A Study on How Coarse-grained Clock System Influences NDN Rate-based Congestion Control

Toshihiko Kato, Kazuki Osada, Ryo Yamamoto, and Satoshi Ohzahata
University of Electro-Communications
Tokyo, Japan
e-mail: kato@is.uec.ac.jp, osada@net.is.uec.ac.jp, ryo_yamamoto@is.uec.ac.jp, ohzahata@is.uec.ac.jp

Abstract—**Named Data Networking (NDN) is a widely adopted future Internet architecture that focuses on large scale content retrieval. The congestion control is one of the hot research topics in NDN, and the rate-based congestion control method is considered to be well suited. From the viewpoint of implementation, however, the rate-based method has an issue that it requires the fine-grained clock management. This is hard to implement in off-the-shelf computers. This paper evaluates the performance in the case that consumers use a coarse-grained clock system. We use the Stateful Forwarding as a target, which is a rate-based method proposed by the group proposing NDN. The simulation results show that a coarse-grained clock system increases congestion. This paper also proposes a smooth Interest sending scheme under a coarse-grained clock system, which relieves congestion.**

Keywords- NDN; Congestion Control; Rate Control; Clock Management.

I. INTRODUCTION

Named Data Networking (NDN) [1] is widely adopted as a platform for the future Internet architecture well suited for large scale content retrieval. The fundamental adopted in NDN is the name of required content, not the address of hosts containing the content. NDN uses two types of packets in all communications: Interest and Data. A consumer requesting a content sends an Interest packet containing the content name. A producer providing the corresponding content data returns a Data packet to the consumer. NDN routers transferring the Data packet cache the packet for future redistribution [2].

The congestion control is one of the hot research topics in NDN [3]. It is also a hot topic in TCP, but the mechanisms in TCP congestion control are limited to the congestion window management at end nodes [4], and the explicit congestion notification, which is recently introduced [5]. In contrast, the NDN congestion control introduces a variety of techniques. The receiver-driven window-based congestion control method is similar to that in TCP. Here, congestion is detected by timeout [6][7] or the congestion notification [8], and the window for Interest packets are managed heuristically, e.g., through an Additional Increase and Multiplicative Decrease (AIMD) mechanism. In NDN, the rate-based congestion control method is also studied actively. Here, a consumer and routers maintain a rate, by which Interest packets are transmitted contiguously. The rate is determined heuristically by use of congestion notification [9]-[11] or by the explicit rate reporting [12]-[14].

In NDN, the Round-Trip Time (RTT) between an Interest packet and the corresponding Data packet changes largely because of the Data packet caching at routers. The window-based congestion control method needs to determine a window size corresponding to the delay and bandwidth product, but the delay changes in NDN. Therefore, it is pointed that the rate-based method is more appropriate for NDN congestion control.

From the viewpoint of implementation, however, the rate-based congestion control method has some problems. Since the transmission speed in recent data links becomes high, such as 1 Gbps and 4 Gbps, the fine-grained clock management is required in the rate-based congestion control. For example, if the Data packet size is 10,000 bits and the link speed is 1 Gbps, the duration of one Data packet transmission is 10 micro seconds. It is supposed that higher precision clock will be required to control the Interest packet sending timing. On the other hand, the fine-grained clock management is hard to implement in off-the-shelf computers. TCP implementation uses 200 msec and 500 msec clocks for the delayed acknowledgement and retransmission, respectively [15]. So, it is considered that implementing rate-based mechanism with micro second order clock is extremely hard.

In this paper, we discuss how a coarse-grained clock system influences the NDN rate-based congestion control. We adopt the Stateful Forwarding [9] as a target system of evaluation, because it is implemented in ndnSIM, which is a widely used network simulator of NDN. Moreover, we propose a method to send Interest packets more smoothly even in the coarse-grained clock environment.

The rest of this paper is organized as follows. Section II explains the related work on NDN congestion control and discusses clock management. Section III describes the simulator base performance evaluation of the Stateful Forwarding over the coarse-grained clock system. Section IV gives our proposal of smooth Interest packet sending even if the coarse-grained clock management is used. In the end, Section V concludes this paper.

II. RELATED WORK

A. *Related work on NDN congestion control*

As described above, the congestion control methods in NDN are categorized as the window-based and the rate-based methods. The Interest Control Protocol (ICP) [6] and the Content Centric TCP (CCTCP) [7] are examples of the traditional TCP like window-based method, where a consumer sends Interest packets with the limitation of window size, and window size is changed according to the AIMD mechanism triggered by Data packet reception and congestion detected by timeout. The Chunk-switched Hop Pull Control

Protocol (CHoPCoP) [8] is another window-based method. It introduces explicit congestion notification with random early marking instead of timeout-based congestion detection, and the Interest sending control is done at a consumer with the window size changed by the AIMD mechanism. Although the window-based methods are simple, the window size itself may not be optimum when many Data packets are cached in different routers.

On the other hand, the rate-based methods are classified into the non-deterministic scheme, which uses the AIMD mechanism to determine the Interest sending rate, and the explicit rate notification scheme, in which intermediate routers report the optimum rate to a consumer. The Stateful Forwarding (SF) [9] is an example of the former scheme. SF introduces a negative acknowledgment (NACK) packet as a response to an Interest packet, which is generated when a router detects congestion. A consumer and a router manage the Interest sending rate locally by AIMD, and it decreases the rate when an NACK packet is received. The Stateful Forwarding with NACK suppressing [10] is a modification of SF. It resolves a problem that SF suffers from excessive rate reduction invoked by continuous NACK packets generated within one congestion event. The Practical Congestion Control (PCON) scheme [11] uses the CoDel active queue management scheme [16], which watches out the delay of packets in sending queues, to detecting congestion. When congestion is detected, a router signals this to consumers and downstream routers by explicitly marking Data packets. In respond to it, the alternative path forwarding or rate reducing is done by downstream routers or consumers, respectively.

In contrast with those non-deterministic methods, new methods have emerged that enable routers to report a maximum allowed Interest sending rate. In the Explicit Congestion Notification (ECN) based Interest sending rate control method proposed in [12], a consumer uses a minimum rate among the reported rates from all intermediate routers. In the Hop-By-Hop Interest Shaping (HoBHIS) [13], routers decide the maximum allowed Interest sending rate independently and accordingly shape Interest packet. The maximum allowed rate is also reported to a consumer and this allow a consumer to send Interest packets without invoking congestion. The Multipath-aware ICN Rate-based Congestion Control (MIRCC) [14] introduces a similar per-link Interest shaper at every router and rate reporting to consumer. It takes account of the case that a flow uses multipath transfer. In those methods, the maximum allowed rate is calculated from the parameters including link capacity and utilization, queue size, inflated Interest rate and average RTT. They are able to control Interest transmission so as to suppress congestion and to provide higher throughput compared with other rate-based methods.

B. *Discussions on clock management*

Although the rate-based congestion control methods are capable to provide better performance than the window-based method, they have implementation issues. In order to control the timing to send Interest packets, timers need to be implemented that expire when Interest packets are sent out. If the link speed is high and there are a lot of content retrieval flows, the timeout values of those timers become small and the timeout timing will be random. In order to implement those timers over off-the-shelf computers, the fine-grained clock mechanism and multiple timers realized by timer interrupt handler are required. However, they will introduce large processing overhead and reduce processing throughput drastically.

In order to avoid this problem, TCP protocol processing uses very rough clock mechanism, as described above. The Asynchronous Transfer Mode (ATM) [17], a legacy scheme standardized in the framework of broadband integrated services digital network, uses rate-based control for sending ATM cells. However, they do not use clock mechanism but adopt a way that null cells are inserted between cells with user data in order to pace user data cell flows.

Yamamoto [18] tackled a similar issue for high speed TCP data transfer. He pointed out that the TCP over Gigabit link requires the rate control as well as the window control but the clock-based rate control provides large processing overhead for terminals. So, he introduced pause packets over Gigabit Ethernet, corresponding to null cells in ATM, that are used only between end nodes and switching hubs. This approach can be adopted only over the dedicated link and cannot be applied to the shared media type link like high speed wireless LAN.

Kato and Bandai mentioned the similar issue on the processing overhead of fine-grained clock management for the rate-based congestion control, but they took an approach that exploits a hop-by-hop window control [19].

III. PERFORMANCE EVALUATION WITH COARSE-GRAINED CLOCK

Based on the discussions in Section II.B, we evaluate how the rate-based NDN congestion control works when the clock granularity is large. We adopt SF [9] as a target rate-base scheme because it is implemented by its proposer over ndnSIM version 1.0 [20], which uses C++ as a programming language. This section discusses the performance when the clock management becomes coarse-grained.

A. *Experimental configuration*

(1) Software implementation

Currently, ndnSIM has several versions; 1.0, and 2.0 through 2.4. Although SF is proposed by the research group who is maintaining ndnSIM, we believe that SF is implemented only in ndnSIM 1.0. Moreover, there are some bugs and problems in ndnSIM 1.0. For evaluating the influence by coarse-grained clock system, we added the followings to the current ndnSIM software.

- Support of AIMD like rate control

  SF mentions the rate control using AIMD as one possible candidate, but ndnSIM does not implement it. So, we have implemented it in the module managing Interest and Data packets (the `ForwardingStrategy` class) in the following way. The start value of Interest sending rate is given manually. When a router receives a Data packet, it increases the rate by one, under the limitation that it does not exceed the link speed at the outgoing interface. When receiving a NACK packet, it halves the current rate, under

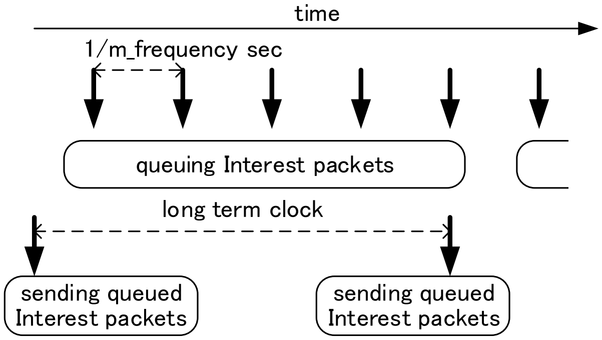


Figure 1. Implementation scheme of coarse-grained clock system.

the limitation that the minimum value of Interest sending rate is 1 packet/s.

It should be noted that the intermediate routers do not provide a shaping function that transmits Interest packets in a fixed rate. Instead, it provides a policing function that checks whether the Interest sending rate exceeds the limit or not. In order to handle a variable sending rate, the policing is performed by use of a leaky bucket.

- Use of constant bit rate (CBR) type consumer

ndnSIM 1.0 provides three types of consumers: rate based (the `ConsumerCbr` class), window based (the `ConsumerWindow` class) and batch type (the `ConsumerBatches` class). We decided to use the `ConsumerCbr` class and have added the AIMD like rate control on it. This class uses a protected static variable `m_frequency` as the Interest sending rate. We changed the variable in the same way described above in the `OnData()` and `OnNack()` methods, which are the methods called when a Data packet and an NACK packet is received, respectively.

- Emulation of coarse-grained clock system

In NDN, the rate control is implemented in the classes `Consumer` and `ConsumerCb`r; the `Consumer` class is the superclass of `ConsumerCbr`. The sending of Interest packets with a specific rate is implemented in the `ScheduleNextPacket()` method of the `ConsumerCbr` class. In this method, the `SendPacket()` method of the `Consumer` class is invoked periodically, every `1.0/m_frequency` seconds. The `SendPacket()` method sends one Interest packet actually.

We emulated a course-grained clock system in the `Consumer` class in the following way (see Figure 1).

- ➢ A clock system with longer tick, such as 100 ms, is implemented in the `Consumer` class. It calls itself periodically with the `Schedule()` method of the `Simulator` class.
- ➢ We also introduced a queue storing Interest packets temporarily. This queue is implemented using the `list` class.
- ➢ In the `SendPacket()` method, Interest packets are stored in the queue, instead of being sent actually.

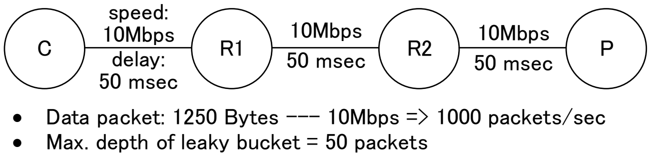


Figure 2. Network configuration and conditions.

- ➢ When the longer clock tick is invoked, all the queued Interest packets are transmitted actually.

(2) Experimental setting

We conducted the performance evaluation in the network configuration shown in Figure 2, which is a linear configuration where one consumer (C), two routers (R1 and R2), and one producer (P) are connected via 10 Mbps link with 50 msec propagation delay. The length of a Data packet is 1250 bytes, and the link speed corresponds 1,000 packets/sec. As described above, a consumer and routers maintain leaky bucket for policing the Interest packet flow. The arriving Interest packet is thrown into the leaky bucket conceptually, and, if the depth of the bucket becomes larger than the maximum value, a NACK packet is replied for the Interest packet. In our experiment, the maximum depth is set to 50 packets.

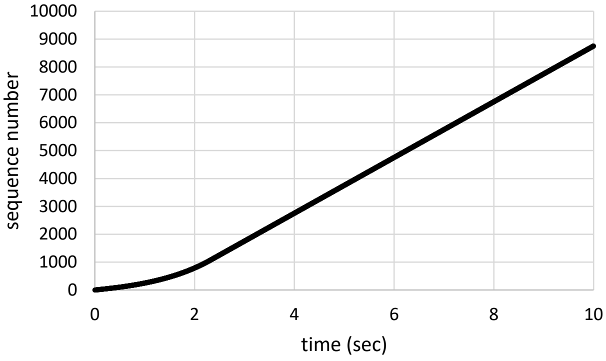
Under these conditions, we evaluated the cases that the coarse-grained clock is 50 msec, 100 msec, and 200 msec. In all the evaluation runs, the consumer starts from 200 packets/sec as the Interest sending rate. Each evaluation run takes 10 sec.

B. Perforamce evaluation results

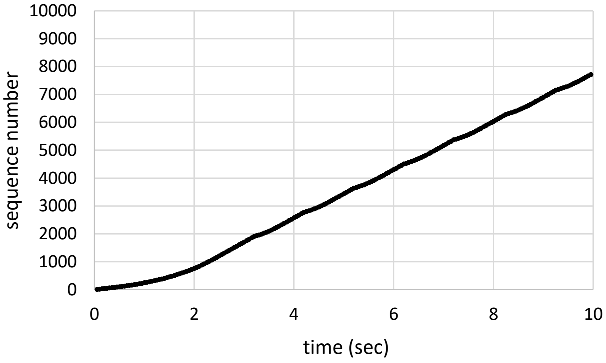
Figure 3 shows the time variation of the sequence number contained in the name of requested content. It corresponds to the number of content request in a content retrieval flow. Each value is plotted when the corresponding Interest packet is sent. Figure 4 shows the time variation of the Interest sending rate at the consumer. In this figure, each value is plotted when the consumer receives a Data or NACK packet and it changes the value of Interest sending rate.

Figures 3 (a) and 4 (a) show the results of the original SF implementation. The sequence number is increasing steadily. The Interest sending rate starts from 200 packets/sec and goes to 1,000 packets/sec, the maximum value corresponding to the link speed. These results show that the rate-based congestion control works well.

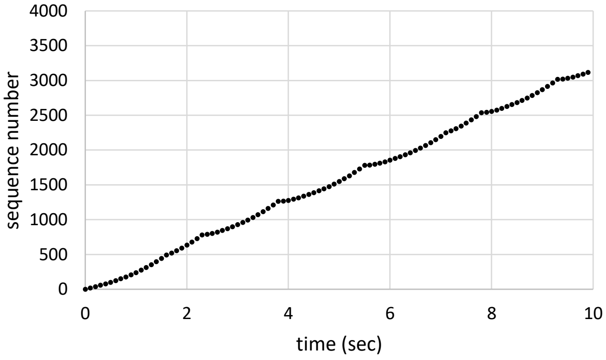
Figures 3 (b) and 4 (b) show the results when the coarse-grained clock system is used and the clock tick is 50 msec. The sequence number is also increasing steadily, but there are several drops in the Interest sending rate. The rate starts from 200 packets/sec and goes to 1,000 packets/sec, but it drops to 500 packets/sec at 3.2 sec. This is triggered by a NACK packet generated locally inside the consumer. That is, the consumer also maintains the leaky bucket for policing the Interest packet flow. When the Interest sending rate is 1,000 packets/sec and the clock tick is 50 msec, fifty Interest packets are generated in one moment by the application, and rush into the leaky bucket. Since the maximum depth of the bucket is 50 packets, all of them are stored in the bucket and leaked in



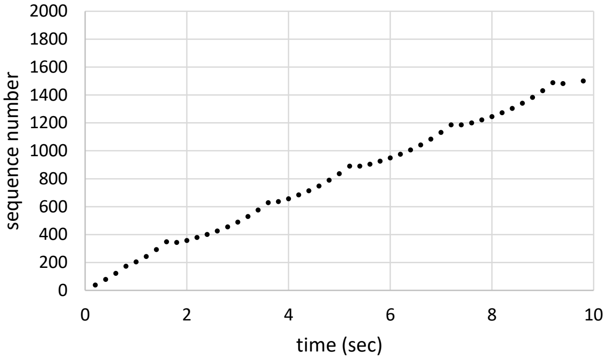
(a) Fine-grained clock



(b) Coarse-grained clock (tick = 50 msec)

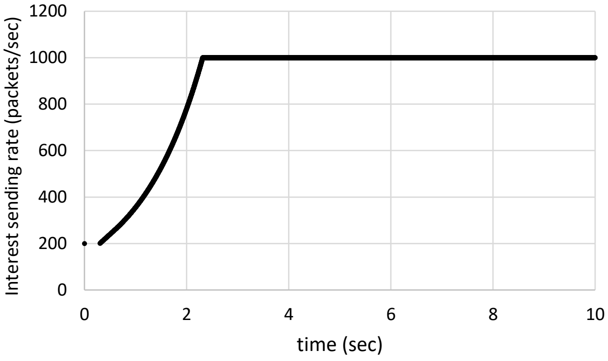


(c) Coarse-grained clock (tick = 100 msec)

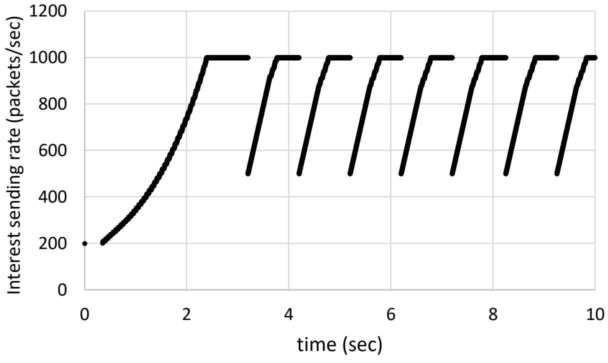


(d) Coarse-grained clock (tick = 200 msec)

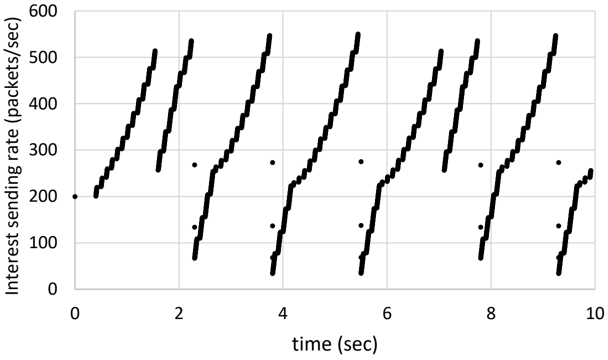
Figure 3. Time variation of Interest sequence number.



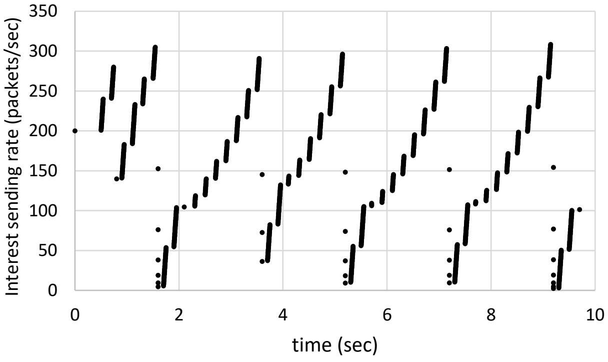
(a) Fine-grained clock



(b) Coarse-grained clock (tick = 50 msec)



(c) Coarse-grained clock (tick = 100 msec)



(d) Coarse-grained clock (tick = 200 msec)

Figure 4. Time variation of Interest sending rate

TABLE I. SUMMARY OF RESULTS WITH COARSE-GRAINED CLOCK.

	Original	Tick = 50 msec	Tick = 100 msec	Tick = 200 msec
Data packet throughput (Mbps)	8.75	7.72	3.12	1.50
Number of NACK packets	0	7	20	27

1,000 packets/sec (actually they are transmitted to R1 in a line speed). But in some timing, fifty Interest packets are generated in the situation that there are some packets remaining in the bucket. Then, a NACK packet is generated.

Figures 3 (c) and 4 (c) and Figures 3 (d) and 4 (d) show the results when the clock tick is 100 msec and 200 msec, respectively. In these cases, the increase of the sequence number is suppressed, and the Interest sending rate is limited up to 600 and 300 packets/sec, respectively. This is because the number of Interest packets transmitted back to back is increasing. These results show that, when the clock tick becomes large in the coarse-grained clock system, the rate-based congestion control does not work correctly.

Table I gives a summary of the results. The Data packet throughput is the total content size transferred during an evaluation run divided by ten seconds. In the case of the fine-grained clock (Original in the table), the throughput is 8.75 Mbps and there are no NACK packets transferred. In the case of the coarse-grained clock with 50 msec tick, the Data packet throughput decreases slightly, because the rate goes to 1,000 packets/sec and there are no contiguous NACK receiving. However, the cases with 100 msec tick and 200 msec tick, the number of NACK packets increases and the Data packet throughput decreases largely.

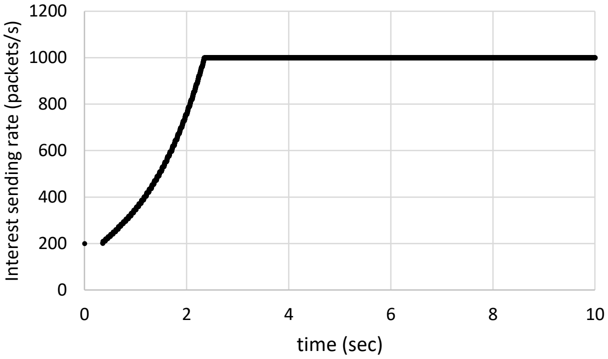
IV. PROPOSAL TO SMOOTHEN INTERST PACKET SENDING

A. *Proposed method*

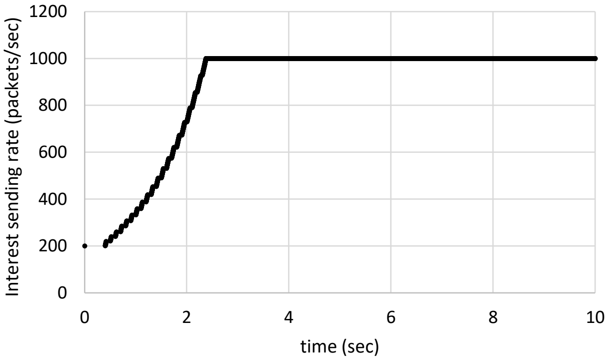
In the SF mechanism with the coarse-grained clock system described in Section III, we supposed that Interest packets are transmitted only in response to clock ticks. As a result, Interest packets were sent in a burst and this triggered the overflow in a leaky bucket.

Here, we propose an Interest control method that utilizes the Data and NACK packet receiving timing. When a consumer receives a Data or an NACK packet, the receiving processing is triggered by a hardware interrupt mechanism, and it does not give large overhead to computers, different from the software based timeout mechanism. So, the receiving timing is a good chance to proceed the Interest packet sending. So, we have added the following mechanism in the coarse-grained clock system described in Section III.A.(1).

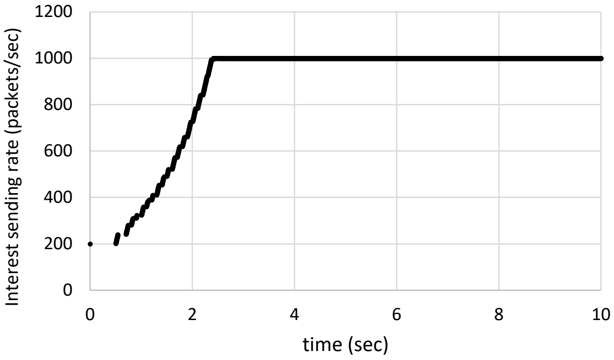
- When a consumer receives a Data or an NACK packet, it processes the received packet and then tries to send the Interest packets stored in the Interest queue.
- This procedure is implemented in the `OnData()` and `OnNack()` methods in the `Consumer` class.



(a) Coarse-grained clock (tick = 50 msec)



(b) Coarse-grained clock (tick = 100 msec)



(c) Coarse-grained clock (tick = 200 msec)

Figure 5. Time variation of Interest sending rate in proposed method

B. *Perforamce evaluation results*

We have conducted the performance evaluation of the proposed method in the same configuration and conditions as the previous section. Figure 5 shows the time variation of the Interest sending rate at the consumer implementing the proposed method.

Different from the results given in Figure 4, all the cases when the clock tick is 50 msec, 100 msec, and 200 msec give the similar results with the fine-grained clock system. That is, the Interest sending rate starts from 200 packets/sec, goes to 1,000 packets/sec straightly, and keeps in this level. This means that there are no NACK packets generated. These

TABLE II. SUMMARY OF RESULTS WITH PROPOSED METHOD.

	Tick = 50 msec	Tick = 100 msec	Tick = 200 msec
Data packet throughput (Mbps)	8.73	8.70	8.69
Number of NACK packets	0	0	0

results mean that the proposed method is effective for smoothening the bursty Interest packet sending caused by the coarse-grained clock system.

Table II shows a summary of the results. There are no NACK packets in all the cases of three clock tick values. The Data throughput are also similar for three cases, and the value is close to that of the fine-grained clock based SF.

V. CONCLUSIONS

This paper described how the coarse-grained clock system influences the NDN rate-based congestion control. Currently, the rate-based congestion control is considered to be effective in NDN. However, the rate-based control over high speed links requires highly precious clock management and this gives a serious processing overhead to off-the-shelf computers. So, we think that commodity based consumers need to use a coarse-grained clock system.

Even if the network did not cause any congestion, the clock ticks 50 msec, 100 msec, and 200 msec generated some NACK packets. Especially, in the cases of 100 msec and 200 msec ticks, the Data throughput decreases largely. These results mean the NDN rate-based congestion control has some problem when it is used with a coarse-grained clock system.

This paper also proposed a scheme to smoothen Interest sending, which allows a queued Interest packets for sending to be transmitted when any Data or NACK packets are received. As the result of simulation evaluation, the proposed method did not generate any NACK packets even if 50 msec, 100 msec, and 200 msec are used as clock ticks.

This paper uses a relatively large tick value, but smaller tick values can be used in actual computers. Besides, this paper uses a simple network configuration with a relatively slow link speed. So, we need to evaluate the performance in a realistic computer / network condition.

REFERENCES

[1] V. Jacobson, et al., “Networking Named Content,” Proc. of CoNEXT ’09, pp. 1-12, Dec. 2009.

[2] N. Minh, R. Yamamoto, S. Ohzahata, and T. Kato, “A Routing Protocol Proposal for NDN Based Ad Hoc Networks Combining Proactive and Reactive Routing Mechanism,” Proc. of IARIA AICT 2017, pp. 80-86, Jun. 2017.

[3] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, and B. Zhang, “Congestion control in named data networking - A survey,” Computer Communications, vol. 86, pp. 1-11, Jul. 2016.

[4] A. Afanasyev, et al., “Host-to-Host Congestion Control for TCP,” IEEE Commun. Surveys & Tutorials, vol. 12, no. 3, pp. 304-342, 2010.

[5] K. Ramakrishnan, S. Floyd, and D. Black, “The Addition of Explicit Congestion Notification (ECN) to IP,” IETF RFC 3168, Sep. 2001.

[6] G. Carofiglio, M. Gallo, and L. Muscariello, “ICP: Design and Evaluation of an Interest Control Protocol for Content-Centric Networking,” Proc. of IEEE INFOCOM 2012, pp. 304-309, Mar. 2012.

[7] L. Saino, C. Cocora, and G. Pavlou, “CCTCP: A Scalable Receiver-driven Congestion Control Protocol for Content Centric Networking,” Proc. of IEEE ICC 2013, pp. 3775-3780, Jun. 2013.

[8] F. Zhang, Y. Zhang, A. Reznik, H. Liu, C. Qian, and C. Xu, “A Transport Protocol for Content-Centric Networking with Explicit Congestion Control,” Proc. of IEEE ICCCN 2014, pp. 1-8, Aug. 2014.

[9] Y. Cheng, A. Afanasyev, I. Moiseenko, B. Zhang, L. Wang, and L. Zhang, “A case for stateful forwarding plane,” Computer Communications, vol. 36, no. 7, pp. 779-791, Apr. 2013.

[10] T. Kato and M. Bandai, “Congestion Control Avoiding Excessive Rate Reduction in Named Data Network,” Proc. of IEEE CCNC, pp. 1-6, Jan. 2017.

[11] K. Schneider, C. Yi, B. Zhang, and L. Zhang, “A Practical Congestion Control Scheme for Named Data Networking,” Proc. of ACM ICN 2016, pp. 21-30, Sep. 2016.

[12] J. Zhang, Q. Wu, Z. Li, M. A. Kaafar, and G. Xie, “A Proactive Transport Mechanism with Explicit Congestion Notification for NDN,'” Proc. of IEEE ICC 2015, pp. 5242-5247, Jun. 2015.

[13] N. Rozhnova and S. Fdida, “An extended Hop-by-Hop Interest shaping mechanism for Content-Centric Networking,” Proc. of IEEE GLOBECOM 2014, pp. 1-7, Dec. 2014.

[14] M. Mahdian, S. Arianfar, J. Gibson, and D. Oran, “Multipath-aware ICN Rate-based Congestion Control,” Proc. of ACM ICN 2016, pp. 1-10, Sep. 2016.

[15] K. Fall and W. Stevens, “TCP/IP Illustrated, Volume1; The Protocols, Second Edition,” Addison-Wesley,

[16] K. Nichols and V. Jacobson, “Controlling Queue Delay,” ACM Magazine Queue, vol. 10, issue 5, pp. 1-15, May 2012.

[17] ITU-T, “B-ISDN asynchronous transfer mode functional characteristics,” Series I: Integrated Services Digital Nework, Recommendation I.150, Feb. 1999.

[18] Y. Yamamoto, “Estimation of the advanced TCP/IP algorithms for long sistance collaboration,” Fusion Engineering and Design, vol. 83, issue 2-3, pp. 516-519, Apr. 2008.

[19] T. Kato and M. Bandai, “A Congestion Control Method for NDN Using Hop-by-hop Window Management,” Proc. of IEEE CCNC 2018, pp. 1-6, Jan. 2018.

[20] A. Afanasyev, I. Moiseenko, and L. Zhang, “ndnSIM: NDN simulator for NS-3,” NDN, Technical Report NDN-0005, 2012, Oct. 2012.