A Top-k Concast Service for Multiple Tiny Data Retrieval in NDN

Zhuhua Liao,Jian Zhang, Zengde Teng, Yizhi Liu, Aiping Yi
College of Computing Science and Engineering, Hunan University of Science and Technology, Xiangtan, China
Key Laboratory of Knowledge Processing and Networked Manufacturing, University of Hunan Province, Xiangtan, China
Email:zhliao@hnust.edu.cn, zjhunst@gmail.com, tengzengde@163.com

***Abstract*—Offering a paradigm for retrieving and aggregating multiple data from multiple sources is a crucial requirement in a large content-centric network. However, the major hindrances to this paradigm are network's dynamic feature, traffic balance, wired forwarding and the absence of cooperation between communications and computations. In this paper, we present a scalable and top-k Concast service on Named Data Networking (NDN). The service enables cooperation between top-k tiny data discovering and aggregating among multiple routers and paths for a user's Interest that contained a hierarchical name and other constraints. Specifically, multiple types and strategies of tiny data aggregation for merging and processing the positive data and suppressing the negative, futile data, as well as a determination of response completeness are introduced for enhancing relevant results recall and sharing. The experimentation demonstrated the top-k Concast service can effectively improve the service quality, reduce network traffic and shorten response time.**

I. Introduction

Distributed data storages and content-centric networking technologies connect big distributed data from everywhere. And more and more consumers fancy collecting and aggregating (or combining) the distributed data from the Internet. But most of positive data are further needed processing, while all of negative data are needed to be suppressed for reducing bandwidth consumption. However, in distributed networks, these issues are often separately handled by third-party platforms or applications, since the IP network only focuses on transmitting individual data via addresses. Today, many applications have been proposed to address the data discovery and aggregation on P2P [1, 2], WSN [3] and V2X [4] over the Internet. However, these applications are confined to the host-based communication model and the separation of data addresses and semantics, so the data discovery and aggregation problems are still not addressed well in distributed networks. While transmitting large amounts of individual data or data segments over a network, they often cause duly response time, heavy network traffic and lower service quality.

Recently, a novel data-centric network paradigm, called Named Data Networking (NDN) [5], offers flexible access to huge collections of hierarchical categories and objects, since Interests can be purposefully routed to related parts of (or multiple sources in) the network through their respective hierarchical names. For returning the results, the outgoing Interests will be recorded in the PIT (Pending Interest Table) of each passing router. In a NDN router, a received Data Packet is identified by its hierarchical name and gets returned faces by Exact Match (EM) between its name and the entries in PIT, that is, the name of the data that will be returned has to be the same as the name of a related PIT entry. A router may receive multiple different response data from different faces, but only one of them will be returned to downstreams. So, one Interest only fetches one Data from one source at most, and which Data will be fetched is up to the response time of the data. Generally, NDN uses the FCFS (First Come First Service) policy. In addition, introduced by the in-network caching mechanism, a large number of cached data is also a kind of wealth for sharing and speeding up the response time and reducing network traffic. So, potentially NDN could become an efficient network architecture for distributed and dynamic network services. However, NDN does not focus on a group of distributed data response at a time.

Based on the principles of NDN, the paper focuses on a new distributed cooperation mechanism which combines computations and concast (the reverse of multicast) [6] communications to concurrently discover and aggregate multiple tiny data from multiple sources, we called a top-k Concast service. With the top-k Concast service, each powerful router or processing nodes can aggregate multiple answers that coming from different sources or paths based on limited cache. And then the aggregated data are returned fleetly to downstreams by the hints of PIT. The Concast service also aims to support a growing number of applications involving data discovery and aggregation in network layer by reinforcing NDN.

II. RELATED WORKS

Lots of applications or components, such as Database, WWW and WSN sensors, often produce a mass of tiny data on the Internet. The concast communication patterns arise in a variety of application domains, including reliable multicast, where receivers must all transmit ACK/NACK messages to the senders [6, 5], concast service, where send multiple tiny data from different receivers to one sender [7, 6] on the IP networks. On the other sides, there are other kinds of services proposed for data collecting and aggregation in the data-oriented networks. The Collectcast [8] is a peer-to-peer (P2P) service for media streaming where a receiver peer is served by multiple sender peers. The in-network data aggregation [9] is a global process of gathering and routing data through a multi-hop network, processing data (mainly performing MAX, MIN, AVG operations) at intermediate nodes with the objective of

978-1-5386-4870-4/18/$31.00 ©2018 IEEE

reducing data size or compressing data. The ActiveCast project [10] introduced the concast for achieving some other benefits, such as implosion avoidance, improving throughput, besides bandwidth reduction. But these concast communication do not focus on data discovery and semantic aggregation (or combination), and should rely on IP addresses.

Nowadays, many kinds of services or applications have been developed on NDN besides data transmission, such as data discovery [11–13], data collection [7], Publish/Subscribe [14] by extending NDN. Of which, most discovery algorithms in NDN consider transmitting Interests to exchange available information of local caches among neighbors. Although all the discovered data will concast to the sender, they do not require aggregation because short distance and absence of common path. For another, the Fuzzy Interest Forwarding [15] in NDN exploits semantic similarities between the names of Interests and the names of cached data. Considering the congestion signals and bandwidth limits, the B-ICP method [16] presents a forwarding solution to retrieve content via multiple paths concurrently. But these improvements of NDN still focuses on retrieving individual data for an Interest. In a word, the NDN resembles Web semantics at packet granularity and provides pervasive storage and request / response data exchange [17]. Nevertheless, for now, there is no mechanism to support multiple data discovering and aggregating from sources in NDN.

For selecting out multiple needed data from a distributed network, distributed top-k collection or query have received considerable attention. For example, The SPEERTO [18] broadcasts summary information of the k-skyband, and utilizes a threshold-based super-peer selection mechanism for efficient top-k query processing in distributed environments. [19] proposes an alternative way, named BRANCA, which combines semantic/branch caching with routing indexes to significantly reduce query cost. [20] proposes a PT-Topk query processing which can reduce data transmissions by only returning the tuples that fallen in sufficient set. [2] proposes a top-k query processing framework in unstructured P2P networks for reducing network traffic, which considered data's scores ranking and score-list merging and backward. But these schemes do not have the advantages that NDN brought about, such as the ability of hierarchical name-based matching, data multicast on network layer and optimal routing [21].

In brief, current works are short of semantic data discovery and aggregation simultaneously on dynamic networks for exploiting or retrieving multiple distributed data with timely response and a limited size of response traffic.

III. The Top-k CONCAST service FRAMEWORK

Our Concast service is based on the NDN, which presents a novel communication infrastructure because of some distinct communication modes for request / data transmission.

Definition 1: **Semantic multicast**. In NDN, one Interest will be semantically forwarded to all related routers or sources through the out-faces that obtained from FIB by matching between the hierarchical name of an Interest and the prefixes in FIBs in every passed router.

Definition 2: **Smart Unicast**. As a whole, NDN can implement the best data retrieval and does not need to specify the detailed addresses of data sources (or destinations) but the hierarchical name or the prefix of the needed data.

In NDN, a packet should carry a hierarchical name, which is similar to the URI form, to enable demultiplexing and provide structured context. But only employing the LPM (Longest Prefix Matching) algorithm will hardly retrieve multiple related data from different sources. Therefore a new formulation is required for distributed data discovery and aggregation. Inspired by this, we present a novel concast service by extending NDN.

Definition 3: **Concast Service** is a service that layered gathering distributed response data to a sender in the reverse of request multicast paths. And in each passed router the response data aggregation will be performed for improving the service quality, reducing network traffic and shortening response time.

The Concast service in NDN means to synthesize multiple tiny data and pack them into one or more Data Packets. For improving the service performance, there are some constraints should be considered, for instance, Interest-Data flow balance, response time and results completeness.

For declaring an Interest for data discovery and aggregation, the Interest should carry more necessary information than that of NDN. Here we define a new format of the expression of an Interest below:

"$Hierarchical\ name\ +\ Constraints\ +\ OPs$"

Here, the *Constraints* is an expression for discovering relevant data in sources or routers, the OPs are the names of operations or functions for data aggregation in some routers. Here is an instance of an Interest for a Concast service: "Computer network / routing protocol /+ Const(filetype = pdf, date>2016) /+ Agg(10,0.8), Proc(top-k) /+ ReStra(1)". In the instance, the Agg(10,0.8) means aggregating 10 data at most and their max score is 0.8, the Proc(top-k) means performing top-k ranking and the ReStra(1) means returning one packet which contained the top-k data for an Interest.

In the Interest multicast phase, the LPM algorithm is used to find next hops for an Interest. So the Interest can be forwarded to related routers or sources.

In the data concast phase, if the number of returned data responsive to an Interest is very larger, these data will be ranked by a ranking function. And the aggregation operation will filter out the data that are not "the most needed". So, a router will return a fixed-size dataset with no matter how many data are came from different upstreams for an Interest.

A. Multicast forwarding

In our scheme, an Interest packet, shown in Figure 1(a) includes Name, Selector and Signature fields, that is similar to NDN. To discover data in NDN, the logic and arithmetic operations and wildcard characters are used in an Interest. Whereupon, the *Name* field is extended by attribute -value pairs besides a name or prefix. The Aggregation bits include the maximal number of data, maximal score, and response

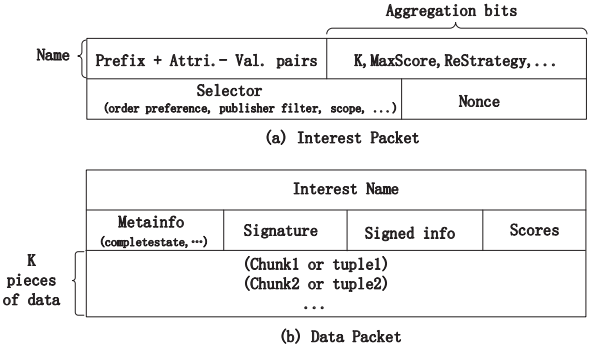

Fig. 1. Concast packet types

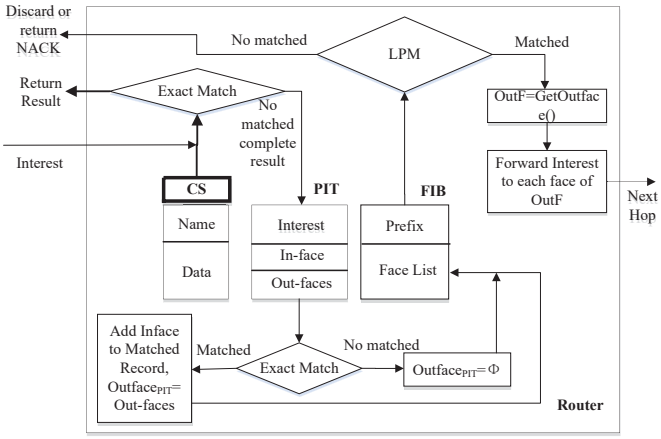

Fig. 2. Multicast forwarding of Interest

strategy. In a big network, it needs more time and traffic to return all relevant data to a router from upstreams. So a response strategy is used to pick up the (or approximate) top-k results. Other preferences (i.e maximum hops and outfaces) are added in *Selector*.

In the multicast forwarding phase, shown in Figure 2, if there are no matched results with complete status in the *ContentStore* (CS), the Interest will forward to other neighbors. And the next hops of the Interest is determined by the Algorithm 1. In the algorithm, the new coming Interest do not need to forward to the In-faces and Out-faces (if no results returned from these faces or the results still cached in CS) that recorded by the same pending Interest again.

Algorithm 1 GetOutface($Interest\ I_1, FIB, PIT$)

Input: I_1, FIB, PIT;
Output: Outfaces set $OutF$;

1: **if** There is matched Out-faces between I_1 and PIT **then**
2: $Outface_{PIT} =$ Out-faces
3: **else**
4: $Outface_{PIT} = \phi$
5: **end if**
6: Do LPM match between I_1 and FIB.
7: **if** There is matched faces **then**
8: $Outface_{FIB} =$ matched faces
9: **end if**
10: Computing $OutF = Outface_{FIB} - Outface_{PIT} - Inface_{PIT}$
11: **if** $OutF$ is not NULL **then**
12: Forwarding I_1 to the first $Maxfaces$ faces of the $OutF$ set
13: **end if**

In addition, there are other constraints in Interest multicast for speeding up the concast performance, for example, setting the $MaxOutFace$ or $MaxHop$ in the *Selector* of an Interest Packet for limiting the forwarding scope and distance.

B. Concast processing for responses

The Data Packet in NDN is a pipeline connecting consumers and sources. For delivering multiple tiny data or tuples in the Concast service, the Data Packet, shown in Figure 1(b), is composed of 6 fields: Name, MetaInfo, Signature method, Signed info, Scores and Data. In the *MetaInfo*, a new *Completestate* sub-field is added. The Completestate shows whether the returned data are complete to upstreams. The Scores field carries each score (an array) of k pieces of data that carried by the Data packet. The scores will be used to rank for selecting out top-k data when a router received multiple Data packet with the same name. In a Data Packet, the structure of data field consists of two components: (PID, TL). The *PID* is the source or publisher ID(s), and the *TL* is a dataset. The value of a score initialized by a score function at a source shows the match degree of a data. A data more matched the Interest, higher score it will be. By these scores, a router can get the most needed data with a limited size through ranking.

In the Concast service, when a router received a Data Packet, the top-k data aggregation will be performed. And then the Completestate of the data packet may be set for finishing aggregation, and better for regional top-k data sharing at intermediate nodes. Finally, it uses the EM to find the response paths according to the name of the Interest recorded in the PIT. One Interest Packet should bring back no more than a certain size of data on each link. Considering the response time, some later returning Data Packets related to the same Interest may be suppressed rather than aggregated. Ultimately, all or at least the most needed data will be returned. In the concast phase shown in Figure 3, the response faces of a Data Packet in a router are determined by EM between the name of the Data Packet and the entries of PIT, and then the Data Packet will be forwarded to the matched faces (or downstreams).

C. Completeness of response

Unlike in NDN, an Interest that recorded in PIT needs to keep for a while after receiving the first Data packet that related to the Interest for receiving complete response data from multiple sources. And we suppose each response

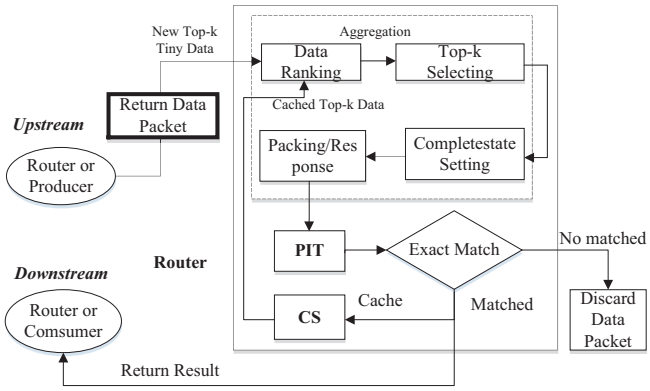

Fig. 3. Concast for results response

data from each source is complete. In a route, the value of Completestate is set by the following formulation:

$$Completestate = \begin{cases} true, \text{ Received complete answers about} \\ \quad \text{the Interest from all out-faces;} \\ false, \text{ Otherwise .} \end{cases} \tag{1}$$

So a router can decide if the Interest record or its outface(s) can be deleted from the PIT according to the value of Completestate. When the Completestate of an answer coming from an out-face is true, the out-face will be deleted, otherwise, it will be kept. When the Completestate of an answer coming from the last kept out-face is true, the Interest entry will be deleted from the PIT. And to the late same Interests, if the results are still cached in CS, they can be immediately returned and the Interests are not required to forward to upstreams. If a source has no response data, the source should return an answer with NULL (the data field is empty) to its downstreams. The *PID* is used to discern whether the answers are came from different sources and whether the answers are new or duplicate from an out-face.

IV. Data Aggregation in Concast

In the concast phase, to efficiently aggregate the distributed and local response data in NDN, the minor aggregations (including data combining and processing, so we called it Agg&C) are introduced. Totally, with minor and asynchronous aggregations in response paths, we can empower NDN with the ability to discover and aggregate the distributed data, but no violating the network principles, or seriously, such as traffic balance and wired forwarding. And the monolithic aggregation that usually assigned to network edge near the consumer before will be decomposed. For saving the cache resource, the aggregation operation in the Concast service doesn't need to cache multiple or all response data and can bring forward the aggregation operation as long as a new coming Data packet arrived. Each data's score can be reused to rank and limit the answers' size in data aggregation. So, after an aggregation operation is finished, each router only needs the limited cache to store the last aggregated answers.

For the sake of various data aggregation, there are four types of Agg&C (see Figure 4), of which only the Overlapped and Disjoint Agg&C will aggregate the newer data and the cached data. The other two types of Agg&C are simple. they just replace the cached data by the new coming data or do nothing. Once a set of data with scores arrived, they will be quickly aggregated by a type of Agg&C.

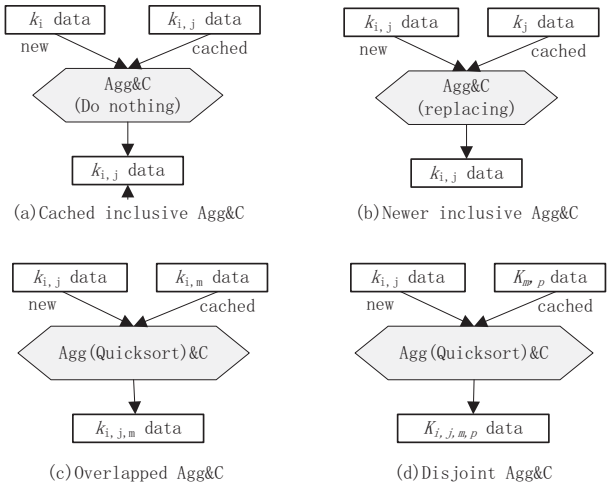

Fig. 4. MINOR AGGREGATION AND COMPUTATION TYPES

In the $Overlapped$ or $Disjoint$ Agg&C, for fleetly aggregating answers, the $Quicksort$ algorithm is introduced to sort all new coming and cached data that pertained to an Interest. And then the new k data are selected out from them and will replace the cached data. Certainly, the scores of all received data can be adjusted or recomputed in a router when considering the traffic, response time, forwarding distance and other factors. Besides, in our system, to simplify the Agg&C operation, we adapt same criterion to calculate the scores at each router for an Interest.

In a router, an Agg&C operation only involves two set of *k* data (chunks or tuples). The Agg&C algorithm is showed in Algorithm 2 for accomplishing the $Overlapped$ or $Disjoint$ Agg&C. And the computational complexity of the Agg&C algorithm is $O(2k * log_2 2k) + O(k) + O(Preprocess)$, which is unrelated to the number of response.

After an Agg&C operation gets executed, when and how often to return the answers determines the traffic amount of NDN and the response time of an Interest. Here, we introduce two strategies for answers response: (1) only return the final answer after Agg&C (called *MAR-1*); (2) return an answer after *m* times of Agg&C (called *MAR-2*).

To the *MAR-1*, an Interest will only receive a Data packet from NDN. When every outface received a Data packet whose Completestate is true, the packet for the last aggregated results will be returned and its Completestate will be set true. Supposing the response time less than the minimum time

Algorithm 2 Two_k_Agg&C($Head\&new\_head, Data\&new\_content, int\ k$)

```
// preprocessing the new coming data
Preprocess(new_head, new_content);
Storedata = Lookup_CS(new_head.getName());
if Storedata == nullptr then
    //if no cached data, the new data will be inserted
    Storedata.head = new_head;
    Storedata.content = new_content;
    Insert_to_CS(Storedata);
else
    //quick-sort new coming data and cached data
    Top-k_struct fres, frescs, topkres;
    //copy top-k chunks or tuples of new_data to fres
    Memcpy(&fres.content, new_content, new_content.size());
    Memcpy(&fres.head, new_head, new_head.size());
    //ranking new_content with their scores
    QuickSort(&fres, 0, k − 1);
    //copy top-k chunks or tuples of storedata to frescs
    Memcpy(&frescs, storedata, storedata.size());
    //select out new top-k chunks or tuples
    topkres= Pick_Top-k(fres, frescs, k);
    storedata.content = topkres;
    //cache the aggregated data
    Replace_CS(storedata);
end if
```

of caching, the maximum size of response data $S_{MAR-1} = |top - k\ data|$. Supposing T_{lf} is the longest forwarding time of an Interest, for example, the Interest is forwarded to the farthest source, and $T_{Agg\&C}$ is the time of distributed data aggregation in the reverse of the forwarding path, the response time of *MAR-1* $T_{MAR-1} \approx 2T_{lf} + T_{Agg\&C}$. The *MAR-2* can shorten the response time and reduce the total number of response data. If returning the output after *m* times of Agg&C operations, the maximum size of response data to its consumer $S_{MAR-2} = |k\ data| \approx S_{MAR-1}$. Supposing T_{NDN} is the response time of an NDN Interest, the response time of *MAR-2* is $T_{NDN} \leq T_{MAR-2} \leq T_{MAR-1}$.

So, for the best recall rate of retrieval, a user can choose the *MAR-1*, otherwise, he/she can choose the *MAR-2* for quickly retrieving *k* pieces of data. Certainly, if the network traffic allows, a user also can set *MAR-2* to get the response data after every *m* times of Agg&C operations for balancing the response time and network traffic.

V. Implementations

Based on the top-k Concast services framework, we implement a hierarchical name-based multiple tiny data retrieval application on ndnSIM (a popular NDN simulator based on ns-3) [22], which can realize top-k data discovering and aggregating for large distributed data on the distributed network. In our experimentation, we use the DMOZ data (http://dmoztools.net/), an Open Directory Project that categorizes large Web sites.

TABLE I
THE FEATURES OF A DMOZ CATEGORY

Type	**Information**	
Hierarchical categories	Tuples (websites)	Size(B)
Kids_and_Teens \	50584	11741530
Kids_and_Teens \ Arts \	1229	339603
Kids_and_Teens \ Arts \ Online Stories \	311	49651
Kids_and_Teens \ Arts \ Online Stories \ Animals and Insects \	41	7322

The attributes of the data includes title, topic, description and URL(Web site address). The topic describes the hierarchical categories that content semantics of every website fallen into, for example: The overall features of the category "Kids_and_Teens" are showed in Table I. In this case, each hierarchical class included numerous websites. We partition these websites data into several groups and each group of the websites data stores on different sources.

In this section, we evaluate the performance of our scheme the Concast service and the *Smart Unicast* (using specific interest or plus some constraints to retrieve an specific answer from a producer in NDN) in a 11*11 grid network. And we set the network parameters of NDN as below:

```
PointToPointNetDevice::DataRate = 1Mbps
PointToPointChannel::Delay = 10 ms
DropTailQueue::MaxPackets = 10
```

The Figure 5 plots the completion time of both the Concast service and the *Smart Unicast* approaches as the producers increase, but fixing the forwarding hops to 9, and set *k* = 20 tuples (each tuple = 80B).Both of the Concast and the *Smart Unicast* use the Q-multicast to deliever an Interest to all related sources. From the Figure 5, we observed the completion time of the Q-multicast is relatively very small. The *Smart Unicast* approach retrieves multiple answers from different producers by multiple times of unicast communication. Because the *Smart Unicast* approach require multiple Interests multicast (Q-multicast) for retrieving multiple answers from different producers, the completion time (except aggregation time) of this approach will proportionally increase as the producers increase. However, even the completion time of the Concast approach includes the aggregation time, it will only increase very slowly as the producers increase. And regardless of which response strategy (MAR-1 or MAR-2), they do not affect the completion time of the Concast approach.

The Figure 6 shows the completion time for the Concast service and the *Smart Unicast* as the hops increase, but fixing the producers to 3 and *k* to 40 tuples. Because the Concast saves more time for Interest forwarding when producers more than one, the completion time of this approach is sublinear increased but the *Smart Unicast* is linear growth. And we noticed that the completion time of the Concast service increased

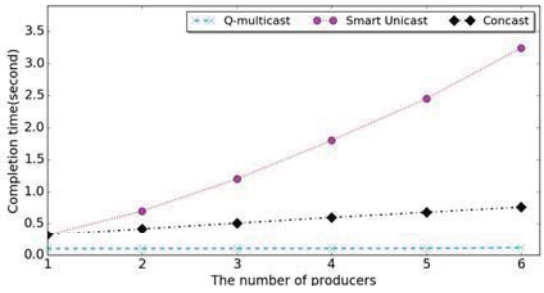

Fig. 5. The completion time with different producers (k=20)

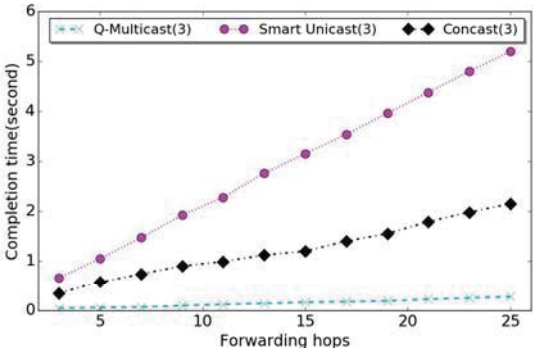

Fig. 6. The completion time with different hops (k=40)

at nearly half of the rate of the comletion time growth of the *Smart Unicast*.

The Figure 7 shows the completion time for the Concast service and the *Smart Unicast* as the *k* increase, but fixing the producers to 3 and hops to 9. We found that the Concast will also save more time than the *Smart Unicast* as the *k* increase from 10 to 90.

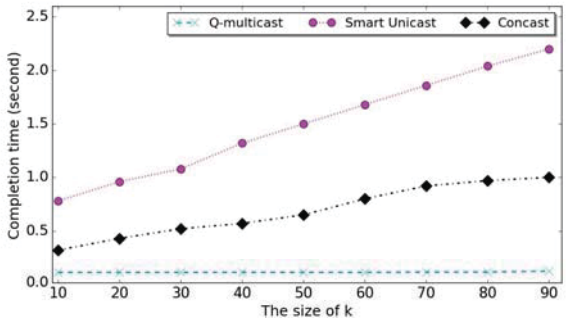

Fig. 7. The completion time with different answer size (9 hops, 3 producers)

The Figure 8 shows the response traffic for the *Smart Unicast*, and the Concast service with *MAR-1* and *MAR-2*. In *MAR-2* response strategy, we set response results after every 2 times of Agg&C on the consumer node. The results illustrate the Concast approaches have less response data than *Smart Unicast* although all these approaches can get top-k answers because the Concast bring the aggregation early on response paths. Especially, the Concast with *MAR-1* has the minimizing volume (the size of top-k answers) of response data than other strategies, which is equal to that of NDN and meets the Interest-Data traffic balance. Moreover, from the Figure 8, we can infer that when the *m* is smaller, the more total response traffic of the Concast service with *MAR-2*. If the $m = 1$, the response traffic of the Concast service with *MAR-2* will close to the response traffic of the *Smart Unicast*. But when the *m* is equal to the number of answers, the response traffic of the Concast service with *MAR-2* will close to that of the Concast service with *MAR-1*.

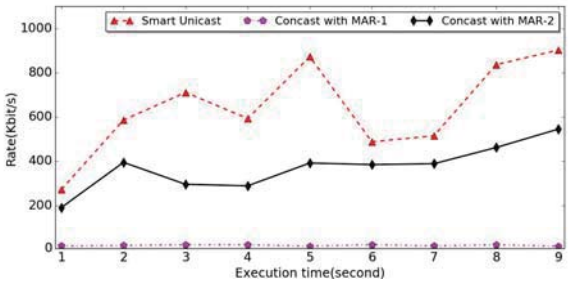

Fig. 8. Packet-level tracing with different response strategies (8 hops, 3 producers, f = 100, k=20)

VI. Conclusion

In the paper, we present a Top-k Concast service framework by extending NDN. In our framework, a cooperation approach for concast communication and top-k data aggregation from multiple sources and paths are presented in a multiple-hop content-centric network. The framework can efficiently discover and aggregate most needed tiny data simultaneously during data routing and transmission, and also shorten response time or largely reduce network traffic or balance between the two in the whole network, as well as only using limited cache space to aggregate multiple tiny data those are from different upstreams in a router. Moreover, it does not violate the principles of NDN and can compatible with the architecture of NDN. Besides, a consumer can customize the forwarding range, preferences and response strategies in their Interests for data discovery and aggregation. This work will potentially extend traditional semantic data discovery, distributed data processing and enable them to have cooperation in each router and in the whole NDN or content-centric Internet.

Acknowledgment

This work was supported by the grant from the Key Project of Hunan Provincial Education Department (17A070), the National Natural Science Foundation of China (Grant No.61370227) and the Hunan Provincial Natural Science Foundation of China (2017JJ2081). We are grateful to Prof. Lan Wang (the University of Memphis) and the anonymous reviewers for their helpful comments.

REFERENCES

[1] A. Abdullah, M. Othman, M. N. Sulaiman, H. Ibrahim, and A. T. Othman, "Data discovery algorithm for scientific data grid environment," *Journal of Parallel & Distributed Computing*, vol. 65, no. 11, pp. 1429–1434, 2005.

[2] R. Akbarinia, E. Pacitti, and P. Valduriez, "Reducing network traffic in unstructured p2p systems using top- k queries," *Distributed & Parallel Databases*, vol. 19, no. 2-3, pp. 67–86, 2006.

[3] J. N. Al-Karaki, R. Ul-Mustafa, and A. E. Kamal, "Data aggregation and routing in wireless sensor networks: Optimal and heuristic algorithms," *Computer Networks*, vol. 53, no. 7, pp. 945–960, 2009.

[4] J. Jiru, L. Bremer, and K. Graffi, "Data aggregation in vanets a generalized framework for channel load adaptive schemes," in *Local Computer Networks*, 2014, pp. 394–397.

[5] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[6] K. L. Calvert, J. Griffioen, B. C. Mullins, A. Sehgal, and S. Wen, "Concast: design and implementation of an active network service," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 3, pp. 426–437, Mar 2001.

[7] W. Drira and F. Filali, "Ndn-q: An ndn query mechanism for efficient v2x data collection," in *2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking Workshops (SECON Workshops)*, June 2014, pp. 13–18.

[8] M. Hefeeda, A. Habib, D. Xu, B. Bhargava, and B. Botev, "Collectcast: A peer-to-peer service for media streaming," *Multimedia Systems*, vol. 11, no. 1, pp. 68–81, 2005.

[9] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 14, no. 2, pp. 70–87, 2007.

[10] M. Bond, K. Calvert, J. Griffioen, B. Mullins, S. Natarajan, L. Poutievski, A. Sehgal, S. Venkatraman, S. Wen, and E. Zegura, "Activecast: toward application-friendly active network services," *Proceedings DARPA Active Networks Conference and Exposition*, pp. 274 – 290, 2002.

[11] C. Anastasiades, A. Sittampalam, and T. Braun, "Content discovery in wireless information-centric networks," in *Local Computer Networks*, 2016, pp. 28–36.

[12] M. Lee, J. Song, K. Cho, S. Pack, T. T. Kwon, J. Kangasharju, and Y. Choi, "Content discovery for information-centric networking," *Computer Networks the International Journal of Computer & Telecommunications Networking*, vol. 83, no. C, pp. 1–14, 2015.

[13] S. M. A. Iqbal and Asaduzzaman, "Adaptive forwarding strategies to reduce redundant interests and data in named data networks," *Journal of Network & Computer Applications*, vol. 106, pp. 33–47, 2018.

[14] J. Chen, M. Arumaithurai, L. Jiao, X. Fu, and K. K. Ramakrishnan, "Copss:an efficient content oriented publish/subscribe system," in *ACM/IEEE Symposium on Architectures for Networking & Communications Systems*, 2011, pp. 99–110.

[15] K. Chan, B. Ko, S. Mastorakis, A. Afanasyev, and L. Zhang, "Fuzzy interest forwarding," in *Asian Internet Engineering Conference*. ACM, 2017, pp. 31–37.

[16] Y. Ye, B. Lee, R. Flynn, N. Murray, and Y. Qiao, "B-icp: Backpressure interest control protocol for multipath communication in ndn," in *IEEE Global Communications Conference (GLOBECOM)*, 2017, pp. 1–6.

[17] K. Shilton, J. A. Burke, K. Claffy, and L. Zhang, "Anticipating policy and social implications of named data networking," *Communications of the ACM*, vol. 59, no. 12, pp. 92–101, 2016.

[18] A. Vlachou, C. Doulkeridis, and M. Vazirgiannis, "On efficient top-k query processing in highly distributed environments," in *ACM SIGMOD International Conference on Management of Data*, 2008, pp. 753–764.

[19] K. Zhao, Y. Tao, and S. Zhou, "Efficient top- processing in large-scaled distributed environments," *Data & Knowledge Engineering*, vol. 63, no. 2, pp. 315–335, 2007.

[20] W. Lee, D. L. Lee, M. Ye, and X. Liu, "Probabilistic top-k query processing in distributed sensor networks," in *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*, vol. 00, 03 2010, pp. 585–588. [Online]. Available: doi.ieeecomputersociety.org/10.1109/ICDE.2010.5447875

[21] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *International Conference on Emerging Networking Experiments & Technologies*, 2009, pp. 1–12.

[22] I. M. Alexander Afanasyev, Spyridon Mastorakis and L. Zhang, "Ns-3 based named data networking (ndn) simulator," http://ndnsim.net/2.3/, 2017.