Evaluating Named Data Networking For Large Scientific Data

Susmit Shannigrahi
Colorado State University
susmit@cs.colostate.edu

1. INTRODUCTION

Host to host communication protocol in IP is a mismatch for the current Internet which focuses on content delivery. This mismatch makes data management and retrieval complex - often requiring complicated applications or middleware. With increasingly large amount of scientific data being generated, management and retrieval challenges are getting more difficult.

Named Data Networking (NDN)[2] is a potential next generation Internet architecture. Unlike IP, NDN routes directly on content names, not by the location of the content. All data is signed and publicly verifiable, therefore can be retrieved from any entity holding it. Content is also cached along the delivery path, therefore making them reusable for subsequent requests. In addition, NDN comes with a flexible strategy layer which enables network level strategies without need for complex middleware or applications.

In this preliminary study, we evaluate NDN for large scientific data and show that NDN can potentially reduce data transfer and management complexities.

2. NDN FOR LARGE SCIENTIFIC DATA

Scientific dataset is often very large and requires cataloging for tracking - an expensive, labor intensive and application specific task. Moreover, transporting such large datasets can put stress even on high-bandwidth links. Data is also often distributed, requiring complex middleware [1] for locating and retrieving them.

The central idea of NDN is that content is addressed and routed by its name. For achieving data transport, NDN uses two packet types, Interest and Data packet. Interest and Data packets can be considered as request and reply, respectively. Each interest packet contains a hierarchical, human readable name. For detailed description of NDN, readers can refer to "Networking Named Content" by Jacobson et. al.[2].

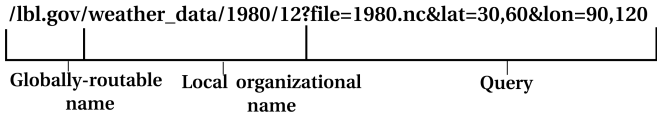

Figure 1: An interest name in NDN - naming convention is flexible, we used the following - the first portion of the name is globally routable name, second portion is local organizational name and the last portion is the actual query.

NDN has few useful properties for large scientific data:

- Availability: In NDN, content is not bound to location. If content source is unavailable, content can be retrieved from other entities (e.g., routers) holding it. All content is signed hence can be verified upon retrieval.
- Source location: Currently, locating distributed data requires application and dataset specific cataloging. Using its name-based routing table, NDN can locate and retrieve content without needing a catalog.
- Caching: NDN caches content at each node(e.g., routers) along the return path. When data is large and can not be cached entirely, partial data is fetched from intermediate caches and the rest from the source.
- Remote query for subsetting data: For scientific analysis, a smaller portion of a large dataset is often required. Fetching only the required subset can save transport time. NDN can easily encode a query in the interest name. The source can execute the query and send the data back. This reduces the transport time and necessity for complex applications.
- Scalable retrieval: Unlike IP, NDN can take advantage of multiple simultaneous connections. A dataset replicated over multiple sites can be retrieved in parallel using the strategy layer. When data source is the bottleneck (e.g., disk I/O), parallelizing retrieval reduces transport time.
- Flexibility: Using the strategy layer, NDN can support various application requirements, such as parallelism or dynamic load balancing. Currently, this requires complex middleware or manual involvement.

Table 1, compares NDN with current application and middleware solutions. While some of these applications are capable of supporting some aspects of scientific workflows, none of them are as versatile as NDN. Moreover, adapting NDN for workflows require much less effort than existing solutions.

3. EXPERIMENTS AND RESULTS

For our experiments, we used a testbed using nodes at ANL, BNL, LBNL. The nodes were of same specifications and connected using 10G links on ESNet. We used a dataset having 26 years worth of water vapor data (33GB) with one months data in one file. For showing NDN's ability of scalable retrieval when source is the bottleneck, we used PlanetLab. The data was published under a hierarchical naming

	NDN	CDX	DataCutte	EVPath	GridFTP	OpenDAP
Transparent data access	Y	N	N	N	N	N
Application independent strategies	Y	N	N	N	N	N
Multi-source retrieval	Y	N	N	Y	Y	N
Remote subsetting on multiple sources	Y	N	Y	Y	N	N
Caching	Y	N	N	N	Y	N
Name supported embedded query	Y	N	N	N	N	Y
Where Implemented	Network	Application	Middleware	Middleware	Application	Application
Approximate LOC	595	2.2K	17.8K	37.2K	66.9K	70.6K

Table 1: Comparison of NDN with current methods

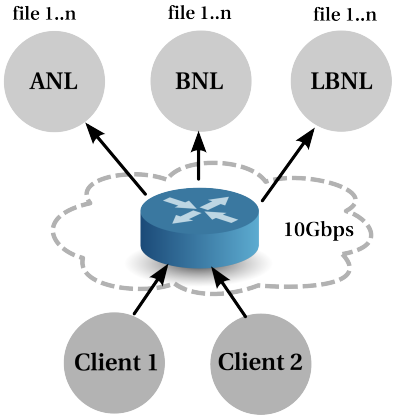

Figure 2: Testbed Topology

scheme - $/TMQ/cam5/year-month.nc$. We extracted water vapor data from the files for a given range of latitude and longitude.

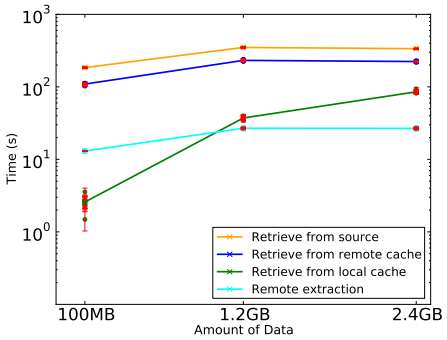

Figure 3: Retrieval time in different scenarios

First, we fetched the full dataset from source followed by retrieval from intermediate cache and local cache. We then extracted the data locally. NDN caches the data along the return path. Therefore, the retrieval time decreases. Even with one intermediate cache, retrieval time can be reduced. We repeated this experiment using different amount of data - one months, one years and two years. We also embedded a query in the interest, extracted the data at the source and retrieved it.

Considering retrieval from source as the baseline, remote computation reduced the retrieval time by over 90%. A single intermediate cache was able reduced the retrieval time by 33-40% depending on the size of the data. When the data was cached locally, the retrieval time was reduced by about 75% for large data and by about 98% for one month's data. Therefore, all three methods offered considerable performance improvements over what is used today. However, unless data is small, remote computation is the fastest of all four methods. This experiment demonstrates NDN's usefulness for locating data by name, ensuring data availability, caching and supporting remote query for subsetting large data and potential performance improvement.

To demonstrate scalability, we replicated the data over multiple sources and retrieved the dataset in parallel. NDN's strategy layer achieved the parallelism without requiring any middleware. The retrieval speed-up was almost linear with the number of sources. This shows that NDN is able to provide scalability at the network level. The strategy layer is flexible and appropriate strategy can be installed depending on the user's needs.

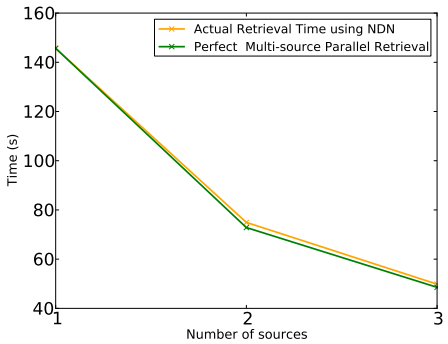

Figure 4: Scalable Retrieval using NDN

4. CONCLUSION

With its name based routing and caching, NDN simplifies workflows involving large scientific data. NDN also offers several interesting properties - name based routing reduces the need for catalogs, interest packets can encode information for performing remote tasks. Moreover, the strategy layer enables intelligent and flexible data management and retrieval without the need for complex applications or middleware.

5. REFERENCES

[1] CERN. Cms computing: technical design report. 2005.

[2] Van Jacobson et.al. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009.