.S. in S&T Information Science from the University of Science and Technology (UST), South Korea. His research area covers various issues in NDN design for big science applications.

Dabin Kim received her Ph.D in Computer Engineering from Ajou University, South Korea in 2017. Her research interests are in the area of wireless networks, tactical networks and information-centric networks.

Young-Bae Ko is currently a professor in the Department of Software at Ajou University, South Korea. Prior to joining Ajou in 2002, he was with the IBM T.J. Watson Research Center, New York, as a research staff member in the Department of Ubiquitous Networking and Security. He received his Ph.D. in Computer Science from Texas A&M University, College Station, TX, in 2000. His recent research interests are in various topics of Future Internet, 5G, Connected cars, and Internet of Things (IoT). He has served in various activities, most notably as the general chair in IEEE SECON 2012 and editorial board of Elsevier Digital Communications and Networks.

Susmit Shannigrahi is a Ph.D. student at Colorado State University. He received his M.S in Computer Sc. from Jadavpur Univeristy, India. He is currently working on supporting large data based scientific applications over Named Data Networking.

Christos Papadopoulos is currently a professor at Colorado State University. He received his Ph.D. in Computer Science in 1999 from

Washington University in St. Louis. He currently works on future Internet architectures, network security and measurements. Active projects include Named Data Networking (NDN), Supporting Climate Applications over NDN, Netbrane, PREDICT, Retro-Future, WIT-II and Making Internet Routing Data Accessible To All. He is a senior IEEE member and has served on numerous ACM and IEEE conference program committee.

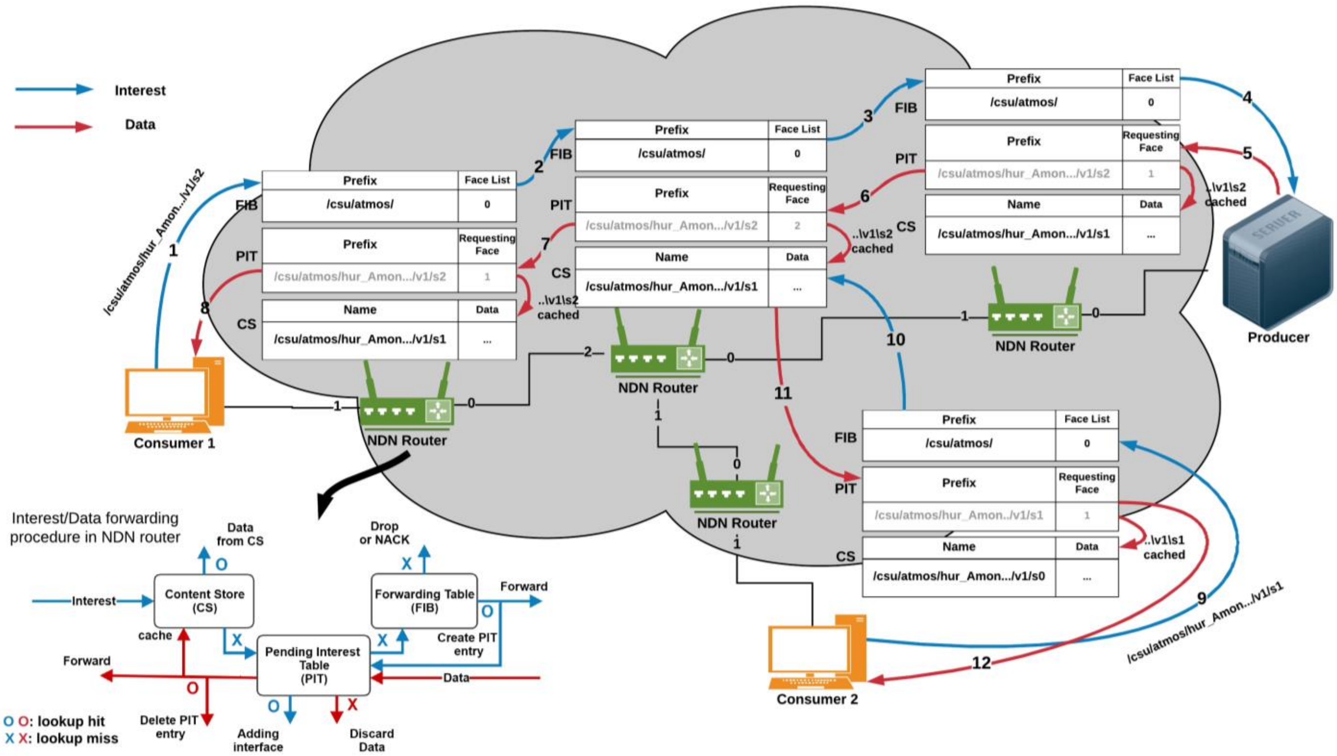


Figure 1: Data fetching scenario using Interest and Data symmetrical forwarding in NDN network.

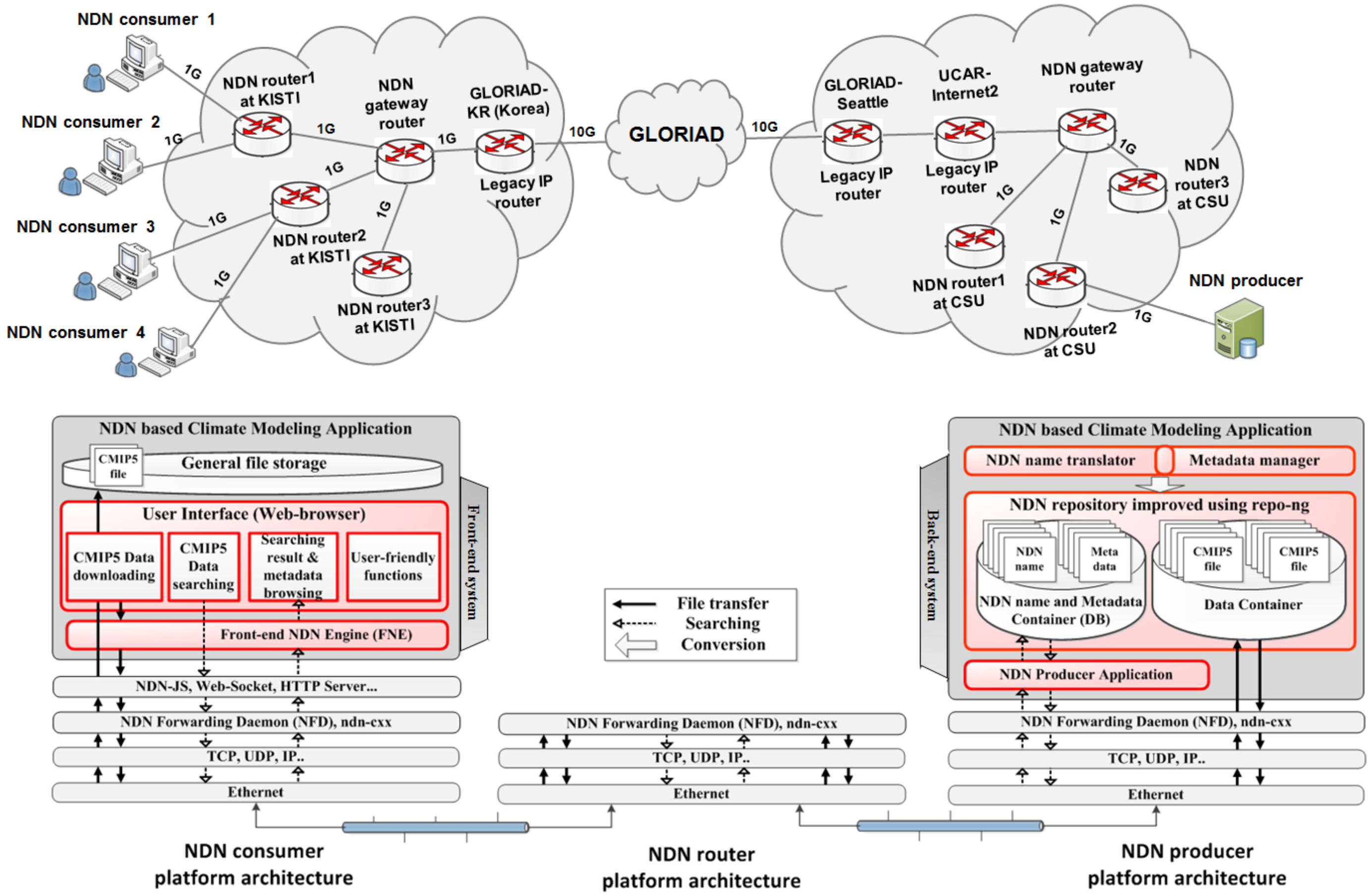
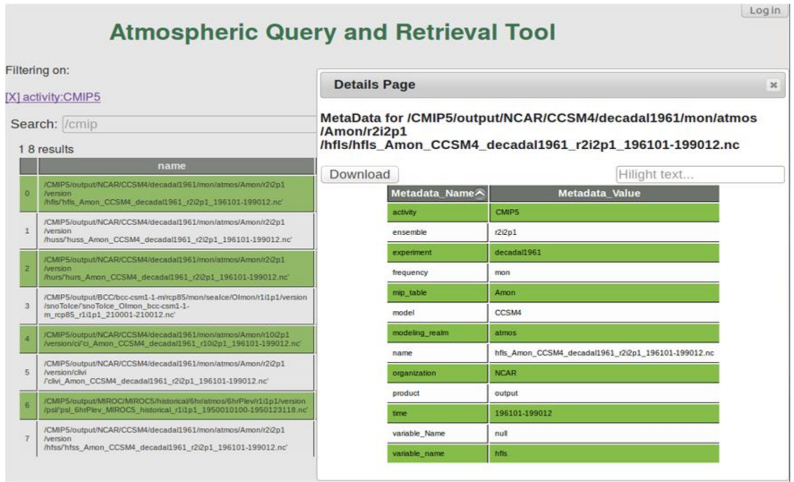
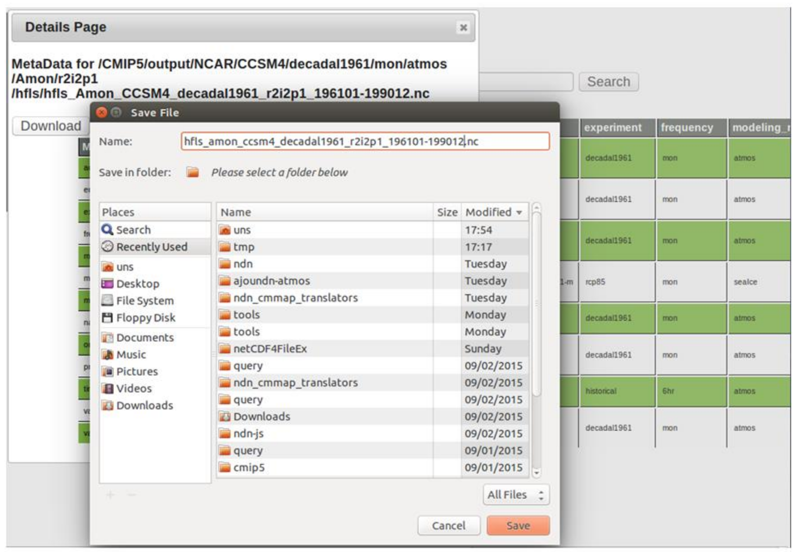


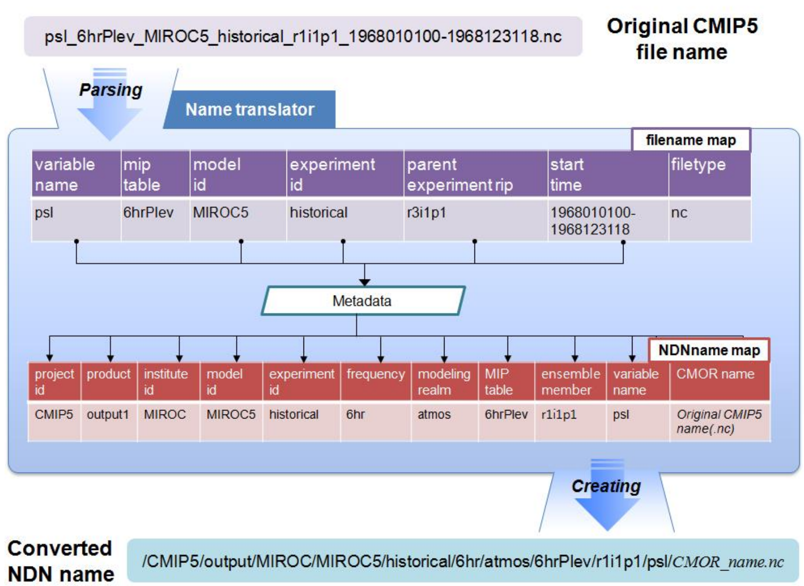
Figure 2: NDN testbed established between continents for climate modeling application.



(a)



(b)



(c)

Figure 3: User interface and name translation procedure in NDN-based climate science application: (a) user interface for data searching and metadata browsing for consumer, (b) user interface for data fetching for consumer, and (c) NDN name translation procedure.

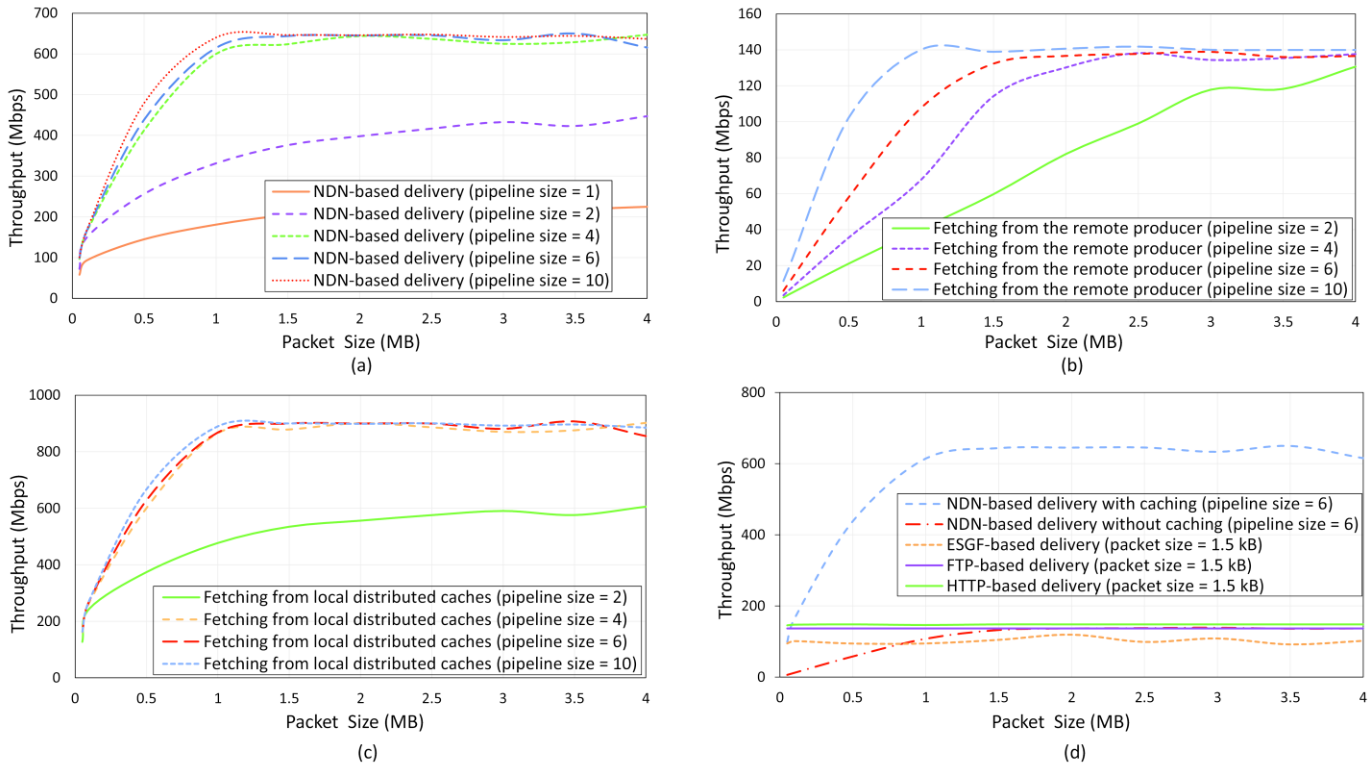


Figure 4: Throughput measured as a function of packet size (in case of no TCP tuning): (a) for NDN-based delivery with different pipeline sizes, (b) for fetching from remote producer with different pipeline sizes, (c) for fetching from local distributed caches with different pipeline sizes, and (d) for NDN-based delivery and HTTP/FTP/ESGF-based delivery techniques.

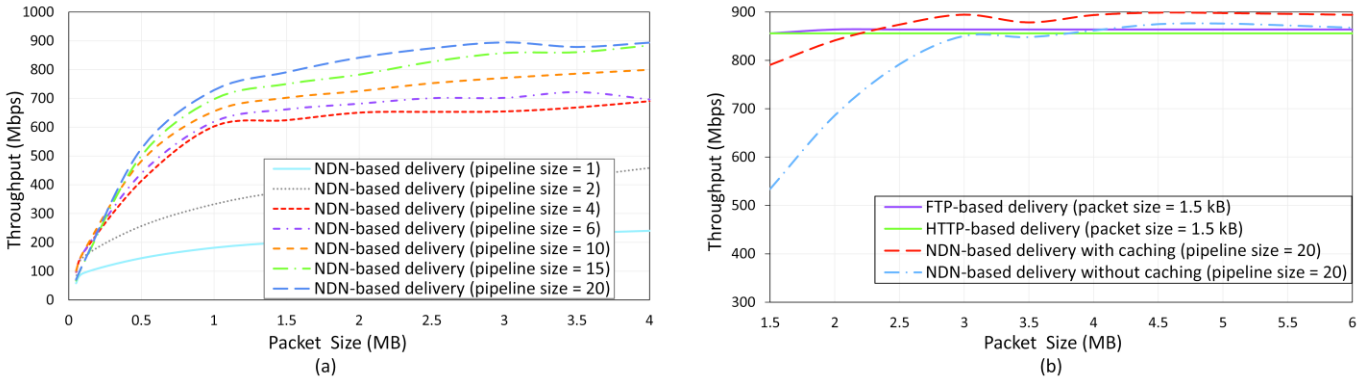


Figure 5: Throughput measured as a function of packet size (in case of TCP tuning): (a) for NDN-based delivery with different pipeline sizes and (b) for NDN-based delivery and HTTP/FTP-based deliveries.

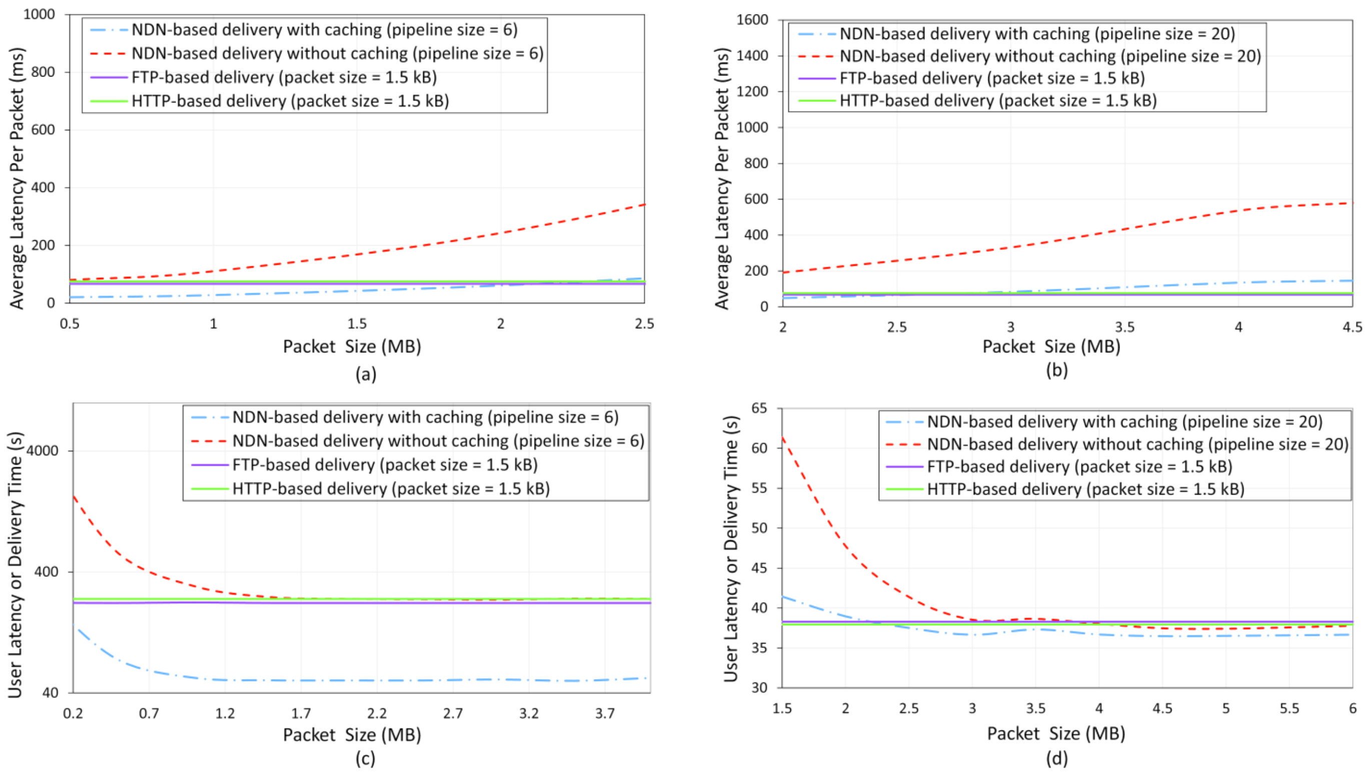


Figure 6: Latency measured as a function of packet size for NDN-based delivery and classical delivery techniques: (a) average latency per packet for no TCP tuning case, (b) average latency per packet for TCP tuning case, (c) average user latency (delivery time) for no TCP tuning case, and (d) average user latency (delivery time) for TCP tuning case.

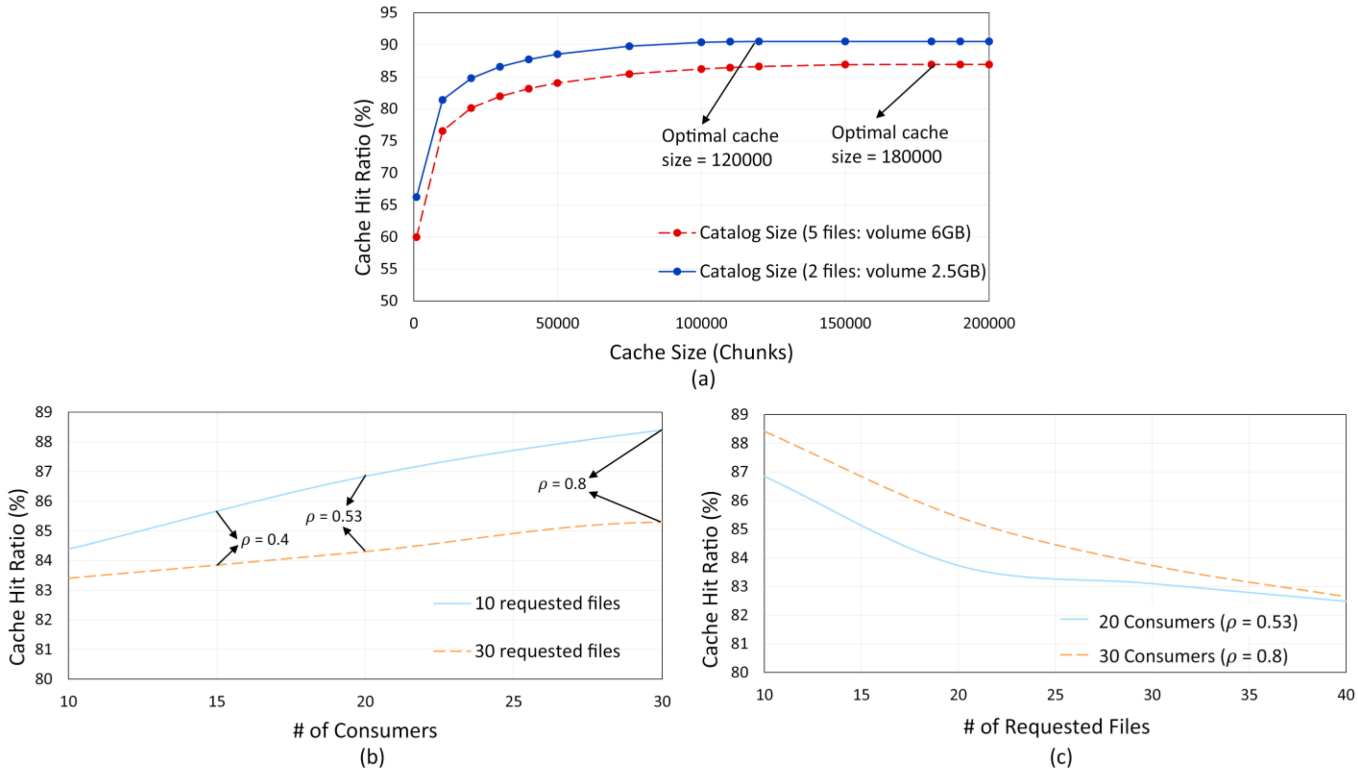


Figure 7: Cache hit ratio (α = 1.15): (a) as a function of cache size for different numbers of files (catalog size) at producer [traffic load (ρ) = 0.575], (b) as a function of number of consumers for different numbers of requested files, and (c) as a function of number of requested files for different numbers of consumers.