RAPIT: RTT-Aware Pending Interest Table for Content Centric Networking

Yalei Tan, Qing Li, Yong Jiang, Shutao Xia
Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
tanyalei13@tsinghua.org.cn, {li.qing, jiangy, xiast}@sz.tsinghua.edu.cn

Abstract—**The current Internet is facing a lot of challenges, including the transmission inefficiency due to the popularity of content-delivery applications. Content-Centric Networking (CCN) solves this problem by name-oriented routing and in-network caching. As a core component of the CCN router, the Pending Interest Table (PIT) is crucial for the forwarding performance. In this paper, we propose the RTT-Aware PIT (RAPIT) to improve the PIT efficiency in CCN by reducing the residence time of non-responded PIT entries. First, we provide an approach to measure the Round-Trip Time (RTT) from a CCN router to the content publisher and set PIT entry residence time dynamically based on the measured RTT. Second, we prioritize the PIT entries based on their aggregated request amounts and the possibility of being responded. When the PIT is exhausted, the lowest-priority entry will be replaced by the latest one. The evaluation results show that: 1) compared with current PIT with scheme in CCNx, RAPIT reduces the PIT size to 57.6% and improve the network throughput by 200%; 2) when replacement is triggered as the PIT is full, RAPIT is more accurate and improves the PIT efficiency by 20-50% compared with FIFO.**

I. Introduction

The current Internet, designed 40 years ago, is facing a lot of challenges, including the transmission inefficiency due to the popularity of content-delivery applications (*e.g.*, Youtube, Netflix). Therefore, many brand-new Information Centric Networking (ICN) architectures are proposed, *e.g.*, Data-Oriented Network Architecture (DONA) [12], Network of Information (NetInf) [4], Content Centric Network (CCN) [10] and Publish/Subscribe Internet Routing Paradigm (PSIRP) [13]. As a widely-accepted ICN architecture, CCN solves this problem by name-oriented routing and in-network caching. In CCN, contents are requested and forwarded by names; routers can identify and cache the forwarded contents. In this way, the redundant transmission along the same path can be avoided thus the transmission efficiency is greatly improved.

As a core component of the CCN router, the Pending Interest Table (PIT) is crucial for the forwarding performance. PIT records the content request (Interest). Then the corresponding content packet (Data) traces the recorded Interests back to the consumer. Thus PIT is involved in the forwarding of both request and data packets. As packet arrival rate increases, PIT requires the larger capacity and higher access rate. Although faster memory chips like SRAM [16] can guarantee PIT forwarding performance, they are generally expensive and power-consuming. Moreover, as shown in Fig. 1, the Ethernet link rate increases much faster than memory chip capacity does: the SRAM capacity increases by 144 times from 1987 to 2009, while the link speed increases by 10,000 times in the same period. In 2012, an evaluation on a 20Gbps gateway trace indicates that at such a link rate, the PIT contains 1.5M entries and needs 1.4M lookups, 0.9M inserts and 0.9M deletions per second [8]. This mismatching brings great pressure to CCN routers. **Therefore, it is very important to improve the PIT efficiency while controlling the PIT size.**

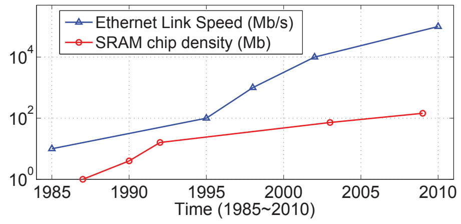

Fig. 1. Ethernet link speed and SRAM chip capacity VS. Time [7], [9], [18]

To provide a solution to the problem, we propose RTT-Aware Pending Interest Table (RAPIT) for CCN. RAPIT improve the PIT efficiency by reducing the residence time of non-responded PIT entries. First, we employ a scheme with the dynamic (instead of fixed) residence time, where the residence time is set according to the estimated return time of the data packet. We provide a lightweight method to measure the return time of the data packet by marking the initial request and the corresponding data packet. Then we store the measured time in the FIB of the router for reuse. Second, to further reduce the residence time of non-responded PIT entry, we propose a strategy for PIT replacement when the PIT is fully filled. In our strategy, the PIT entry with lower responding possibility is prior to be replaced.

The main contributions of RAPIT can be concluded as follows: 1) the dynamic PIT entry residence time setting scheme significantly reduces the PIT memory requirement and improves the network throughput; 2) the PIT replacement strategy recognizes non-responded entries with high accuracy and removes them in advance, further improving the PIT efficiency when the packet arrival rate is high.

The evaluation results on both generated and real-world topologies show that: 1) compared with current residence time setting schemes in CCNx [17] and ndn-ccx [15], RAPIT reduces the PIT size to 87% and 60% respectively to handle the same network load; 2) compared with Random strategy, RAPIT improves the accuracy of removing non-responded entries to more than 99% in PIT replacement; 3) with the

978-1-4673-8590-9/15/$31.00 ©2015 IEEE

same devices, RAPIT improves the network throughput by 20%∼50% and 200%, compared with the $FIFO$ PIT entry replacement strategy and the fixed residence time setting scheme in CCNx, respectively.

II. Background and Motivation

A. Content-Centric Networking

In CCN network, a content can be identified by a human-readable, hierarchical and structural name, which is used in almost all networking processes, including routing, forwarding and content retrieving. There are two types of packets in CCN: Interest and Data. Both of them are identified by the content name. Interest is sent by consumer to request content and Data is sent by content source to reply the Interest with requested content. A CCN router has following core components:

- Content Store (CS): the router caches passing contents in the CS and serves as a dynamic content source to reduce redundant transmission.
- Pending Interest Table (PIT): the PIT keeps tracks of Interest to guide corresponding Data back and aggregates Interests requesting the same content to a single entry.
- Forwarding Information Base (FIB): the FIB contains forwarding information to forward Interests to data sources.

Fig. 2 shows the packet processing procedures in the CCN router. If an Interest cannot be replied to by the CS, the PIT is queried. The PIT creates or updates an entry to record the information of the Interest, then forwards the packet. If the query in the PIT gets missed, the router queries the FIB then forwards the packet. When a Data arrives, the router queries the PIT to find the outgoing interface(s) and forwards the Data, then caches the content carried by the Data in the CS.

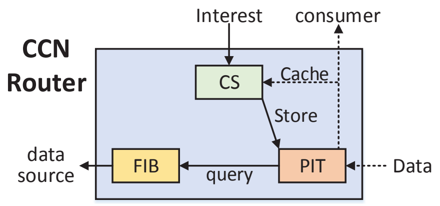

Fig. 2. The packet processing procedures in the CCN router

As described above, the PIT is highly dynamic and capacity-required. However, Fig. 1 has shown the difficulties to provide fast and large enough memory chip for the PIT with current technology. Moreover, network contents and link rates are exploding at high speeds: 400Gbps Ethernet standard will be proposed in 2017 [9], and 79% of the Internet traffic can be video traffic in 2018 [21]. As the PIT plays a crucial role in the aforementioned procedures, it can be the performance bottleneck of the whole CCN router. Thus improving the PIT performance is a critical work in CCN deployment.

B. Motivations

The most common way to improve PIT performance is to optimize its structure, yet in this paper we focus on another alternative way: optimizing PIT entry residence time.

Assume that h_{cs} is the CS hit rate, h_{pit} is the PIT hit rate, λ is the average Interest arrival rate, and τ is the average PIT entry residence time. Then $\mathit{\Gamma}$, the average number of entries in the PIT, can be estimated as follows [2]:

$$\mathit{\Gamma} = (1 - h_{cs})(1 - h_{pit})\lambda\tau \tag{1}$$

We see if $\mathit{\Gamma}$ is fixed, λ is inversely correlated with τ; if λ is fixed, $\mathit{\Gamma}$ is positively correlated with τ. Therefore, reducing τ helps both improve PIT processing efficiency and reduce PIT size. However, current CCN implementations, like CCNx and ndn-cxx, set PIT entry residence times to large fixed values. Furthermore, in PIT replacement, improper strategy may remove PIT entries which can be responded and leave non-responded ones. These cases make non-responded entries stay longer in the PIT, leading to higher τ in Eq. 1 and poorer performance. With this background, we propose RAPIT to provide an alternative optimization to PIT entry residence time.

III. Optimize PIT entry residence time

Based on Section II, to reduce the negative effect of non-responded entries on PIT performance, smaller PIT entry residence time and better PIT replacement strategy are required.

We first give a further discussion to Eq. 1. Assume the average RTT is RTT_{avg}, the packet loss rate in the network is β, the PIT entry residence time is set to T_{res}, the number of different Interests processed during a period of T_{res} at stable status is N. Generally a PIT entry gets responded by the corresponding Data and then deleted, staying in the PIT for RTT_{avg} time; yet if the entry never gets responded, it is finally deleted as a timeout entry, staying in the PIT for T_{res} time. Then during a period of T_{res}, there are βN Interests lost in the network and their PIT entries get timeout. For the rest $\mathit{\Gamma} - \beta N$ PIT entries, each of them processes T_{res}/RTT_{avg} Interests during the period. Thus we have $\beta N + (\mathit{\Gamma} - \beta N)\frac{T_{res}}{RTT_{avg}} = N$, then R, the average processing rate of PIT, can be estimated as follows:

$$R = \frac{N}{T_{res}} = \frac{\mathit{\Gamma}}{(1-\beta)RTT_{avg} + \beta T_{res}} \tag{2}$$

RTT_{avg} is decided by many external factors, yet T_{res} is set by the router. To guarantee enough time for a Data to return in any case, CCNx and ndn-cxx set large fixed T_{res} (4s and 1s, respectively [17, 15]) for all PIT entries on all routers. Actually these T_{res} are "overkilled" for the following disadvantages: 1) on a router closer to the publisher of a content, the residence time of the PIT entry for the content can be shorter than that on a farther router; 2) when network condition gets worse, due to the increasing packet losses, there are more timeout entries, thus PIT entry residence time should be reduced [22]. Therefore, to reduce PIT entry residence time to proper value, it is necessary to set it dynamically based on different cases.

Second, we discuss the cases that may produce improper PIT replacements. If a new Interest arrives but the PIT is fully filled, the router has to perform PIT replacement to delete the Interest or an existing entry. A status of PIT entry should be considered in the replacement: Data response. Fig. 3 shows this status. The status means whether the entry can be responded by the corresponding Data. The Data of entry E_1 is returning and E_1 can be responded soon, while the Data of E_2 is lost thus E_2 will never be responded. A simple strategy ($e.g.$ Random and FIFO) may delete E_1 and leave E_2 in the PIT. In this case, the Data of E_1 cannot be forwarded on the router, but E_2 stays in the PIT until being timeout.

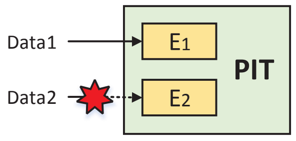

Fig. 3. Status of PIT entries

In summary, RAPIT should use the following schemes to optimize PIT entry residence time: 1) setting PIT entry residence time dynamically; 2) recognizing and deleting non-responded entries. In the following two sections, we will introduce our schemes in detail.

IV. Dynamic residence time setting for PIT entry

This dynamic residence time setting scheme is the first step of our optimization: to reduce the upper bound of PIT entry residence time. We measure the RTT from a router to the content publisher, then set PIT entry residence time according to the measured RTT. The scheme is based on RTT for the following reasons: 1) RTT directly expresses the distance and network condition between routers and content publishers; 2) it is possible to estimate network conditions on the paths ahead by measuring RTT.

A. The storage location of measured RTT

In this scheme, measured RTTs are stored at the router for later utilizations. To build a new table to store the RTTs for different contents, memory chip with large capacity and high access speed are required, thus adding greater pressure to the router scalability. Therefore, it is better to store the RTT in an existing table. Compared with highly dynamic PIT, the FIB, which has stability and high speed, is appropriate to store RTTs. Original CCN FIB stores the prefixes of content names, as the contents with the same prefix may have different RTTs, we use new content naming method and query principles.

The new method names the content with the form of $P : L$. The part P represents the publisher of the content and the part L is a unique label of the content in the network [12]. We assume that a content publisher (not a cache node) has relatively constant location and L is long enough to avoid collision. Note that the same contents provided by different publishers have the same L in their names. The FIB maintains an entry for every different P, the entry stores the forwarding information and RTT from the router to P. Besides, the PIT maintains an entry for every different L.

This naming method has the following advantages: 1) RTTs from a router to content publishers are much more stable than that to cache nodes, thus safer for reuse. 2) Recording publishers instead of full names of contents in the FIB greatly reduces the memory requirement. The increase of single entry size due to additional information ($e.g.$ RTT and update time) can be offset. 3) Storing globally unique labels of names in PIT keeps the Interest aggregation function of original PIT design.

As the publishers sharing the same prefix may have different RTTs, with our naming method, FIB entries cannot be aggregated. However, it is not a fatal flaw. A study on the feasibility of CCN indicates that current technologies support the deployment of CCN at the scale of Content Distribution Network (CDN) [16]. In 2014, ChinaNetCenter, the biggest CDN provider in China, holds about 30000 servers [1]. Assume we deploy CCN in the network of ChinaNetCenter and each of those servers serves as a content publisher, the size of our FIB is still one order of magnitude smaller than that of the BGP Forwarding Table in a core Internet router [5].

B. RTT measurement and PIT entry residence time setting

We set a measuring cycle period T_{msr} for RTT measurement. The router measures RTT for every publisher stored in the FIB periodically. The RTT is measured by marking special Interest-Data pairs.

Assume a router R receives an Interest requesting content from publisher P, and the FIB entry E_{FIB} is queried for the forwarding information of P. When R receives this Interest but finds it has been a period of T_{msr} since last RTT measurement for P, R launches a new RTT measurement. R sets a measuring flag and the router number of R in the Interest, and resets the measuring flag and update time in E_{FIB}. Note that an RTT measuring Interest cannot be used for RTT measurement by other routers, it is always forwarded before reaching P. Cache nodes don't reply to this Interest even if they have the requested content. P replies a Data carrying the same measuring flag and router number as that in the RTT measuring Interest. When the Data reaches R, R queries the FIB and finds E_{FIB}, then updates its RTT and update time, finally cancels its measuring flag.

By observing the RTT measuring flag and the update time in a FIB entry, it is possible to estimate the network condition on the path to the corresponding content publisher. If an Interest or Data gets lost due to bad network condition, the measuring flag in the FIB entry is kept until next successful RTT measurement. If an Interest matches a FIB entry whose RTT measuring flag has been set for more than a period of RTT, the router knows the RTT measuring packets of the entry may have been lost. Generally this indicates bad network condition, yet there are two exceptions: 1) network condition is good but RTT measuring packets are lost; 2) the RTT measuring Data is returning but hasn't reached the router. The possibilities of both the exceptions are low, as good network condition means low packet loss rate, and the RTT of Interest-Data exchange is much more shorter than T_{msr}. Furthermore, our scheme to set PIT entry residence time based on network condition will leave enough redundancy, thus these exceptions can hardly have negative effect on content request.

For the PIT entry of an arrival Interest, the router sets the residence time T_{res} of the entry mainly based on the RTT and the network condition of the path to the content publisher. The detailed rules are listed as follows:

$$T_{res} = \begin{cases} packet\ lifetime,\ if \text{ RTT measuring Interest} \\ 3 \times RTT,\ else\ if \text{ good network condition} \\ 2 \times RTT,\ else \end{cases} \quad (3)$$

Eq. 3 shows one of our basic principles: leave enough redundancy in the residence time to ensure the returns of Datas. Therefore, even on bad network conditions we set the T_{res} to two times of the normal RTT. Furthermore bigger redundancy is left for the PIT entry of RTT measuring Interest to ensure the completion of RTT measurement. Consider RTT_{avg} = 80ms [14], then the residence time upper bound of most PIT entries is 240ms, much lower than that in CCNx and ndn-cxx, thus bringing higher efficiency based on Eq. 2.

Fig. 4 shows the packet processing procedures on a RAPIT router. For an arrival Interest, when the query in CS gets missed, the FIB is queried to get the forwarding interface, the importance (explained in Section V) and T_{res} of the PIT entry for the Interest on this router. Then the PIT is queried. If the query gets missed, a new entry is created with residence time set to T_{res}, then the Interest is forwarded. Otherwise, the router updates the matching entry and decides whether the Interest needs to be forwarded. For a Data, if it carries RTT measuring flag and reaches the router launching the measurement, the router finds the corresponding FIB entry and sets the RTT before processing the Data in the PIT. Otherwise the Data is processed in the PIT directly. In RAPIT, the process of Data in the PIT is the same as that in original CCN.

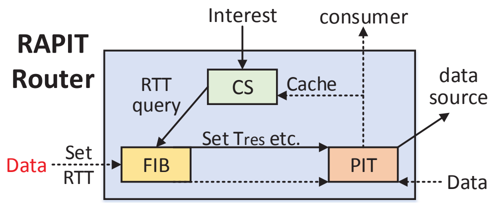

Fig. 4. The packet processing procedures in the RAPIT router

V. PIT REPLACEMENT STRATEGY

A better PIT replacement strategy is the second step of our optimization: to further reduce the time that timeout entry stays in the PIT when the Interest arrival rate is high. As analyzed in Section III, Our PIT replacement strategy should be able to recognize and delete non-responded entries. To achieve this object, we define the importance of PIT entry and Interest, and design a special PIT structure.

A. The importance of PIT entry and Interest

To estimate the importance of PIT entry, firstly we define $Timeout\ judgment\ coefficient$ η to describe the possibility that a PIT entry gets non-responded ($i.e.$, the possibility that the entry gets timeout). η is the ratio of the entry's remaining lifetime to its residence time:

$$\eta = 1 - \frac{t_{current} - t_{update}}{T_{res}} \quad (4)$$

With the precondition that the PIT entry residence time has enough redundancy for Data to return, if η is lower than a threshold δ, the router can judge that it has taken much more time than normal RTT to get the content, thus it is highly possible that the corresponding Data will never come back. Since we generally set PIT entry residence time to three times of normal RTT, we believe $1/3$ is an appropriate value for δ.

In PIT replacement, as analyzed before, the entries with higher η should have higher priority to be kept. Therefore, we directly use η to define the importance of PIT entry, I_{pe}. Notice the precondition to replace an entry is that it is non-responded ($i.e.$, the η of the entry is lower than the timeout judgment threshold δ). The entries with η higher than δ should be kept, thus their I_{pe} are set to 1. Then:

$$I_{pe} = \begin{cases} \eta,\ if\ \eta \le \delta \\ 1,\ else \end{cases} \quad (5)$$

For an Interest, if it is used for RTT measurement, it should have the highest importance as it must be forwarded. Otherwise, the importance of the Interest is estimated with the network condition of the path to the content publisher. If the network condition is good, we set the importance to 1; otherwise the importance is reduced to δ as the Interest or its Data has higher possibility to be lost. Therefore, the Interest's importance I_I is set according to following rules:

$$I_I = \begin{cases} MAX,\ if \text{ RTT measuring Interest} \\ 1,\ else\ if \text{ good network condition} \\ \delta,\ else \end{cases} \quad (6)$$

Except for PIT replacement, the timeout judgment coefficient can be used in the PIT entry updating. If an Interest matches a PIT entry, the η of the entry is computed. if $\eta > \delta$, the router knows the possibility that the entry gets non-responded is very high, thus the Interest should be forwarded after updating the entry. Otherwise, the router just updates the coming interface list of the entry. Meanwhile, if the Interest is used for RTT measurement, it is always forwarded and the entry is updated whatever the η of the entry is.

B. Special PIT structure design

In PIT replacement, the router computes the importance of arrival Interest and PIT entries, then deletes the one with the lowest importance. However, in the worst case the router has to traverse the whole PIT for every Interest. Furthermore, generally most entries are far from being timeout, thus it is unnecessary to check them. To solve this problem, we separate the PIT into two tables: $SmallTable$ holding tens of entries, and $BigTable$ holding the remaining entries.

Fig. 5 shows the structures of $BigTable$ and its entry. $BigTable$ concludes a priority queue and a hash table, the entry of $BigTable$ is a pair of an element E_{pq} in the priority queue and an element E_{ht} in the hash table. Both elements holds the label recorded in the PIT entry: E_{ht} uses it as the key in the hash table, and E_{pq} uses it to locate E_{ht}. E_{ht} holds the main information of the PIT entry ($e.g.$ residence time, update time and interface list). Besides, E_{ht} keeps the index of E_{pq} in the priority queue to locate E_{pq}.

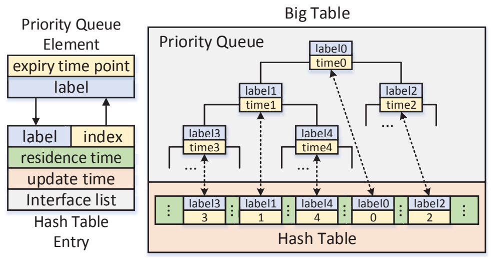

Fig. 5. The structures of Big Table and its entry

The priority queue is implemented with binary minimum heap, and the element is prioritized by the expiry time point of the corresponding $BigTable$ entry, earlier expiry time point means higher priority. Assume the capacity of $BigTable$ is N_{BT}. To insert a new entry, it takes $O(1)$ time to add E_{ht} to the hash table and $O(lgN_{BT})$ time to add E_{pq} to the priority queue. To look up an entry with a particular label, it takes $O(1)$ time to locate E_{ht} with the label. To remove an existing entry, it takes $O(1)$ time to delete E_{ht} and $O(lgN_{BT})$ time to delete E_{pq}. Note that when the index of an E_{pq} varies, the corresponding E_{ht} has to be modified. The operation takes $O(1)$ time as E_{pq} holds the key of E_{ht} in the hash table.

$SmallTable$ is a hash table using the label in the PIT entry as the key, and all the entries in $SmallTable$ are polled from the head of $BigTable$ priority queue. Therefore, $SmallTable$ always holding the entries closest to their expiry time points in the whole PIT.

Algorithm 1 PIT replacement strategy

```
Input:
    $ipkt$: Interest packet
    $T_B$: $BigTable$ of the PIT
    $T_S$: $SmallTable$ of the PIT
Main Program:
 1: traverse($T_S$)
 2: $E_{min}$ ← the entry with lowest importance in $T_S$
 3: if $ipkt.I_I \leq E_{min}.I_{pe}$ then
 4:     discard($ipkt$)
 5: else
 6:     $T_S$.remove($E_{min}$)
 7:     $E_{head} \leftarrow T_B$.pollHead()
 8:     $T_S$.put($E_{head}$)
 9:     $T_B$.insert($ipkt$)
10: end if
```

C. PIT replacement strategy

Algorithm 1 describes our PIT replacement strategy. First, the router traverses $SmallTable$ to find the least important entry. If the importance of this entry is lower than that of the arrival Interest, the entry is removed from $SmallTable$. Then the head entry of $BigTable$ is polled and inserted to $SmallTable$. Finally the Interest is inserted to $BigTable$. If the Interest has lower importance, it is discarded. Assume the capacity of $SmallTable$ is N_{ST}. This algorithm needs $O(N_{ST})$ time to traverse $SmallTable$, $O(lgN_{BT})$ time to poll the head entry from $BigTable$ and $O(lgN_{BT})$ time to insert the Interest to $BigTable$. As N_{ST} is generally larger than lgN_{BT}, the time complexity of our strategy is $O(N_{ST})$, much more efficient than traversing the whole PIT.

VI. Performance evaluation

In this section we evaluate the performance of RAPIT. Our evaluations are performed by simulations in two steps. In Step 1, we generate a simple topology to evaluate the performance of a single RAPIT router. In Step 2, we use real network topologies selected from Rocketfuel [19] to evaluate the performance of networks deployed with RAPIT.

A. Simulation settings

We implement an event-driven Java simulator to conduct simulations. In our simulations, a content name has a part P of 16-bit integer which is the publisher node number in the topology, and a part L of 32-bit integer. The contents in the network follows Zipf distribution [6] with the parameter of 0.75. To balance network load, all publishers have the same probability to be chosen. The publishers drop Interests with a certain rate (packet loss rate, PLR) to simulate packet loss in the network caused by congress or link failure. Note that the Interests discarded in PIT replacements are not counted in PLR. The default PLR is 2% [20]. We measure RTT every 3 seconds and run the simulator for 300s in every simulation.

In Step 1, as Fig. 6 shows, the topology has one router, four consumers and four content publishers. The PIT of the router has a 4000-entry $BigTable$ and a 96-entry $SmallTable$. We set the RTTs between the router and Publisher 1∼4 to 80ms [14], 60ms, 100ms, and 120ms respectively. As every publisher has the same probability to be chosen by consumers, the RTT expectation is 90ms. If there is no timeout entry, the highest Interest arrival rate (IAR) the PIT can handle is $(4000+96)/0.09 \approx 45.5$K pkt/s. To evaluate the performance of RAPIT when IAR exceeds normal level, we set the default IAR to 48K pkt/s, 5% higher than 45.5K pkt/s.

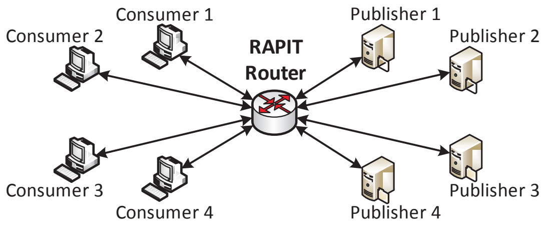

Fig. 6. The topology of single router performance evaluation

In this step we study the benefit brought by RAPIT on a single router, and show the advantages of RAPIT over several schemes: 1) the fixed PIT entry residence time setting scheme ($Fixed$); 2) the dynamic PIT entry residence time setting scheme without redundancy ($NonRddt$); 3)the random PIT replacement strategy ($Random$).

In Step 2 we use five PoP-level topologies collected from Rocketfuel and Table I shows a summary of them. To avoid direct link between a consumer and a publisher, only single-homed node can be consumer or publisher. 50%-60% of the

TABLE I
REAL TOPOLOGIES USED IN THE SIMULATIONS

	node	edge	consumer	N_{BT}	N_{ST}
AS-174	42	76	10	60n+683	2n+20
AS-1239	52	84	11	120n+925	2n+30
AS-2914	70	111	12	42n+485	2n+30
AS-3356	62	284	9	30n+260	2n+25
AS-3561	92	329	16	80n+500	2n+30

single-homed nodes are randomly selected as consumers and the rest ones are publishers; all multi-homed nodes serve as routers. We consider that a router with more neighbors needs larger PIT, thus for a router with n neighbors, we set the sizes of $BigTable$ and $SmallTable$ in the form of $a \times n + b$, in which a and b are factors differing by topologies. In this step we study the impact of RAPIT on network throughput.

B. Performance evaluation on single RAPIT router

Fig. 7, 8 and 9 show the advantages of RAPIT over fixed PIT entry residence time (T_{res}) setting schemes. We have three control schemes setting T_{res} to 4s [17], 1s [15] and 360ms, respectively. According to the largest RTT in the network (120ms) and Eq. 3, we set the minimum fixed T_{res} in control schemes to 360ms. We define PIT replacement demand ratio (RDR) here, which is the ratio of the amount of PIT replacements to the total amount of the Interests the processed by the PIT. High RDR indicates the PIT is always fully filled, which may cause more packet losses.

Fig. 7 and 8 show the RDRs at different Interest arrival rates (IAR) and packet loss rates (PLR). At the default PLR (2%), $Fixed-4s$ has an RDR higher than 34%, even when the PLR is very low (0.4%), the RDR is still high (9%). The RDR of $Fixed-1s$ is acceptable only at low IAR (44K pkt/s) and PLR (0.4%), when the IAR exceeds 45K pkt/s or the PLR exceeds 1.2%, the RDR exceeds 6%. $Fixed-360ms$ has almost the same RDR as RAPIT does at low IAR ($\leq$46K pkt/s) and PLR ($\leq$0.8%). However, as the IAR and PLR increase, RAPIT begins to show advantages over $Fixed-360ms$. At the default IAR (48K pkt/s) and PLR, the RDR in RAPIT is only 0.44%, 71% lower than that in $Fixed-360ms$. When the IAR and PLR further increase, due to more arrival Interests and less usable space in the PIT, RDRs in both RAPIT and $Fixed-360ms$ increase, yet the RDR of RAPIT is still at least 35% lower than that of $Fixed-360ms$.

Fig. 9 shows the variations of the PIT size (counted by the amount of entries) in different schemes, when the PIT has enough entries to avoid PIT replacement. We record the PIT size every 3s from a stable period of 300s during the simulation. The average PIT sizes in RAPIT, $Fixed-360ms$, $Fixed-1s$ and $Fixed-4s$ are 4017, 4102, 4621 and 6978 respectively. Compared with $Fixed-4s$ and $Fixed-1s$, RAPIT requires 42.4% and 13.1% less PIT entries to handle the same IAR, respectively. $Fixed-360ms$ requires 2.1% more PIT entries than RAPIT does.

In the aforementioned simulations, for most PIT entries, their T_{res} range from 120ms$\sim$360ms in RAPIT, significantly lower than that in the $Fixed$ schemes. In RAPIT, the timeout entries stays in the PIT for very short time, thus the occupied memory space are soon reused to record arrival Interests. Therefore, compared with the $Fixed$ schemes, RAPIT has lower RDR and requires smaller memory space. We also find that $Fixed-4s$ has the worst performance. $Fixed-1s$ performs better, yet still not well enough. Though the performance of $Fixed-360ms$ is acceptable, note that 360ms is set based on the largest RTT in the network, which means this scheme actually needs the same cost as RAPIT does ($e.g.$ RTT measurement and storage).

Fig. 10 is the CDF of the Data unmatched rate (DUR) for RAPIT and the $NonRddt$ when RTT varies in the network. DUR is the ratio of the amount of Datas missed in the PIT queries to the total amount of the Datas processed by the PIT. As the heavy-tailed Pareto distribution approximates the actual RTT distribution in the Internet [3] most accurately, we assume the RTT varies following a type I Pareto distribution Pareto(I)(σ, α) [11], with the scale parameter σ, the shape parameter α and the distribution function $F(x) = 1 - (\frac{\sigma}{x})^{\alpha}$.

Here we set the following parameters: 1) the RTT ranges from σ to σ+80ms [20]; 2) the 1/18th of the range covers 75% of the distribution [3]; 3) the mean value of RTT is the set value (80/60/100/120ms) in $SubsectionA$. Based on these settings, for the RTT distributions between the RAPIT Router and Publisher 1$\sim$4, we set their (σ, α) to (76.75, 24.63), (56.74, 18.38), (96.76, 30.86) and (116.77, 37.10), respectively. The three $NonRddt$ schemes set T_{res} directly to the measured RTT, but update the RTT in FIB entry only when the measured value is larger than the stored value. To keep the freshness of RTT, they reset the stored RTT to a smaller initial value every 0.5, 1 and 2min, respectively.

Fig. 10 shows that the DUR of RAPIT is always lower than 0.08%. Three $NonRddt$ schemes set the T_{res} to the maximum measured RTT during a time window to deal with RTT variation, yet in only 15% of the cases their DURs are lower than 2%, and the highest DURs reach 7.2%, 5.3% and 3.9%, respectively. Though longer time window brings lower DUR, it actually harms the dynamic of the T_{res} setting. The result indicates that leaving redundancy in T_{res} setting is necessary and effective to improve the robust to RTT variation.

Fig. 11 and 12 compare the PIT replacement strategy in RAPIT with random strategy. We define replacement loss as the amount of the discarded Interests and mistaken replacements. In mistaken replacements, PIT entries that can be responded are deleted. Then we define PIT Replacement Loss Rate (RLR) as the ratio of the replacement loss to the total amount of the Interests processed by the PIT. In Fig. 11 we denote different curves by the form of $Scheme - IAR$.

Fig 11 shows RAPIT always has a much lower RLR than $Random$ does in the same case, thus the variation of PLR has less impact on the RLR in RAPIT than in $Random$. When the IAR is 49K pkt/s and the PLR is 2%, the RLR in RAPIT is only 0.34% while the RLR in $Random$ reaches 1.9%, which is unacceptable considering the 2% PLR in the network. Moreover, though not displayed in the figure, there are always less than 10 mistaken replacements among the replacement losses in RAPIT, while more than 90% of the replacement losses of

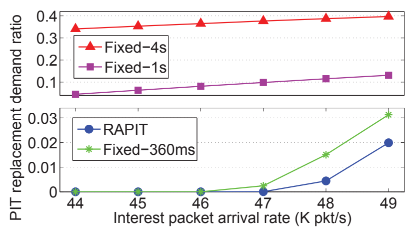

Fig. 7. PIT replacement demand ratio vs. IAR

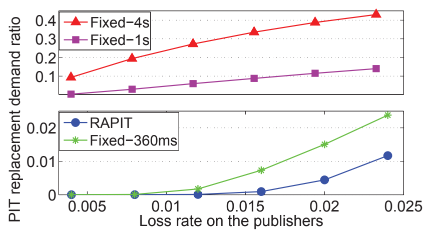

Fig. 8. PIT replacement demand ratio vs PLR

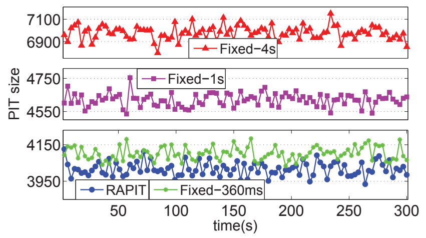

Fig. 9. PIT size vs. time

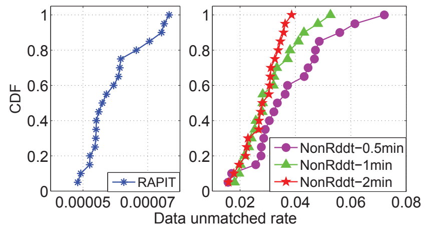

Fig. 10. The CDF of Data unmatched rate

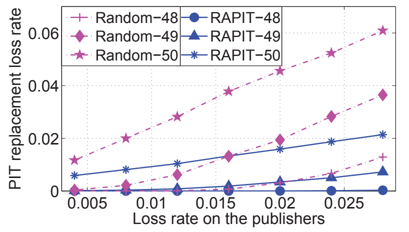

Fig. 11. PIT replacement loss rate vs PLR

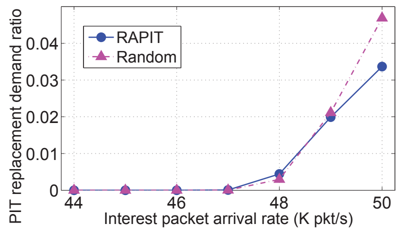

Fig. 12. PIT replacement demand ratio vs. IAR

$Random$ are mistaken replacements. RAPIT achieves nearly 100% accuracy, thus much fewer PIT entries with responses in transmissions are replaced improperly.

Though $Random$ needs less time for each replacement, Fig. 12 shows that this advantage does not help improve performance. $Random$ has lower RDR than the RAPIT does when the IAR $\leq$ 48K pkt/s, as higher replacement speed helps reduce RDR. However, when the IAR exceeds 49K pkt/s, due to the poor accuracy, $Random$ leaves too many timeout entries in the PIT, thus its RDR exceeds that of RAPIT.

These results lead us to the following conclusions: 1) RAPIT significantly improve the processing and space efficiency of the PIT, compared with fixed T_{res} setting scheme; 2) RAPIT shows very high adaptability to RTT variation; 3) RAPIT achieves the goal to recognize non-responded entries in the PIT replacement, thus has very high replacement accuracy; 4) RAPIT can handle high network load up to 108% (49K pkt/s here) of the normal level (45.5K pkt/s) with a very small cost.

C. Simulations on real topologies

If a consumer sends an Interest but does not get responded during the lifetime of the Interest, this Interest is "unsatisfied". It is important to keep the Interest unsatisfied rate (IUR) of the consumer acceptable while increasing network throughput. Considering the 2% PLR of the network, we assume 5% is acceptable for IUR. Table I shows some PIT size settings that limit the average IUR of RAPIT in different topologies to 5% at a request rate of 12K pkt/s. In this step, with the PIT size settings in Table I, we compare the effect of RAPIT on network throughput with two control schemes: $Fixed-4s$ and First In First Out replacement strategy ($FIFO$).

Fig. 13 shows the average IUR of different schemes in the five topologies. When RAPIT has an IUR of 5%, $FIFO$ causes 10%$\sim$23% Interests to be unsatisfied, at least twice as many as that of RAPIT. The performance of the $Fixed-4s$ is completely unacceptable: it causes an IUR higher than 87%.

Then we study the IUR variation with request rate in AS-1239. As shown in Fig. 14, $Fixed-4s$ has a quite low IUR of about 2.75% before the request rate exceeds 4K pkt/s, yet 5K pkt/s or higher request rate cause the IUR to jump to higher than 80%. When the request rate $\leq$ 8K pkt/s, RAPIT has almost the same IUR as $FIFO$ does. The advantage of RAPIT is highlighted when the request rate $>$ 8K pkt/s: the increase of IUR in RAPIT is slower than that in $FIFO$. When the request rate is 12K pkt/s, the IUR of RAPIT just reaches the upper limit while the IUR of $FIFO$ has exceeded 10%.

Fig. 15 shows the highest request rates from the consumers of different schemes with the settings in Table I. In RAPIT, consumers keep a request rate of 12K pkt/s. Compared with $FIFO$, in three of the five topologies RAPIT raises the highest request rate by 50%, in the rest two topologies the increase also reach 20% and 33% respectively. Still, $Fixed-4s$ shows poor performance. compared with RAPIT, in $Fixed-4s$ the consumers have to cut down their request rates by at least 67% to bring the IUR down to the acceptable level.

The simulations on real topologies indicates that the efficiency improvement of RAPIT on the PIT increases the network throughput indeed. Compared with $FIFO$ and the $Fixed-4s$, RAPIT increases the network throughput by 20%$\sim$50% and at least 200%, respectively.

VII. Related works

There have been a lot of works to improve PIT performance for CCN. These works can be classified into two directions: to optimize PIT structure, or to optimize PIT entry residence time (T_{res}). Up to now the PIT performance researches have mainly focus on PIT structure optimization. Many typical solutions in this direction bring significant performance improvement, $e.g.$ DiPIT [23], NCE [8] and Scalable Pending Interest Table Design [24].

It has been noticed that optimizing T_{res} can also improve PIT performance. Paper [24] mentions that overlong T_{res} is "overkilled", thus T_{res} should be set dynamically. A research

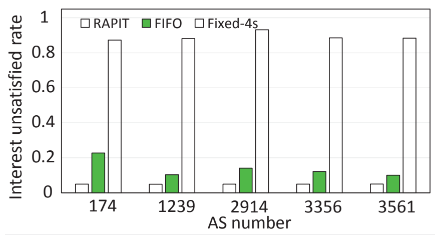

Fig. 13. Average IUR in real topologies

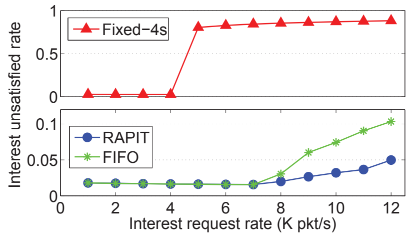

Fig. 14. Average IUR vs. request rate in AS-1239

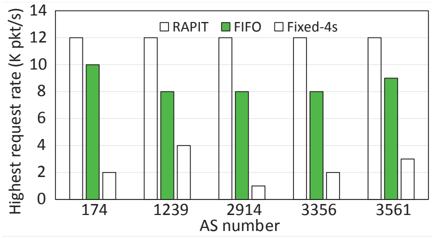

Fig. 15. The highest supported request rate

on PIT overload [22] indicates that smarter T_{res} setting scheme can be a countermeasure against DDoS attack. In 2014, Abu *et al.* propose a new scheme [2] and prove the potential of this direction. The scheme sets T_{res} to the maximum RTT over all requested Datas received in a time window of one minute, thus outperforming the T_{res} setting scheme in CCNx.

Among all these PIT performance improvement schemes, ones based on PIT structure optimization have gotten good results, yet few of them consider residence time. On the other hand, most ideas on dynamic PIT entry residence time setting lack further researches. Although Abu *et al.* propose a relatively complete scheme, they leave some problems unsolved (*e.g.* RTT storage and the robust to RTT variation). In this paper, RAPIT provides alternative answers to these problems.

VIII. Conclusion

In this paper we analyze the factors related to the PIT entry residence time optimization, then based on these factors, we propose RAPIT to improve PIT performance. There are two main points in RAPIT: a dynamic PIT entry residence time setting scheme based on RTT, and a PIT replacement strategy based on the priorities of PIT entries and Interests. Evaluations suggest that: 1) the dynamic PIT entry residence time setting scheme significantly improves the processing efficiency of PIT; 2) the PIT replacement strategy has high accuracy and low replacement loss. Because of the two aforementioned advantages, RAPIT is able to increase network throughput significantly, compared with the fixed-value residence time setting scheme and the *FIFO* replacement strategy.

IX. Acknowledgement

This work is supported by the National Research Program of China (973) under grant No. 2012CB315803, the National Science Foundation of China under grant No. 61402255, the R&D Program of Shenzhen under grant No. ZDSYS20140509172959989, No. JSGG20150512162853495, and No. Shenfagai[2015]986.

References

[1] *2014 Content Distribution Network (CDN) White Paper*. White Paper. China Academy of Telecommunication Research of MIIT, Oct. 2014.
[2] A. J. Abu, B. Bensaou, and J. M. Wang. "Interest packets retransmission in lossy CCN networks and its impact on network performance". In: *Proceedings of ACM ICN*. Paris, France, Sept. 2014, pp. 167–176.
[3] A. Acharya and J. Saltz. *A study of internet round-trip delay*. Tech. rep. UMIACS and Department of Computer Science, University of Maryland, Oct. 1998.
[4] B. Ahlgren et al. "Design considerations for a network of information". In: *Proceedings of ACM CoNEXT*. Madrid, Spain, Dec. 2008, pp. 2049–2057.
[5] *BGP Report*. http://bgp.potaroo.net/. Feb. 2015.
[6] L. Breslau et al. "Web caching and Zipf-like distributions: Evidence and implications". In: *Proceedings of IEEE INFOCOMM*. New York, USA, Mar. 1999, pp. 126–134.
[7] *Cypress Reports First-Quarter 2009 Results*. http://www.cypress.com/. May 2015.
[8] H. Dai et al. "On pending interest table in named data networking". In: *Proceedings of ACM ACNS*. Austin, USA, Oct. 2012, pp. 211–222.
[9] *IEEE 802.3 ETHERNET*. http://www.ieee802.org/3/. Dec. 2014.
[10] V. Jacobson et al. "Networking named content". In: *Proceedings of ACM CoNEXT*. Rome, Italy, Dec. 2009, pp. 1–12.
[11] A. Kesselman and Y. Mansour. "Optimizing TCP retransmission timeout". In: *Proceedings of Springer ICN*. Reunion Island, France, Apr. 2005, pp. 133–140.
[12] T. Koponen et al. "A data-oriented (and beyond) network architecture". In: *Proceedings of ACM SIGCOMM*. Kyoto, Japan, Aug. 2007, pp. 181–192.
[13] D. Lagutin, K. Visala, and S. Tarkoma. "Publish/Subscribe for Internet: PSIRP Perspective". In: *Towards the Future Internet* 84 (2010), pp. 75–84.
[14] *Meridian Project*. http://www.cs.cornell.edu/. July 2015.
[15] *Named Data Networking*. http://named-data.net/. June 2015.
[16] D. Perino and M. Varvello. "A reality check for content centric networking". In: *Proceedings of ACM SIGCOMM workshop on ICN*. Toronto, Canada, Aug. 2011, pp. 44–49.
[17] *Project CCNx*. http://www.ccnx.org/. Mar. 2015.
[18] *Semiconductor Consulting*. http://www.maltiel.com/. Nov. 2014.
[19] N. Spring, R. Mahajan, and D. Wetherall. "Measuring ISP topologies with Rocketfuel". In: *Proceedings of ACM SIGCOMM*. Pittsburgh, USA, Aug. 2002, pp. 133–145.
[20] *Telecommunication Services Rules*. http://www.miit.gov.cn/. Mar. 2015.
[21] *The Zettabyte Era: Trends and Analysis*. White Paper. Cisco, May 2015.
[22] M. Virgilio, G. Marchetto, and R. Sisto. "Pit overload analysis in content centric networks". In: *Proceedings of ACM SIGCOMM workshop on ICN*. HongKong, China, Dec. 2013, pp. 67–72.
[23] W. You et al. "Dipit: A distributed bloom-filter based pit table for ccn nodes". In: *Proceedings of IEEE ICCCN*. Munich, Germany, July 2012, pp. 1–7.
[24] H. Yuan and P. Crowley. "Scalable pending interest table design: From principles to practice". In: *Proceedings of IEEE INFOCOM*. Toronto, Canada, Apr. 2014, pp. 2049–2057.