SMIC: Subflow-level Multi-path Interest Control for Information Centric Networking

Junghwan Song
School of Computer Science and Engineering, Seoul National University
Seoul, Korea
jhsong@mmlab.snu.ac.kr

Munyoung Lee
Electronics and Telecommunications Research Institute
Daejeon, Korea
mylee@mmlab.snu.ac.kr

Ted "Taekyoung" Kwon
School of Computer Science and Engineering, Seoul National University
Seoul, Korea
tkkwon@snu.ac.kr

ABSTRACT

In this paper, we focus on multi-path transport layer mechanisms of ICN. We first provide the comprehensive analysis of design criteria and possible solutions for designing multi-path Interest forwarding and congestion control mechanisms. We then propose a Subflow-level Multi-path Interest Control (SMIC) scheme to enhance the performances of content deliveries over multiple paths, while achieving throughput-friendliness with single path flows (controlled by TCP-like congestion control). Simulation results show that SMIC shortens the content retrieval time compared to other multi-path transport schemes in ICN, achieves the throughput-friendliness with single-path flows, and mitigates the traffic load imbalance among the network links.

CCS CONCEPTS

•**Networks → Transport protocols;**

KEYWORDS

ICN; Congestion control; Multipath flows

ACM Reference format:
Junghwan Song, Munyoung Lee, and Ted "Taekyoung" Kwon. 2018. SMIC: Subflow-level Multi-path Interest Control
for Information Centric Networking. In *Proceedings of ICN '18: 5th ACM Conference on Information-Centric Networking, Boston, MA, USA, September 21–23, 2018 (ICN '18),* 11 pages.
DOI: 10.1145/3267955.3267971

1 INTRODUCTION

Information Centric Networking (ICN) [10] has received a lot of attention since its architectural design reflects the vast majority of content-centric Internet services. We focus on transport layer mechanisms like congestion control in ICN, which has been actively studied (e.g., [3–5]). Especially, we study how to make use of multiple paths to one or more content holders, which has gained attention recently (e.g. [13–15]) since getting a content object from its multiple copies is one of the key advantages of ICN.

ICN '18, Boston, MA, USA

DOI: 10.1145/3267955.3267971

In this paper, a flow refers to the traffic in both directions between a consumer and one or more content holders that exchange Interests and data packets. Then a subflow is defined to be Interests[1] and data packets forwarded in both directions over a particular path. Different subflows may overlap along a certain segment between each other, but not the entire path. We seek to design a subflow-level congestion control mechanism to utilize the capacity of multiple paths.

A congestion control mechanism based on subflows may have the following merits if they operate multiple congestion windows, compared with the one with a single congestion window. First, the interference between different subflows will be mitigated. If a subflow suffers from frequent packet drops, only its window will be reduced, and other flows will adjust their windows on their own. Second, how to schedule Interests among multiple candidate paths may not be crucial with the subflow-level congestion windows. In case of a single congestion window, we should be careful in choosing subflows for sending Interests to achieve high throughput. However, in case of subflow-level congestion control, subflows with the higher throughput would be used more naturally since they have the larger congestion windows on average. Finally, using subflow-level congestion windows can exploit "fast retransmit" in TCP [2]. In multi-path ICN, receiving out-of-order data packets may occur frequently since they will arrive in an interleaved fashion over multiple paths. However, congestion control with a single congestion window may find it difficult to support fast retransmit since data packets are coming over multiple paths. On the other hand, with subflow-level congestion windows, fast retransmit can be supported at subflow level since the Interests for the same subflow will be forwarded over the same path.

To the best of our knowledge, window-based congestion control mechanisms in the ICN literature use only a single congestion window for a flow even if they use multiple paths (toward one or more content holders). We first analyze the design criteria and solution space in congestion control and Interest forwarding. After that, we propose subflow-level multi-path congestion control and Interest forwarding mechanisms. The main idea of the proposed mechanisms is sending Interests to one or more content holders (i.e., content publishers and network caches) through multiple subflows, while identifying each subflow and managing the subflow information on the consumer side.

More specifically, we address the following questions in this paper: (Q1) How to do congestion control at subflow level?, and (Q2)

[1]We use an Interest and an Interest packet exchangeably.

How to identify different subflows that follow different paths? The first question is about how to adjust congestion windows of multiple subflows. If multiple subflows share the same bottleneck, what should be done for throughput-friendliness with single-path flows. The second question arises due to the lack of the endpoint location in ICN, which makes the consumer unable to identify individual subflows and to control traffic at subflow-level granularity. Note that the importance of identifying paths in ICN is also discussed in PathSwitching [14], which allows for consumers to keep track of the subflow-level information such as the number of in-flight Interests.

The contributions of this paper can be summarized as follows: We propose a Subflow-level Multi-path Interest Control (SMIC) scheme, which substantiates Interest forwarding and congestion control mechanisms to support multi-path content deliveries in ICN. We discuss the design criteria and analyze the possible alternatives for designing a multi-path congestion control scheme. We evaluate the performance of SMIC in terms of the content retrieval time and throughput-friendliness with single-path flows.

The remainder of this paper is organized as follows. In Section 2, we survey the related work. In Section 3, we describe design issues and our approaches for designing SMIC, which is detailed in Section 4. After that, we evaluate SMIC by simulation in Section 5. Finally, the paper is concluded in Section 7.

2 RELATED WORK

One of the most well-known transport layer protocols for ICN is Interest Control Protocol (ICP) [3]. ICP introduces a window-based Interest congestion control protocol driven by an Additive Increase Multiplicative Decrease (AIMD) mechanism. ICP controls a window of Interests rather than that of data packets since a consumer can trigger data transmissions by issuing Interests. ICP is extended by suggesting joint hop-by-hop and receiver-driven ICP [4] that controls the Interest rate (in upstream direction) by predicting the throughput in downstream. Another extension of ICP is made by proposing Multipath Congestion Control [5] that drops Interests probabilistically at routers by introducing Remote Adaptive Active Queue Management (RAAQM). In [6], the authors formulate a global optimization problem to find an optimal path that maximizes the user throughput and minimizes the overall network cost. These studies focus on ICN transport layer mechanisms by mathematical formulation with the ICN model.

Another category of studies have focused on finding the close content copies to enhance the delivery performance by modifying the Interest forwarding mechanism. INFORM [7] introduces hop-by-hop dynamic request forwarding to discover content copies through the best interface in terms of performance metrics like throughput. When an intermediate router receives Interests, it forwards Interests through the interface with the minimum delivery time for the content object. In [17], the authors propose an adaptive forwarding scheme by recording the RTTs of PIT entries. Based on the measured RTTs, the proposed scheme decides forwarding policies of all the interfaces in a FIB entry. This scheme can shorten the content retrieval time by selecting the fastest interface when Interests are forwarded. More recently, how to identify each path and forward Interests to a particular path is studied, called PathSwitching [14]. To identify individual paths and scheduling Interests over the paths, the authors introduce a path label. Each router encodes the incoming interface into the path label of the Data packet. Eventually, a consumer receives the data packet with the encoded path label, and discovers a new path by the path label. The size of a path label increases as the number of intermediate routers increases. Some encoding techniques are suggested; however, it is difficult to be adopted in large networks.

Recently, more sophisticated studies to utilize multiple paths are proposed. In [13], a multi path-aware rate-based congestion control scheme is suggested. The authors introduce a rate calculation algorithm that revises mathematical equations in RCP [8]. They extend single-path MIRCC to a multi-path scenario. They split flows into two classes, the primary and secondary, and perform different congestion control for the two classes. In [15], window-based congestion control that can support multiple paths is suggested. In this work, the authors introduce an explicit congestion notification mechanism. Data packets passing through congested paths are marked by monitoring the outgoing queues of intermediate routers. When consumers receive the marked data packets, they adjust their window sizes. Routers forward some of Interests to one of not-the-best interfaces in the FIB entry after receiving marked data packets.

In the TCP/IP literature, there have been many studies about multiple path utilization. One of the most well-known works is MultiPath TCP (MPTCP) [9, 16]. A flow using MPTCP can get at least as much throughput as a TCP flow, while MPTCP guarantees that MPTCP-flows do not unduly harm other TCP-flows at a bottleneck link. MPTCP is proved mathematically and widely approved; however, it cannot be adapted to ICN due to the path-unawareness characteristics of ICN.

3 DESIGN CONSIDERATIONS

In this section, we first discuss the criteria for designing the SMIC scheme in ICN. Then we consider the potential approaches to satisfy the criteria.

3.1 Design Criteria for SMIC

When a consumer wishes to get a content object, she sends Interests (each specifying the name of a piece of the object). Then the producer (who generates the content object) will send back the requested data packets over the same path that the Interest traverses. The Interests and the corresponding data packets for the consumer are collectively referred to as "a flow". If there are multiple paths between the consumer and the producer, the Interests and the data packets of the same flow may be forwarded over the multiple paths. Note that there can be multiple data holders as well; the data can be replicated like CDNs or cached in in-network entities. With the multiple data holders, there would be more multiple paths. The Interests and the data packets forwarded over the same path (in both directions) are referred to be a subflow. Thus, a congestion control mechanism for the multi-path transport would be totally different from the one for the single-path transport [9, 16]. In prior studies [5, 6, 15], only a single congestion window is used when the consumer receives the pieces of the content object over multiple

paths. However, if only a single congestion window is applied to multi-path networking environments, it may cause the subflow interference problem or the path selection problem (to be discussed in Section 3.2).

A challenging issue in Interest forwarding in multi-path environments is caused by the inherent nature of ICN. In ICN, a content name is used for routing/forwarding. A content object is accessed by its content name, not by its location (i.e, IP address). Note that it is assumed that a content object can be replicated and stored at multiple locations in this paper. (If there is only a single-path for a single copy, our scheme will fall back to a single-path case.) A consumer simply sends Interests with the content name without caring about the locations of the same content object (and hence paths towards its copies). Then the question that arises is how we can identify and control subflows without the knowledge of content locations. Also, how to schedule/partition Interests among the multiple subflows needs to be investigated. To address the above issues, we take the similar approach to the one in PathSwitching [14], which introduces a path label in a packet header. The PathSwitching scheme will be detailed in Section 3.3

In the following subsections, we detail the above two issues: **congestion control** and **forwarding & identification**. We also discuss possible solutions for each mechanism, and explain which solution is chosen for SMIC.

3.2 Congestion Control Mechanism

Since SMIC utilizes multiple paths, congestion control using only a single window like ICP [3] may not be suitable. Prior congestion control mechanisms in ICN are commonly designed based on a single-path networking model; thus they do not consider issues such as multiple congestion windows, the number of paths to use, and so on. However, if we utilize multiple paths, a flow may consist of multiple subflows. Thus we have multiple choices on the number of congestion windows: a single congestion window or multiple congestion windows.

3.2.1 ***Congestion Window: Single or Multiple?*** Since SMIC seeks to exploit multiple paths, we consider only the cases of multiple paths (or subflows) for a flow. As described in [5], there can be two choices on the number of congestion windows.

1. A single congestion window: A single congestion window may mean the simple adoption of a single-path congestion control in multi-path environments. A consumer keeps only a single congestion window W, which is adjusted based on the AIMD algorithm[2]. The consumer sends Interests if the number of in-flight Interests is less than W. Then there is no need to identify each path or store the additional information about paths.

Apparently, it has three drawbacks. The first one is the negative effect of congested paths on the overall throughput, called the *subflow interference problem*. For instance, if a timeout (while waiting for the requested data packet) happens due to a congestion in a path, it will decrease the congestion window. If there are n subflows, the window size may be decreased n times more frequently if n paths are non-overlapping at congestion points (which is not so likely though) and have similar congestion frequency. Thus, the other subflows are also degraded due to the packet losses of the congested path, resulting in overall throughput degradation.

The second one is that we have the limits in interpreting packet losses in ICN. In TCP congestion control, a timeout means the heavier congestion, and the reception of three duplicated acknowledgements means the lighter congestion ("fast retransmit" in TCP). However, in congestion control in ICN, it is difficult to use the duplicate acknowledgements as the indicator of a congestion since a consumer may receive data packets over multiple paths (potentially from multiple sources), which can result in out-of-order deliveries. ICN consumers cannot figure out which packet comes from which source/path and thus out-of-order arrivals (or duplicate acknowledgements) can happen without congestions. If ICN consumers can identify which packets are forwarded over which paths and adjust separate congestion windows accordingly, they can use the reception of duplicated acknowledgements to trigger "fast retransmit" in TCP.

The last one is the path selection problem. With a single congestion window, basically every path will have its share of the congestion window (of sending Interests) depending on some path selection algorithm. It is not easy to determine how much share of the window to be assigned to each subflow (or path) since path selection and path throughput are intertwined and time-varying.

2. Separate congestion windows: A separate congestion window can be assigned for each subflow. Each subflow r has its own congestion window w_r, and if the number of in-flight Interests of subflow r is less than its w_r, Interests can be sent through subflow r. A packet loss of a subflow does not affect other subflows; the windows of other subflows are not decreased, which solves the subflow interference problem.

Using separate congestion windows can solve the path selection problem naturally since each subflow r will send Interests whenever its congestion window w_r becomes available. Thus, by using separate congestion windows, the paths (or subflows) with the higher bandwidth capacities are likely to be used to carry more traffic than the ones with the lower bandwidth capacities. However, to use separate congestion windows effectively, each path should be identified. (The issue of overlapping links or segments among the multiple paths will be addressed later.) For this, we introduce a notion of a "path identifier," which will be detailed in Section 3.3.2.

3.2.2 ***Window control algorithm***. SMIC chooses to use a separate congestion window for each subflow to address the problems of a single congestion window. Now, we need to substantiate an algorithm to control each congestion window size depending on the dynamic conditions of the subflows. As already shown in [16], a window control algorithm of multiple paths must be different from that of single path.

Last but not least, we have another goal in terms of fairness. A SMIC flow should have an equal share of a bottleneck link with a single-path flow even if its multiple subflows go through the link. This goal is similar to that of MPTCP since SMIC and MPTCP have similar motivations and environments as follows. (1) With multiple available paths for a flow, we seek to make a flow use network resources efficiently. (2) They have similar operating environments that a flow can exploit multiple paths, and a consumer holds a congestion window for each subflow. These make SMIC and MPTCP

[2]We simply use the packet count instead of the byte count.

have similarities in their window control algorithms. As to how to adjust the window size on congestions, we basically adopt the Linked Increase Algorithm (LIA) in MPTCP [16] except for how to handle subflows that share the same bottleneck point.

The LIA algorithm focuses on the fair share of a bottleneck link when MPTCP and TCP flows coexist. As a consumer cannot identify which subflows are sharing the bottleneck link with TCP flows, LIA increases its congestion window size conservatively (for the worst case that all of its subflows share the bottleneck link). It might result in unnecessarily slow data transfer of traffic in MPTCP flows. Thus, if a consumer can identify which subflows share the bottleneck link and adjust their congestion windows accordingly, we can expect the higher throughput. The congestion control algorithm in SMIC is designed to enhance the throughput by modifying the conservative LIA algorithm leveraging the subflow identification. The central ideas of SMIC are: (i) detecting subflows sharing the bottleneck link, and (ii) adjusting the window increase rate of shared bottleneck subflows accordingly for TCP friendliness.

1. Bottleneck-sharing subflow detection: The first key feature is detecting subflows sharing a bottleneck link. In this paper, we consider two parameters for detecting bottleneck-sharing subflows. One is the bandwidth of the bottleneck link, and the other is the packet loss pattern. A consumer estimates the bottleneck bandwidth of each subflow by relying on a simple packet pair technique [12]. When a timeout occurs on subflow r, a consumer records the timeout information and keeps the history of recent n timeouts. Then, a consumer compares the packet loss pattern (i.e., the timeout history) of subflow r with those of others. To calculate a similarity between two timeout histories, we can consider a few methods to compare two vectors: the Euclidean distance, cosine similarity, Root Mean Square Error (RMSE), etc. We select the RMSE as the similarity score since the number of elements in timeout histories can be different among subflows. If the RMSE score is smaller than a threshold, we add the compared subflow to subflow r's candidate subflows, and vice versa. The threshold is important since if the threshold is too small, it is deemed that there is no candidate subflow that share the bottleneck. If the threshold is too large, two many subflows are considered as the ones that share the bottleneck. However, this might be not so critical since there is another filter for the bottleneck-sharing detection, which is the estimated bottleneck link bandwidth. If a consumer finds candidate subflows that have similar packet loss patterns with subflow r, then the estimated link bandwidth of the bottleneck of the candidate subflows are compared. Finally, the candidate subflows that do not have similar bottleneck bandwidth with subflow r are removed; thus the remaining subflows are considered as the bottleneck-sharing subflows.

2. Bottleneck-sharing subflow-aware window control: Basically, if a single SMIC subflow shares the bottleneck link with a single-path flow, TCP-like congestion control will suffice for each flow to get a fair share. However, if n SMIC subflows share the bottleneck link with a single-path flow, their aggregated throughput may overwhelm that of the single-path flow. Inspired by the recent MPTCP improvements, SMIC reduces the window increase rate of subflows sharing the bottleneck link by $1/n^2$ to share the bottleneck link fairly with single-path flows. Let us denote by r and w_r the subflow of interest and its window size, respectively. If the RTT variation of the subflows is small, we can achieve the single flow-friendliness by increasing w_r by $\frac{1}{n^2 w_r}$ for the arrival of each data packet. Otherwise (if RTT variation is large), the LIA algorithm can be used among the bottleneck-sharing subflows. The final window control algorithm is SMIC is described by Algorithm 1.

Algorithm 1 SMIC window control algorithm

```
1: R = set of all subflows
2: b_r = estimated bottleneck bandwidth history of subflow r
3: t_r = timeout history of subflow r
4: S_r = set of subflows sharing bottleneck with subflow r (including r
   itself)
5: |S_r| = the number of elements in S_r
6: w_r ← current value of subflow r's cwnd
7: if Data arrives through subflow r then
8:     update b_r
9:     w_r ← w_r + 1/(|S_r|^2 w_r)
10: end if
11: if timeout loss is detected on subflow r then
12:     w_r ← w_r/2
13:     update t_r
14:     for all i ∈ R do
15:         if t_r, b_r is similar with t_i, b_i then
16:             S_r ← S_r ∪ {subflow i}
17:             S_i ← S_i ∪ {subflow r}
18:         end if
19:     end for
20: end if
```

3.3 Subflow Identification & Forwarding

In the ICN architecture, consumers do not care about the locations of content objects, not to mention paths towards them. To realize the subflow-level congestion control mechanism, however, we should be able to forward Interests to a specific subflow (or over a specific path) depending on its congestion window and the status of other subflows. To do so, we should first identify each subflow, so that we can manage the subflow-related information, e.g., its congestion window. There are two requirements to fully realize subflow-level congestion control: (R1) identifying each subflow to manage the subflow-relevant information, and (R2) forwarding Interests to a specific subflow.

As PathSwitching [14] addresses the above requirements, we first introduce the path discovery and path steering mechanisms in PathSwitching, and discuss the possible enhancements.

3.3.1 *PathSwitching*.

Path discovery: Satisfying the requirement (R1) is not easy since the ICN architecture does not have any mechanism for identifying subflows basically. To satisfy the requirement, PathSwitching [14] introduces a "path label" by adding a field in the header. The path label is updated by every router when a data packet is forwarded. Each intermediate router encodes the incoming interface into the path label of the data packet. Thus a consumer who receives the data packet (with the encoded path label) can discover a new path.

Path steering: For subflow-level congestion control, (R2) should be satisfied. To this end, Interests should pass through the same path

(or subflow) when they have the same path identifier. This concept is called path steering in [14]. Due to the symmetry of forward and reverse paths in ICN, a consumer can reuse the newly discovered path label to fetch the different piece of the same object over the discovered path. Path steering can be achieved by forwarding the Interest to the specified next hop in the path label when routers process the Interest.

Challenges in PathSwitching: The first issue in PathSwitching is the growth of the path label as the number of intermediate routers increases. Since the path label stores all the next hop information, with a naive approach, the size of the path label may grow up to N times M, where N is the number of hops and M is the size of next hop information. Some more efficient encoding techniques are considered in [14], such as Bloom filter, pairing function encoding, polynomial encoding and stack encoding, each of which has some limitations like false positives or path label increases.

3.3.2 *Our solution: path identifier and trajectory identifier.*

To handle the above issue, a path label is composed of two fixed-length identifiers: a "path identifier" and a "trajectory identifier." The path identifier is a field in an Interest header to indicate a particular subflow. The trajectory identifier is a field in the Data packet header, which guarantees uniqueness of the path over which the data packet is forwarded. The reason why we need two kinds of identifiers will be given in next paragraphs.

Path identifier: While the path label in PathSwitching is encoded by routers, the path identifier in an Interest is set by a consumer. There is a pool of path identifiers at the consumer (say, from 0 to 10), one of which is selected by the consumer whenever an Interest is generated.

When a consumer sends the Interest with the path identifier, the Interest packet reaches a router. Then, the router runs a hash function for the path identifier of the incoming Interest while looking up a FIB entry. After that, the router selects a next hop router (out of multiple candidates in the FIB entry) depending on the output of the hash function. For instance, if the number of next hop candidates is c, then the next hop would be $i = Hash(path\ identifier)$ *modulo* c. This operation will continue until the Interest reaches the content holder. Hash-based Interest forwarding might give weak guarantees that the available path space is actually explored, however, using broad pool of path identifiers could be the solution.

Trajectory identifier: By introducing the path identifier, Interests with the same path identifier will be forwarded over the same path. However, Interests (for the same content object) with different path identifiers might be forwarded to the same interface at every intermediate router. Although a hash function is assumed to generate different outputs from different inputs, different outputs do not necessarily result in different next hops since the number of possible interfaces toward the content holders is very limited. To solve this issue, we introduce a trajectory identifier which is located in a data packet header. The content holder initializes the trajectory identifier by hashing its own value, say its MAC address. While the data packet is delivered, every router updates the trajectory identifier (of an incoming data packet) by hashing it with the router's own value (say, its MAC address of an incoming interface). Unlike the path identifier, the trajectory identifier can guarantee

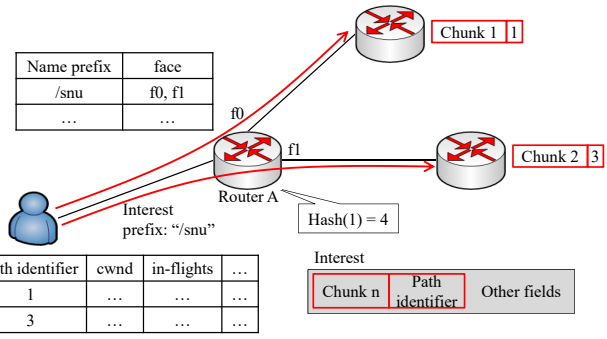


Figure 1: The operation of SMIC is illustrated. Interests are forwarded over different paths by using the path identifiers.

that different trajectory identifiers indicate different paths. With the sufficient size of the trajectory identifier, the probability of hash collisions is negligible.

When a consumer receives the data packet, its trajectory identifier indicates which path has been traversed. If Interests with different path identifiers correspond to the data packets with the same trajectory identifier, we can conclude that these path identifiers correspond to the same path. In this case, the path identifiers with the same trajectory identifier are merged.

Advantage of path & trajectory identifiers: Since both the path identifier and the trajectory identifier are of fixed length, there is no scalability issue. Note that SMIC can adapt to dynamic routing since the consumer will detect route changes as the trajectory identifier will be changed.

4 SMIC OPERATIONS

We now propose the SMIC scheme. SMIC controls congestion windows at subflow level, identifies subflows with the path and trajectory identifiers, and manages the subflow information. In this section, we describe how SMIC operates from the standpoints of consumers and routers.

4.1 Consumer Operations

As described in Section 3.3, consumers send Interests with different path identifiers for different content chunks. For each path, the subflow information should be managed by a consumer. Once an Interest is sent by a consumer, she records its path identifier and the content chunk name. To spread Interests over multiple paths, she sends multiple Interests with different path identifiers (say, rotating integers from 0 to 10). These Interests will be diffused through different paths, and the corresponding data packets will be received by the consumer through the different paths. Note that spreading Interests could be done in the initial phase of data transmissions and periodically (to adapt to dynamic network conditions). For example, in Fig. 1, the consumer sends two Interests with different path identifiers, '1' and '3'. The consumer then figures out there are two subflows with their path identifiers, and manages the subflow information for each subflow.

When the data packet is returned, the consumer can identify the corresponding subflow by using a tuple of (path identifier, content

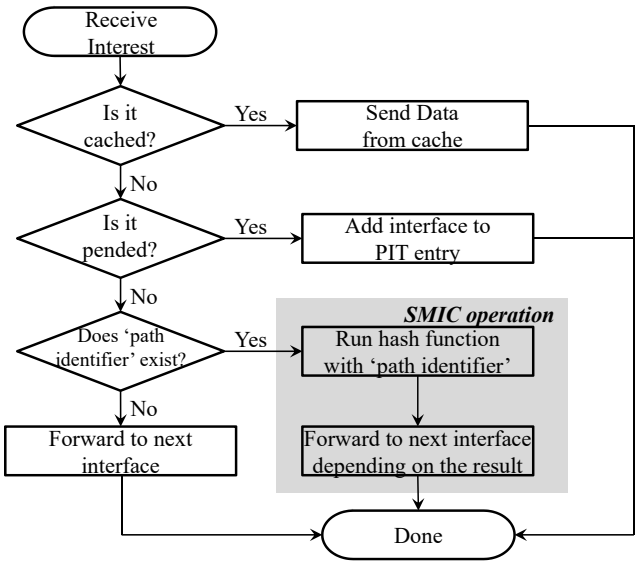


Figure 2: The forwarding operations of a router when an Interest arrives are illustrated.

chunk name, trajectory identifier), where the chunk name is used to bind the path identifier and the trajectory identifier. If there are path identifiers with the same trajectory identifier, the consumer will merge the path identifiers and their subflow information. Also she can update the estimated bottleneck link bandwidth, the congestion window size, the number of in-flight Interests and RTT information for each subflow. Especially, the estimated bottleneck bandwidth is used to detect the subflows that share the bottleneck link, and their congestion windows will be controlled by the subflow-level window control Algorithm 1 in Section 3.2.2.

4.2 Router Operations

When a router forwards an Interest based on the FIB, it first checks the number of possible (outgoing) interfaces in the corresponding FIB entry. If there are c possible outfaces, it runs a hash function for the path identifier and applies the modulo c operation, whose result is the index of the selected interface. For example, in Fig. 1, a consumer sends two Interests with different path identifiers, '1' and '3'. Chunk 1 has '1' as its path identifier, and Router A runs the hash function with the path identifier as its input value. The output of the hash function is 4, and there are two interfaces for the prefix "/snu" in Router A. Router A performs the modulo operation with the output by the number of its interfaces (i.e., 2) in the FIB entry. The result of the modulus operation is 0, so chunk 1 is forwarded to the interface f0. Similarly, in case of chunk 2, it has '3' as its path identifier so it is forwarded to f2 at the first router.

We summarize the operations of a router in a flow chart, Figs. 2 and 3. When an Interest arrives at a router in SMIC, its main operations are similar to those of a normal ICN router. It checks its CS, PIT, and FIB. When the CS and PIT do not have corresponding entries, the SMIC router checks whether the Interest has the path identifier or not. If the Interest has the path identifier, the router runs a hash function with the path identifier and then the modulo

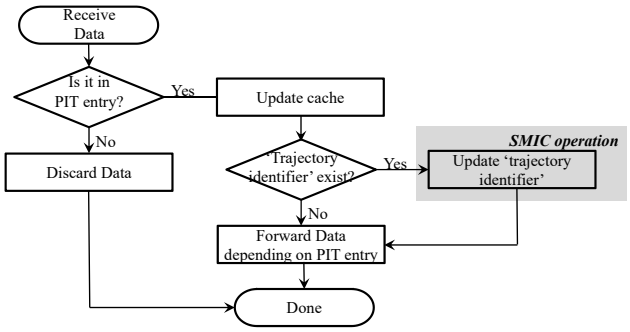


Figure 3: The operations of a router when a data packet arrives are illustrated.

operation. From the output, the router decides a next hop for the Interest. When a data packet arrives at the router, the router checks its PIT, and updates its CS. The router just needs to update a trajectory identifier with its own value by running a hash function.

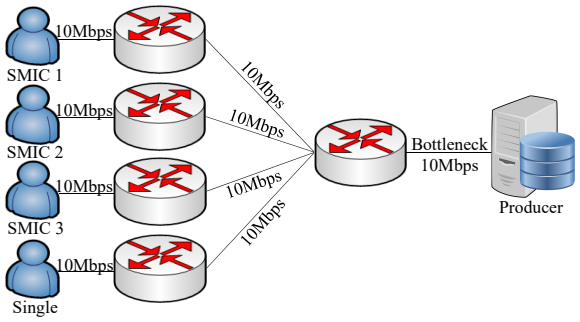
5 EVALUATION

To quantitatively evaluate, we implement SMIC using ndnSIM [1], which implements comprehensive ICN functional modules like caching, Content Store (CS), Pending Interest Table (PIT), Forwarding Information Base (FIB), name-based forwarding & routing, packet loss, and retransmissions. The simulation parameters are shown in Table 1.

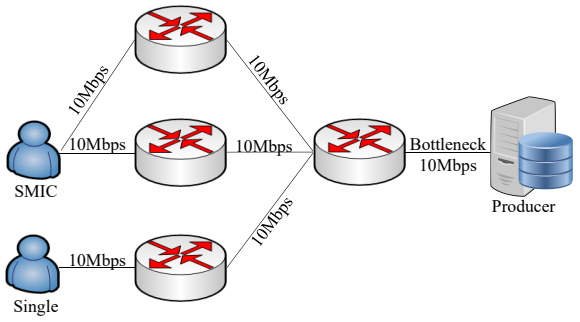
Table 1: Simulation Environments

Parameter	Description
Content request	Requests follow Zipf distribution, skewness parameter α = 1.0
Number of objects per prefix	10,000
Content object size	10MB
Content store size	100MB
Topology	Bottleneck 1-2, Tree, Competition
Congestion control schemes	ICP [3], MPTCP LIA [16], OLIA [11]

We suppose that users' requests all the objects follow the Zipf distribution, whose skewness is set to α = 1.0. New content request is generated when previous request is done. Requests of consumers are independent, so that multiple consumers might request for the same content object at the same time. Content objects of high ranks (i.e., more popular) are requested more frequently, and have more chances to be cached at routers. A content object is 10 MB long and transmitted in the unit of a data packet with 1040 bytes of payload; thus, a content object consists of 10,083 chunks. Each router has a tail-drop queue and content store of 100 MB; thus it can cache maximum 100,830 chunks. Note that a unit of a caching operation is a chunk, which is the payload size. In simulation, a name is used to request a content object, and consists of two parts: a prefix and a more specific content name. The latter is called a suffix for sake of exposition. For example, if a content name is `/whitehouse/trump.mp4/s1`, `/whitehouse` is a prefix, and `trump.mp4/s1` is a suffix. A producer is assumed



(a) The bottleneck topology 1 is shown, where three SMIC flows and a single-path flow share the bottleneck link.

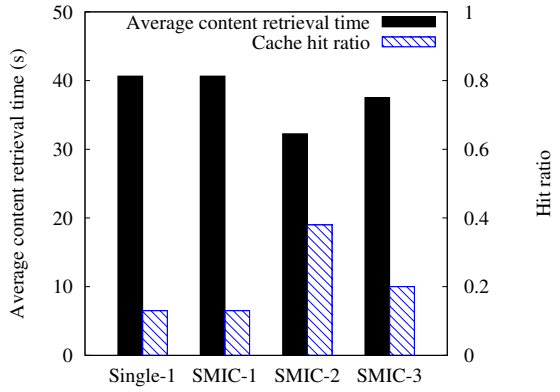


(b) The bottleneck topology 2 is shown, where the bottleneck link is shared by a single-path flow and a SMIC flow utilizing two subflows.

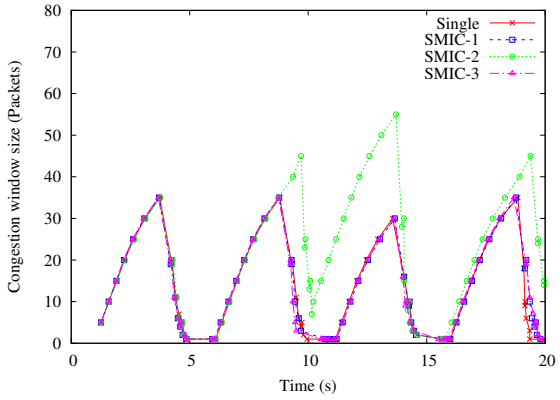
Figure 4: Two bottleneck topologies are illustrated, which show the single-flow friendliness of SMIC in the two bottleneck scenarios.

to store all the objects that start with a given prefix (i.e., the producer's prefix). If a consumer sends an Interest with the prefix and the suffix, it is forwarded to the producer who has advertised her prefix. Then the producer sends back the corresponding data packet to the consumer. Note that a producer may have its replication servers, who have advertised the same prefix at different places. Forwarded chunks are cached at routers by the caching policy of Least Recently Used. In detecting bottleneck sharing subflows, the RMSE threshold is set to 0.05s for timeout histories, and 80% for the difference in estimated bottleneck capacities.

We use four simulation topologies to evaluate SMIC. In Fig. 4, two bottleneck topologies are illustrated. With bottleneck topology 1 in Fig. 4(a), we will show the friendliness of SMIC with the coexistence scenario of multiple SMIC flows and a single-path flow. The single-path flow is assumed to use a congestion control mechanism that manages a single congestion window (i.e., ICP) over a single path. For convenience's sake, a single-window & single-path flow is called "single-path". With bottleneck topology 2 in Fig. 4(b), we will demonstrate that SMIC using multiple subflows fairly utilizes the bottleneck link when SMIC shares the bottleneck link with a single-path flow. With the tree topology in Fig. 7(a), we will explain that SMIC can utilize multiple subflows when there are multiple paths between the consumer and the content holder(s). Finally, with the competition topology in Fig. 7(b), we will show the performance of



(a) Average content retrieval times and cache hit ratio on bottleneck topology 1 are plotted.



(b) The congestion window sizes are plotted on the first bottleneck topology as time goes on.

Figure 5: Simulation results on bottleneck topology 1 are plotted. The results show that multiple SMIC flows support throughput-friendliness with a single-path flow.

SMIC when a SMIC flow shares the bottleneck links with multiple single-path flows, while varying the number of available paths. Also, we will compare SMIC flow with single-path flows using ICP and multi-path flows using MPTCP LIA or OLIA.

5.1 Friendliness with single-path flow

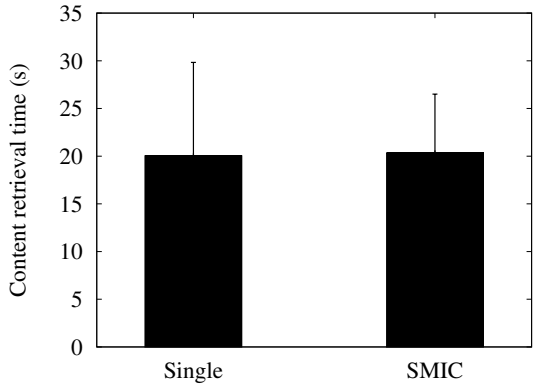
We first focus on the single-path friendliness of SMIC since SMIC might be deployed along with single-path congestion control mechanisms. When both SMIC and single-path flows exist in networks simultaneously, they may share a bottleneck. In such cases, SMIC flows should fairly share the bottleneck link with single-path flows. To evaluate such friendliness of SMIC, we set up two bottleneck topologies in Fig. 4. In bottleneck topology 1, three SMIC flows share a bottleneck with a single-path flow. Each consumer requests content objects through an access router. In bottleneck topology 2, a SMIC flow using two subflows shares a bottleneck with a single-path flow. A SMIC consumer requests content objects through two paths, while a single-path flow requests through another path. Then,

the requests from the SMIC consumer and single-path consumer go through the same bottleneck link. In both topologies, there is only one producer who has all the content objects starting with the prefix. Note that all Interests are forwarded to the producer as there is only one prefix in these scenarios. Propagation delays and link bandwidths are equal for all the links between the consumers and the producer.

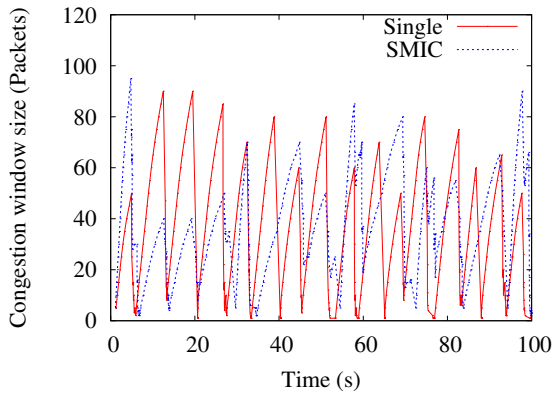
Fig. 5(a) shows the average content retrieval times and cache hit ratio of SMIC and single-path consumers on bottleneck topology 1 (Fig. 4(a)). The result shows that multiple SMIC flows have throughput-friendliness with a single-path flow overall even though there are slight differences in their performance metrics. The differences come from a caching effect of intermediate routers. During the simulation, consumers requested content objects with Zipf distribution, some of which might be cached at intermediate routers. SMIC-2 and SMIC-3 experience more cache hits than Single and SMIC-1; resulting in the difference between average content retrieval times.

Fig. 5(b) shows the change of the congestion window sizes of flows as time goes on in bottleneck topology 1 (Fig. 4(a)). For comparison purposes, all the flows start sending Interests at the same time. Three SMIC flows and a single-path flow increase their congestion windows while competing for the bottleneck link bandwidth. All the flows show approximately the similar tendency of increasing congestion windows, and notice that when timeouts occur, both SMIC and ICP decrease their window size in half. As the propagation delays of all the 4 flows are equal, we can conclude that SMIC flows and a single-path flow share the bottleneck link fairly.

Fig. 6 shows the average and maximum content retrieval times and congestion window sizes of the SMIC flow using two subflows and the single-path flow in bottleneck topology 2 (Fig. 4(b)). The results show the similar tendency to those of bottleneck topology 1. It means that the two subflows of the SMIC flow share the bottleneck link with the single-path flow fairly by adjusting their congestion windows according to Algorithm 1. We plot the change of the congestion window sizes of the SMIC and single-path flows in Fig. 6(b). At the start of the content retrieval, the SMIC flow increases its congestion window more aggressively than a single-path flow. However, the subflows sharing the bottleneck are detected after timeouts, and the window size is adjusted. Finally, as shown in Fig. 6(a), both the SMIC and single-path flows show similar completion times, which demonstrates that a SMIC flow using multiple subflows and a single-path flow can share the bottleneck link fairly.



(a) Average and maximum content retrieval times on bottleneck topology 2 are plotted as bars and lines, respectively.



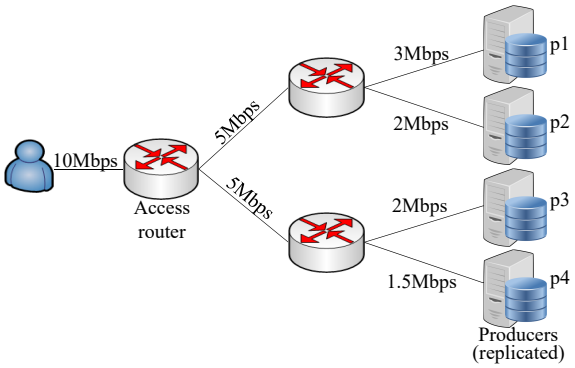
(b) The congestion window sizes are plotted on bottleneck topology 2 as time goes on.

Figure 6: Simulation results on bottleneck topology 2 are plotted. The results show that a SMIC flow utilizing multiple subflows supports throughput-friendliness with a single-path flow.

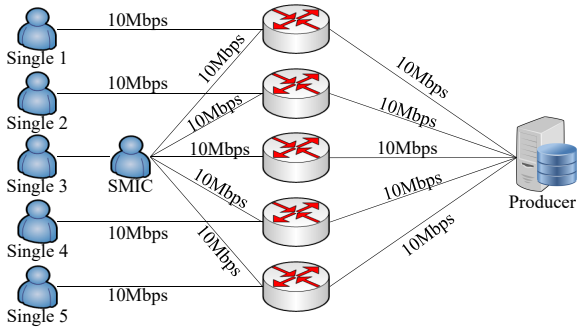
5.2 Exploiting available network resources

As SMIC utilizes multiple subflows, we expect that if there are multiple paths between a consumer and producers, a SMIC flow shows better performance than a single-path flow in terms of content retrieval times or the balance among the network links. To examine the performance metrics, we set up a 2-depth binary tree topology as shown in Fig. 7(a). The root of tree is an access router, and SMIC and single-path consumers are connected to the access router. They request content objects following the Zipf distribution. For the purposes of performance evaluation, we have four replication servers of a single producer. The replication servers are located at four leaves, and each server has different access link bandwidth. Note that these servers have advertised the same prefix, so that Interests can be forwarded to any of them.

Fig. 8(a) shows the average and maximum content retrieval times of SMIC, MPTCP LIA and single-path flows. The average content retrieval time of SMIC is about 11 seconds, and those of MPTCP and single-path are about 15.8 and 27 seconds, respectively. A SMIC consumer achieves about 2.4 times shorter completion time than a single-path consumer, while an MPTCP consumer achieves about 1.7 times shorter completion time than a single-path consumer. A single-path flow only uses the path of the highest capacity, and hence it sends Interests towards only the producer having the largest access link bandwidth. In case of SMIC and MPTCP flows, they exploit the other paths as well as the best path. However, they have different window control algorithms. MPTCP LIA increases its congestion window size conservatively for the worst case (i.e., all of subflows share a bottleneck link). On the other hand, the SMIC

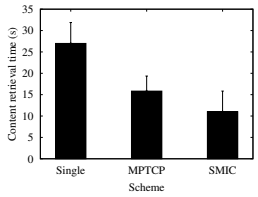


(a) The tree topology is illustrated. A consumer is connected to an access router and requests content objects. There are four replication servers of a producer; thus Interests can be forwarded to any of them.

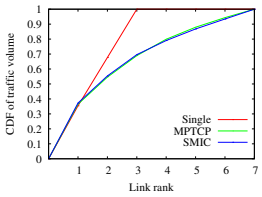


(b) The competition topology is illustrated. The multi-path consumer and single-path consumers request content objects to the producer. The number of available paths varies from two to five.

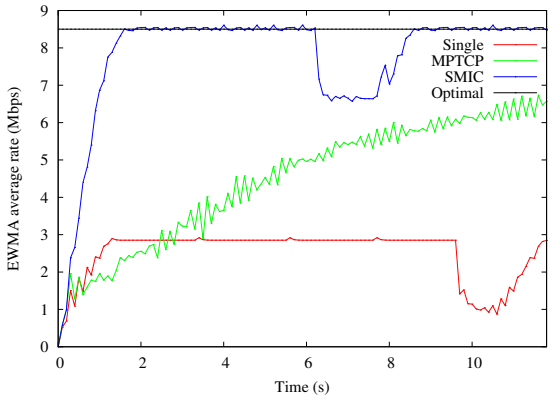
Figure 7: The tree and competition topologies are illustrated.



(a) Average and maximum content retrieval times are plotted by bars and lines, respectively.



(b) The CDF of traffic volume over links is shown.



(c) The consumer's Exponentially Weighted Moving Average (EWMA) throughput over time is shown.

Figure 8: Simulation results on the tree topology are plotted. The results show that SMIC can significantly reduce the content retrieval times and balance the traffic over network links.

flow increases its congestion window size just like the single-path flow does, and conservatively adjusts the congestion windows of the subflows that share the bottleneck link. Therefore, the SMIC flow achieves about 1.4 times shorter completion time. Fig. 8(c) supports above insights. A single-path flow shows fast convergence time, however, its throughput is bound at 3Mbps, the capacity of the best-path. MPTCP flow shows slow convergence time due to conservative window increase. SMIC flow shows fast convergence time which is similar to the single-path flow, and its throughput reaches 8.5Mbps which is the maximum (Optimal) throughput in the tree topology.

We also consider the balance of the traffic loads on the links in terms of the link utilization. Fig. 8(b) shows the CDF of link stresses in descending order of the traffic volume. A single-path flow shows the noticeable imbalance among the links. We notice that in case of the single-path flow, only three links of the best path handle all the traffic volume, and the other links do not experience network traffic. The reason for the differences between the link stresses of the top 3 links is the caching effect. By contrast, SMIC or MPTCP flows show balanced link utilization. They lower the link utilization of the 2nd and 3rd highest links by leveraging the other links. Note that the most-utilized link is the access link between the consumer and the access router.

5.3 Competition Scenarios

We have shown that SMIC has throughput-friendliness with single-path flows, and reduces the average content retrieval time of consumers while mitigating the imbalance of link utilization of networks. Next, we now evaluate a single SMIC flow with multiple single-path flows, where the subflows of the SMIC flow compete for network bandwidth with the single-path flows. For bandwidth competition scenarios, we use a competition topology in Fig. 7(b). A SMIC consumer requests content objects through multiple paths, while each single-path consumer requests through a single path. There is only one producer, who has all of content objects starting with the prefix. As there is only one prefix in this scenario, all of Interests are forwarded to the producer. Thus each subflow of the SMIC consumer competes for the bottleneck bandwidth with a single-path flow. Propagation delays and link bandwidths between consumers and the producer are equal for all the links. While only the case of five available paths for SMIC subflows and five single-path flows are shown in Fig. 7(b), we change the number of available paths and single-path flows from two to five in the experiments. For comparison purposes, we evaluate a MPTCP LIA or a MPTCP

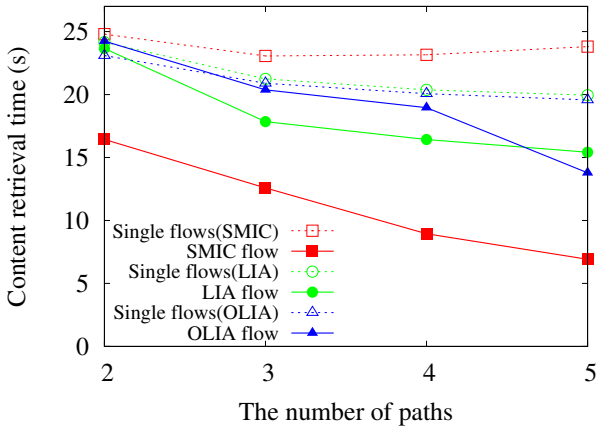


Figure 9: Completion times on the competition topology are plotted. SMIC shows the higher performance gain than the other MPTCP algorithms.

OLIA consumer as well as a SMIC consumer, and run simulations with the same settings.

Fig. 9 shows the average content retrieval times of SMIC, MPTCP LIA, MPTCP OLIA and single-path flows, as the number of available paths increases from 2 to 5. There are total six lines, which show the average completion time. The solid red line plots that of the SMIC flow, and the dotted red line shows that of the single-path flows competing with the SMIC flow. The solid green line and dotted green lines indicate the results of the MPTCP LIA flow and the single-path flows with the LIA flow, respectively. Similarly, the solid blue and dotted blue lines show the results of the MPTCP OLIA flow and the single-path flows, respectively. When the number of paths is two, only the SMIC flow shows outstanding content retrieval time of 16.4 seconds. Others show similar content retrieval times, about 24 seconds in average. It means the SMIC flow achieves about 46% faster content retrieval than the single-path flows. However, the LIA and OLIA flows achieve almost the similar performance to the single-path flows, respectively. This means that each of the two subflows in two MPTCP algorithms (LIA and OLIA) uses much less link bandwidth than the single-path flow due to the conservative window increase algorithm. As the number of paths increases, LIA and OLIA show performance improvements. However, SMIC shows the higher performance gain than LIA and OLIA (50%, 90%, 102% faster content retrieval for 3, 4, and 5 path cases, respectively).

6 DISCUSSION

As described in Section 2, there have been incremental approaches for ICN transport. Most of these studies proposed congestion control mechanisms based on window-based algorithm which are similar to the approaches of traditional IP network (especially, TCP and its variations). However, authors in [13] investigated rate-based alternatives for ICN because of differences between IP and ICN. In this section, we will compare SMIC (as a representative of window-based congestion control) and MIRCC (as a representative of rate-based congestion control) in ICN.

Table 2: Comparison between SMIC and MIRCC

	SMIC	MIRCC
Congestion control algorithm	Window-based	Rate-based
Additional roles of routers	Hash-modulo operations	1. Rate calculation 2. Path steering 3. Path identification
Additional fields in Interests	Path identifier	Path steering hint
Additional fields in Data packets	Trajectory identifier	1. Link rate $R(t)$ 2. PathId
Convergence time	Longer	Shorter

We summarize differences between SMIC and MIRCC in Table. 2. SMIC uses a window-based congestion control algorithm which is based on MPTCP. SMIC improves MPTCP algorithm in terms of window convergence time. By introducing shared bottleneck detection, SMIC can address conservative window increase algorithm of MPTCP. MIRCC uses a rate-based congestion control algorithm which is based on RCP. MIRCC improves RCP algorithm by getting rid of rate oscillation. It also holds two rate classes for supporting multi-path data transmission.

To realize SMIC, routers should be able to support hash-modulo operations, such as forwarding based on hash results or updating trajectory identifiers. In case of MIRCC, routers also should be able to perform path steering or path identification. In addition, MIRCC routers calculate their link rate $R(t)$. They compare $R(t)$ in Data packets and their own $R(t)$, and update Data packets' $R(t)$ with smaller value.

Additional fields in Interests are similarly used for SMIC and MIRCC. They hold path identifier or path steering hint in Interests header. Both of additional fields indicate which path Interests go through. However, MIRCC has more fields in Data packets than SMIC. Both of them include trajectory identifier or pathId, which are used for identifying paths. MIRCC, additionally, has the smallest link rate along the path, $R(t)$ in Data packets. Due to the $R(t)$ value, MIRCC can show extremely short convergence time because consumers can control their rate based on $R(t)$.

In short, to the best of our knowledge, SMIC needs less additional router operations or header fields, however, MIRCC shows shorter convergence time than SMIC at the cost of more operations or header fields. In our future works, we hope to compare SMIC with other ICN transport alternatives.

7 CONCLUSIONS

In this paper, we first analyze the design issues and solution spaces for congestion control and Interest forwarding in multi-path networking environments in ICN. Then we propose the Subflow-level Multi-path Interest Control (SMIC) scheme. SMIC can enhance the content retrieval performance by utilizing multiple subflows, which is realized by identifying different subflows and adjusting their congestion windows accordingly. For this, SMIC introduces "path identifier" and "trajectory identifier" to identify individual subflows, and devises a mechanism to figure out subflows sharing the same bottleneck. Using ndnSIM, we carry out simulations to show that SMIC flows has throughput-friendliness with (non-SMIC)

single-path flows, and SMIC reduces content retrieval times and mitigates the imbalance of link stresses.

ACKNOWLEDGEMENT

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2015-0-00567, Development of Access Technology Agnostic Next-Generation Networking Technology for Wired-Wireless Converged Networks) and 'The Cross-Ministry Giga KOREA Project' grant funded by the Korea government(MSIT) (No.GK18S0400, Research and Development of Open 5G Reference Model).

REFERENCES

[1] Afanasyev, A., Moiseenko, I., and Zhang, L. ndnsim: Ndn simulator for ns-3. Tech. rep., 2012.

[2] Allman, M., Paxson, V., and Blanton, E. Tcp congestion control. Tech. rep., 2009.

[3] Carofiglio, G., Gallo, M., and Muscariello, L. Icp: Design and evaluation of an interest control protocol for content-centric networking. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on* (2012), IEEE, pp. 304–309.

[4] Carofiglio, G., Gallo, M., and Muscariello, L. Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks. In *Proceedings of the second edition of the ICN workshop on Information-centric networking* (2012), ACM, pp. 37–42.

[5] Carofiglio, G., Gallo, M., Muscariello, L., and Papali, M. Multipath congestion control in content-centric networks. In *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on* (2013), IEEE, pp. 363–368.

[6] Carofiglio, G., Gallo, M., Muscariello, L., Papalini, M., and Wang, S. Optimal multipath congestion control and request forwarding in information-centric networks. In *Network Protocols (ICNP), 2013 21st IEEE International Conference on* (2013), IEEE, pp. 1–10.

[7] Chiocchetti, R., Perino, D., Carofiglio, G., Rossi, D., and Rossini, G. Inform: a dynamic interest forwarding mechanism for information centric networking. In *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking* (2013), ACM, pp. 9–14.

[8] Dukkipati, N. *Rate Control Protocol (RCP): Congestion control to make flows complete quickly.* PhD thesis, Stanford University, 2007.

[9] Ford, A., Raiciu, C., Handley, M., Barre, S., and Iyengar, J. Architectural guidelines for multipath tcp development, no. rfc 6182, 2011.

[10] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies* (2009), ACM, pp. 1–12.

[11] Khalili, R., Gast, N., Popovic, M., Upadhyay, U., and Le Boudec, J.-Y. Mptcp is not pareto-optimal: performance issues and a possible solution. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies* (2012), ACM, pp. 1–12.

[12] Lai, K., and Baker, M. Nettimer: A tool for measuring bottleneck link bandwidth. In *USITS* (2001), vol. 1, pp. 11–11.

[13] Mahdian, M., Arianfar, S., Gibson, J., and Oran, D. Mircc: Multipath-aware icn rate-based congestion control. In *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking* (2016), ACM, pp. 1–10.

[14] Moiseenko, I., and Oran, D. Path switching in content centric and named data networks. In *Proceedings of the 4th ACM Conference on Information-Centric Networking* (2017), ACM, pp. 66–76.

[15] Schneider, K., Yi, C., Zhang, B., and Zhang, L. A practical congestion control scheme for named data networking. In *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking* (2016), ACM, pp. 21–30.

[16] Wischik, D., Raiciu, C., Greenhalgh, A., and Handley, M. Design, implementation and evaluation of congestion control for multipath tcp. In *NSDI* (2011), vol. 11, pp. 8–8.

[17] Yi, C., Afanasyev, A., Wang, L., Zhang, B., and Zhang, L. Adaptive forwarding in named data networking. *ACM SIGCOMM computer communication review 42*, 3 (2012), 62–67.