Digital Communications and Networks 8 (2022) 1085–1093

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

# An accurate retransmission timeout estimator for content-centric networking based on the Jacobson algorithm

Mortaza Nikzad [a,**], Kamal Jamshidi [b,*], Ali Bohlooli [b], Faiz Mohammad Faqiry [a]

[a] Faculty of Computer Science, Kabul Polytechnic University, Kabul, Afghanistan
[b] Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran

ARTICLE INFO

*Keywords:*
Content-centric networking
Retransmission timeout
Popularity distribution gain
Jacobson RTO estimator

ABSTRACT

Accurately estimating of Retransmission TimeOut (RTO) in Content-Centric Networking (CCN) is crucial for efficient rate control in end nodes and effective interface ranking in intermediate routers. Toward this end, the Jacobson algorithm, which is an Exponentially Weighted Moving Average (EWMA) on the Round Trip Time (RTT) of previous packets, is a promising scheme. Assigning the lower bound to RTO, determining how an EWMA rapidly adapts to changes, and setting the multiplier of variance RTT have the most impact on the accuracy of this estimator for which several evaluations have been performed to set them in Transmission Control Protocol/Internet Protocol (TCP/IP) networks. However, the performance of this estimator in CCN has not been explored yet, despite CCN having a significant architectural difference with TCP/IP networks. In this study, two new metrics for assessing the performance of RTO estimators in CCN are defined and the performance of the Jacobson algorithm in CCN is evaluated. This evaluation is performed by varying the minimum RTO, EWMA parameters, and multiplier of variance RTT against different content popularity distribution gains. The obtained results are used to reconsider the Jacobson algorithm for accurately estimating RTO in CCN. Comparing the performance of the reconsidered Jacobson estimator with the existing solutions shows that it can estimate RTO simply and more accurately without any additional information or computation overhead.

## 1. Introduction

Content-Centric Networking (CCN) is a new networking architecture proposed to tackle the mismatch between new application requirements and current Transmission Control Protocol/Internet Protocol (TCP/IP) network.

Today's Internet is designed for sharing scarce hardware and software resources in an end-to-end communication paradigm, in which new application requirements such as mobility support, scalability, and secure data distribution are unpredictable. CCN is a fundamental architectural change that evolves Internet architecture away from host-centric communication. It converts the matter of communication to what the user wants irrespective of its origin [1].

In CCN, intermediate routers have extra storage to cache the most popular contents and consumer nodes requests for content, which they designate by issuing interest packets that include the unique name of each piece of requested content to the network. At that point, the most efficient provider, which can be a router or an original server, will satisfy the request. Moreover, intermediate routers aggregate requests for the same content and multicast that content to all requesters when the content is received. Therefore, dynamic multipath forwarding, multicasting, and caching popular contents in intermediate routers increase network performance by decreasing delays and network loads [2,3].

CCN is in the early stages of development, and there are several problems regarding different parts of this architecture [4]. One of the most challenging research fields in CCN is transport control [5–7]. It presents such a challenge because the architecture no longer possesses continuous streams between two end systems however, the congestion and flow control mechanisms in the transport layer are generally implemented using the estimation of end-to-end network path properties [8–10].

To address this challenge, schemes have been proposed that can be categorized as either the user-side receiver-driven transport control group or hop-by-hop interest shaping group [5–7,10–12]. In the first

* Corresponding author.
** Corresponding author.
*E-mail addresses:* m.nikzad@kpu.edu.af (M. Nikzad), jamshidi@eng.ui.ac.ir (K. Jamshidi), bohlooli@eng.ui.ac.ir (A. Bohlooli), faiz.faqiry@kpu.edu.af (F.M. Faqiry).

https://doi.org/10.1016/j.dcan.2022.03.006
Received 1 February 2020; Received in revised form 2 February 2022; Accepted 4 March 2022
Available online 7 March 2022

approach, TCP is redesigned for using in CCN, whereas the second one uses routers' multipath forwarding capability for transport control.

Although a broad range of transport control schemes for CCN has not yet been explored, proposals approaching TCP have many proponents among researchers. Published literature presents two main reasons for justifying the researchers' favor. First, TCP comes with well-established effective algorithms that are well understood and have survived rigorous testing. Second, it is foreseeable that CCN will finally be deployed in a mixed environment with TCP. This provides sufficient urgency for designing TCP-like transport control in CCN because both CCN and TCP could exhibit compatible behaviours and similar rate-based dynamics [6].

Transport control in TCP is a timeout-driven Additive-Increase Multiplicative-Decrease (AIMD)-based rate control, pioneered by Jacobson [13]. This protocol is based on the estimation of Retransmission TimeOut (RTO) using an Exponentially Weighted Moving Average (EWMA) on the Round Trip Time (RTT) of previously received packets and their variations.

In such a protocol, estimating RTO accurately is crucial for using network resources efficiently. This is because, if the RTO is set to a value that is less than the time required to receive an ACK of a sent packet, the packet will be retransmitted prematurely and the sender is forced to decrease its send rate unnecessarily.

Moreover, a conservative RTO policy setting large values will also decrease throughput because lost packets will not be retransmitted until their RTOs expire, rendering networks sluggish and inefficient [14].

Estimating RTO accurately in CCN is more crucial than TCP/IP because, in this architecture, intermediate routers set the retry timers and rank the potential interfaces that can reach data according to estimated RTO [5,6,15,16].

In TCP/IP, the Jacobson algorithm [13] is used to estimate the RTO, and several analyses have confirmed its efficiency in this architecture [14,17]. However, in-network caching, request aggregation, and dynamic forwarding of interest packets in CCN make its performance in this architecture controversial [5,6,16,18].

End-to-end communication sessions do not exist in CCN, allowing a content to be received from different providers. Therefore, delays in receiving requested content in CCN changes over time, depending on the content popularity.

Research has shown that content popularity in modern day traffic follows a Zipf-Mandelbrote distribution with gain $s$ [19–21]. In CCN, $s$ determines the performance of in-network caching; thus, providing it substantial influence on network response time patterns. Increase in $s$ has been shown to coincide with request rate increase for the most popular content, causing higher hit ratios for intermediate routers that, in turn, fulfill requests in fewer hops over shorter period. As a result, network path properties and content popularity distribution gains in CCN affect packet's RTT and RTO estimation accuracy considerably [22].

The inefficiency of the Jacobson algorithm in CCN is demonstrated in several research papers [5,6,16,18], yet few schemes have been proposed to address this problem [23,24]. The main contribution of these schemes is maintaining one estimator per source rather than one estimator for each communication session between clients and servers. These schemes have not considered the request aggregation and dynamic multipath forwarding of contents in intermediate routers. For this reason, they do not estimate RTO better than the Jacobson estimator. Moreover, they have high implementation complexity and computation overhead.

As a result, the Jacobson algorithm is the most promising applicable scheme for estimating RTO in CCN, which has led to its widespread use in current CCN implementation [25,26] and proposed transport control schemes for CCN [5,7,16].

The accuracy of the Jacobson algorithm depends highly upon parameter settings, and several experimental evaluations have assigned them in TCP [14,27,28]. However, the performance of Jacobson algorithm in CCN with a significant architectural difference has not yet been evaluated, and the estimator's proper setting in CCN remains unknown. This paper is the first work on evaluating and reconsidering the Jacobson algorithm for discovering and improving its performance in CCN.

In this paper, a large-scale inclusive simulation analysis is extended to assess the impact of in-network caching, request aggregation, and multipaths forwarding in CCN on the accuracy of the Jacobson estimator. We define two specific performance metrics to calculate the estimator's accuracy and evaluate its performance in CCN by varying the minimum RTO, EWMA parameters, and RTT variance multipliers against different popularity distribution gains.

These analyses are verified using comparative analysis of consumer download rates which indicate that setting the minimum RTO close to average RTT as well as smoother estimations of RTT and RTT variance can roughly double the estimator accuracy. We reconsider the Jacobson RTO estimator based on these points and compare its performance with two other previously proposed estimators in CCN [23,24], showing that the former has higher accuracy without additional information or computation overhead.

The rest of paper is organized as follows. In the following sections, we review related works on RTO estimation in CCN. In Section 3, we introduce the system model, including the simulation configuration and performance metrics used for assessing the estimators. The evaluation results are provided and discussed in Section 4. In Section 5, our reconsidered estimator is proposed and compared to previous schemes. Section 6 summarizes our paper and presents our future work plans.

## 2. Related works

The introduction mentions that the Jacobson algorithm is the most promising approach to estimate RTO in CCN. In this estimator, a consumer receiving a data packet causes the RTO to update as follows:

$$RTO = SRTT + k \times SRTTVAR \tag{1}$$

where $k$ is a constant, $SRTT$ is the smoothed RTT estimate, and $SRTTVAR$ is the smoothed estimate of RTT variance. $k$ was equal to 2 at first [13] and was altered to 4 in the revised version of the paper [29]. Prior to the first measurement, $RTO = 3$ s. $SRTT$ and $SRTTVAR$ are calculated and updated using an EWMA with a gain of $\alpha$ for $SRTT$ and $\beta$ for $SRTTVAR$

$$SRTT = (1-\alpha)SRTT + \alpha RTT_{last} \tag{2}$$

and

$$SRTTVAR = (1-\beta)SRTTVAR + \beta RTTVAR_{last} \tag{3}$$

where $RTT_{last}$ is the RTT for the last received packet and $RTTVAR_{last} = SRTT - RTT_{last}$. In addition, Jacobson recommends $\alpha = \frac{1}{8}$ and $\beta = \frac{1}{4}$ as EWMA averaging gain in (2) and (3). He also determined that initial values of $SRTT$ and $SRTTVAR$ are equal to $RTT_{last}$ (first packet's RTT) and $\frac{1}{2} \times RTT_{last}$, respectively. Furthermore, the value of RTO is bounded by $RTO_{min}$ and $RTO_{max}$, where for common TCP implementation, $RTO_{max} =$ 64 Sec and $RTO_{min} = 200$ ms [30].

According to (3), the values of $RTO_{min}$, $\alpha$, $\beta$, and $k$ have the most impact on the estimated RTO value and must be set carefully to attain an equilibrium point for RTO. In fact, $RTO_{min}$ specifies the low RTO boundary, $\alpha$ and $\beta$ control how rapidly $SRTT$ and $SRTTVAR$ adapt to changes, and $k$ represents the confidence about the small probability that the RTT will exceed the RTO.

The Jacobson algorithm works well in TCP because all packets in a TCP session are received through the same provider. However, in CCN, data can be received from different providers. In addition, consumers may change their original data producer during content transfer because of provider unavailability or discovering a more effective provider. This change in provider can cause drastic RTT fluctuations that impair the Jacobson algorithm's accuracy in this architecture [5,6,16,18]. In the

following, we briefly review the schemes proposed for overcoming this challenge in CCN.

To achieve an accurate RTO estimation in CCN, Carafiglio et al. [23] proposed a weighted averaging estimator for their multipaths congestion control scheme. Their estimator calculates RTO by assigning each provider (intermediate routers or original servers) a unique identifier that is placed into a data packet when an interest packet is satisfied. Consumers then maintain a separate estimator for each provider, updating them based on the appended identifier in the received data packets. The overall round trip delay for a new interest is finally calculated by weighted averaging over all calculated RTO for providers. In this approach, providers' estimators are weighted by either the number of received chunks from each provider or the freshness of the measurement.

Estimator accuracy in this scheme is improved by providing an estimator to each provider rather than for each communication session. However, estimator accuracy remains impaired by request aggregation in intermediate routers and the intrinsic multipath dynamic forwarding of interest packets. Our analyses in Section 6 show that this scheme does not provide better RTO estimation than the Jacobson algorithm.

Maintaining one estimator per source was also adopted by Ref. [24]. They proposed an anticipated interest mechanism to predict the provider of an interest packet and then reliably estimated RTO using the provider's specific estimator. In this method, to predict the provider of an interest packet, consumer nodes attach a list of their next times requests for data chunks to their current interest packets. Thus, whenever an intermediate router receives an interest packet, it is able to check the appended list of subsequent requests while also appending its node identifier to any processed interest packet along with identifiers for stored and listed data chunks and their associated timestamps $T_I$.

Upon forwarding an interest packet, a router must not alter the information appended during previous hops. If a router holds a requested chunk in its cache, it returns the corresponding data packet, appending the anticipated chunks header retrieved from the interest packet.

In the return path of the data packet, each router processes the anticipated chunk header. Should there be appended data in the header, $T_I$ is replaced with a timestamp indicating the difference between $T_I$ and $T_D$ on receiving the data packet.

Using the information contained within a data packet, a consumer can estimate the RTT between itself and router $r$, which holds an anticipated chunk header as follows:

$$RTT(r) = (T_D - T_I) - (T_D(r) - T_I(r)) \tag{4}$$

In fact, they also maintain an estimator for each provider but used a prediction method to determine which estimator should be used for the next interest, rather than weighted averaging on all estimators.

Although the algorithm in Ref. [24] is more robust than that in Ref. [23], it has considerable complexity in its design and implementation, imposing a larger information and computation overhead on both consumers and intermediate routers. Moreover, content provider predictions could be frequently incorrect due to the dynamic forwarding of interest packets or aggregation of requests in intermediate routers.

## 3. System model

### 3.1. Performance metrics

In this study, we assess the Jacobson algorithm to understand how well it works in CCN and how to reconsider it to estimate RTO accurately in this architecture. To this end, metrics proposed by Ref. [14] are redefined and used. These metrics show the mean proportion of premature retransmission of an interest packet due to early timeout and the mean normalized timeout cost in case of necessary timeout. They are described using the proposed notation in Table 1.

We let $b_i$ and $g_i$ be the total number of unnecessary (bad) and necessary (good) timeouts experienced by consumer $i$. We then define

**Table 1**
Notations.

| Variable | Description |
|---|---|
| $b_i$ | Total number of unnecessary timeouts in consumer $i$ |
| $g_i$ | Total number of necessary timeouts in consumer $i$ |
| $\rho_i$ | Normalized number of unnecessary timeout in consumer $i$ |
| $RTO^j_{c,i}$ | $j$ the necessary timeout for content $c$ in consumer $i$ |
| $RTT^j_{c,i}$ | Most recently RTT for content $c$ in consumer $i$ |
| $\omega^j_{c,i}$ | Cost of the timeout in unit of RTTs in consumer $i$ |
| $\varphi_i$ | Average normalized timeout cost in consumer $i$ |
| $W$ | Mean normalized timeout cost |
| $B$ | Mean proportion of unnecessary timeout |

$\rho_i = \frac{b_i}{(b_i+g_i)}$ as the normalized number of bad timeouts for consumer $i$. Moreover, if $RTO^j_{c,i}$ is the RTO estimation for $j$, the good timeout of content $c$ for consumer $i$, and $RTT^j_{c,i}$ is the most recent RTT for content $c$ for consumer $i$, then $\omega^j_{c,i} = \frac{RTO^j_{c,i}}{RTT^j_{c,i}}$ is the cost of the timeout in RTT units, and we can define $\varphi_i = E_i[\omega^j_{c,i}]$ for all good events in a transmission flow as an average, normalized timeout cost for consumer $i$.

For all content flows, we define $B = E_i[\rho_i]$ as the mean proportion of bad timeouts and $W = E_i[\varphi_i]$ as the mean normalized timeout cost, where $B$ reflects how well an estimator avoids from bad timeouts and $W$ corresponds to the overall estimator performance. For the calculation of $B$ and $W$, only the first timeout of an interest packet is considered so the multiplied RTOs after the first retransmission are neglected.

In addition to $W$ and $B$, the growth rate of a consumer's download rate is used as evidence of estimator efficiency when growth rates are compared. As previously stated, premature retransmissions decrease consumer send rates unnecessarily, and late retransmissions make the network sluggish. Therefore, a more accurate estimator will increase the consumer download rates.

### 3.2. Simulation configuration

For a comprehensive analysis that produces reliable and inclusive results, it is necessary to consider various CCN properties, such as caching, request aggregation, and dynamic multipaths forwarding, in simulation scenarios. For this reason, we extend a large-scale simulation analysis using the implementation of Sprint topology [31] (Fig. 1) in an NS3-based Named-Data Networking (NDN) simulator named ndnSIM [32].

We assume 10,000 pieces of content in this network, which consumers request according to the Zipf-Mandelbrote distribution. The gain of distribution, $s$ ranges from 1 to 2. Based on the Zipf-Mandelbrote distribution, with $s = 1$, approximately 75% of requests are for the top 20% popular contents, whereas with $s = 2$, the request rate for these contents is 99% [21].

All intermediate routers have a 100 packets buffer. Routers are

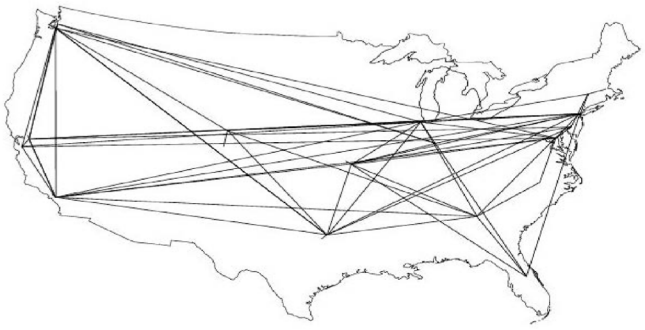


**Fig. 1.** Sprint topology.

connected to all the others by a 1-Mbps bandwidth link but different propagation delays based on the topology file. The router cache size is 5% of the number of all contents in the network, whereas each router uses a least-recently-used cache replacement policy.

In each simulation run, 20 consumer/producer pairs are randomly selected. Each consumer node generates and sends 100,000 interest packets to the network and uses a TCP-like mechanism for congestion control. In this regard, the RTO value is updated using (1)-(3), and the sending window size is increased additively. However, if an RTO timer expires and the requested data is not received by the consumer, the expired interest packets will be retransmitted. In addition, a consumer's sending window size decreases to the minimum size for controlling the presumable network congestion. Table 2 shows the simulation parameters in summary.

## 4. Jacobson RTO estimator evaluation

### 4.1. Varying the minimum RTO

We began our analysis with $RTO_{min}$, which has a major effect on the estimator's behavior as the lower RTO boundary, providing a means of trading off timely retransmissions with premature timeouts. In this analysis, we created different estimators with the same values for $\alpha$, $\beta$, and $k$, based on the standard set by Ref. [30], but $RTO_{min}$ varies in them from 0 to 200 ms. We then assessed these estimators against different popularity distribution gains.

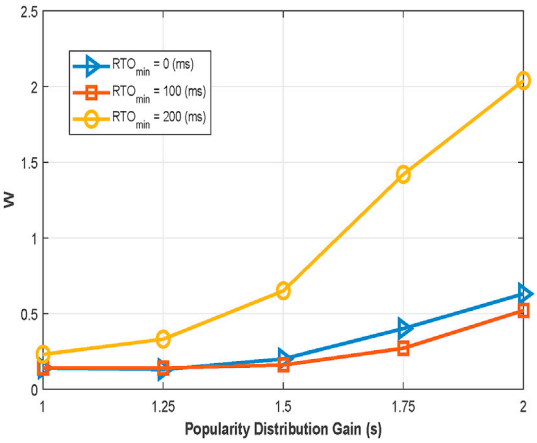
The results are shown in Fig. 2, which indicates that the increasing of $RTO_{min}$ decreases $B$ and increases $W$. Thus, estimators with $RTO_{min} \geq 200$ ms, produce no bad timeouts, but do see $W$ grow rapidly.

To see how these estimators alter network efficiency, the growth rate of consumer download rate for estimators with an $RTO_{min}$ equal to 100 and 200 ms compared to a consumer download rate with no $RTO_{min}$ is shown in Table 3. This table demonstrates that greater RTO estimation accuracy gives a higher consumer download rate. Moreover, avoiding bad timeouts with a conservative estimator produces poor performance because of late retransmissions of lost packets and the inefficient use of network resources.

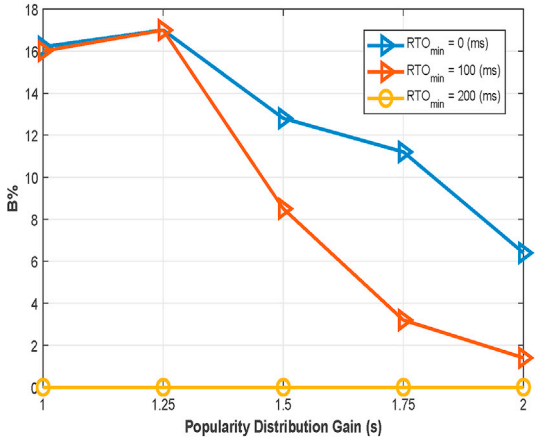
Another important finding from this analysis is that when $s \leq 1.25$, an estimator with $RTO_{min} = 200$ ms achieves the highest performance, whereas when $s > 1.25$, an estimator with $RTO_{min} = 100$ ms attains the highest performance. Fig. 3 shows the reason. This figure indicates the cumulative distributions of RTT and RTO when $RTO_{min}$ is set to 100 ms for different $s$. Based on Fig. 3(a), for high popularity distribution gains, most interests are satisfied at close to 100 ms, but for low popularity distribution gains, RTT values are scattered between 0 and 150 ms. According to Fig. 3(b), the pattern of estimated RTOs is similar to the pattern of RTTs, but only between 100 ms and 200 ms. However, this figure only shows the cumulative distribution of the first RTO estimation and the multiplied RTOs after the first retransmission have been removed.

In general, high popularity distribution gains did not produce bad timeouts for most interest packets because an estimator with $RTO_{min} =$ 100 ms always calculates an RTO higher than 100 ms. These interest packets also had small $W$ value in the case of true timeouts because most RTTs were also near 100 ms.

However, RTTs were more scattered for low popularity distribution gains. Hence, RTO was improperly estimated many times, causing several late and premature retransmissions. In such circumstances, consumers should take a conservative policy to avoid many bad timeout events. However, without care, too many conservative estimators with a large $RTO_{min}$ produces inefficiency and large $W$ values. Fig. 2 indicates that an estimator with $RTO_{min} = 100$ ms successfully balances between these two cases.




(a)




(b)

**Fig. 2.** Effect of varying $RTO_{min}$ ($\alpha = \frac{1}{8}$, $\beta = \frac{1}{4}$, $k = 4$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

### 4.2. Varying EWMA parameters

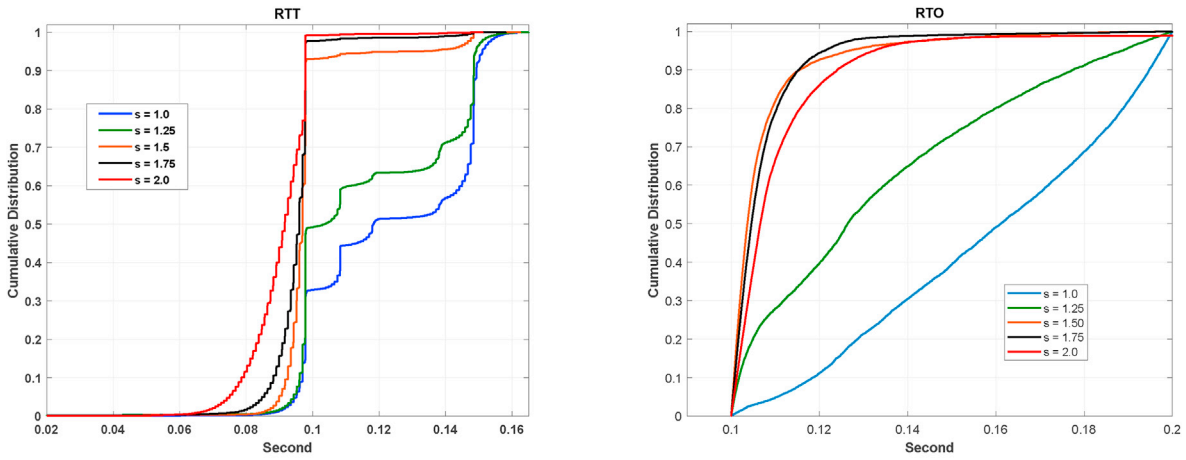
This subsection details our analysis of the Jacobson algorithm to find the most appropriate setting for EWMA parameters in CCN. Toward this, we set $RTO_{min}$ and $k$ to 0 and 4, respectively, except where noted. We then varied the values of $\alpha$ and $\beta$ to create different estimators with "Very

**Table 2**
Simulation parameters.

| Parameter | Value |
|---|---|
| Total number of contents | 10000 |
| Total number of requests | 100000 |
| Queue size | 100 |
| Hop counts | 4–10 |
| Cache size | 5% of total number of contents |

**Table 3**
Growth rate of consumers download rate compared to download rate of estimator with $RTO_{min} = 0$ ms

| Popularity Distribution Gain (s) | $RTO_{min} = 100$ ms | $RTO_{min} = 200$ ms |
|---|---|---|
| | Growth Rate (%) | Growth Rate (%) |
| 1.0 | + 2.44 | + 16.83 |
| 1.25 | + 1.48 | + 5.44 |
| 1.5 | + 2.66 | − 73.63 |
| 1.75 | + 1.42 | − 87.42 |
| 2.0 | + 1.42 | − 85.55 |




(a) (b)

**Fig. 3.** Cumulative distribution of *RTT*s and *RTO*s for estimator with $RTO_{min} = 100$ ms: (a) Cumulative distribution of *RTT*s, (b) Cumulative distribution of *RTO*s.

Slow" ($\alpha = \frac{1}{80}, \beta = \frac{1}{40}$), "Slow" ($\alpha = \frac{1}{16}, \beta = \frac{1}{8}$), "Steady" ($\alpha = \frac{1}{8}, \beta = \frac{1}{4}$), "Fast" ($\alpha = \frac{1}{2}, \beta = 1$), and "Very Fast" ($\alpha, \beta = 1 - TakeLast$) adaption to changes.

Fig. 4 shows the performance of these estimators regarding different content popularity distribution gains. Based on this figure, the estimators that slowly adapted changes estimated RTO more accurately than the estimators that quickly adapted to changes.

Both network path properties and content popularity affect RTT and *RTTVAR* values in CNN. This causes RTO estimating based on a few recent interests to be inaccurate. However, estimators that adapt to changes slowly can calculate RTO based on long period averaging, which reflects overall.

Fig. 5 clearly indicates how estimator with a very slow change adaptation rate estimates an RTO equilibrium value. Fig. 5(a) shows the cumulative distribution of RTTs in a Very Slow estimator for different content popularity distribution gains, clearly illustrating how an increasing $s$ changes the RTT distribution and makes it convergent. In Fig. 5(b), we compare the cumulative distribution of the first estimated RTOs with $s = 1.5$, using different estimators with Very Slow to Very Fast adaption to changes.

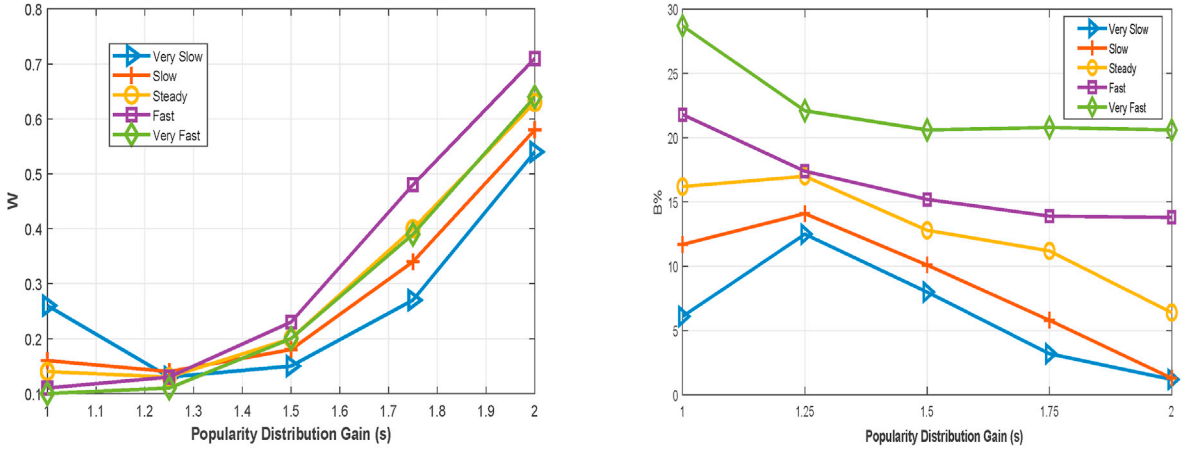
Based on Fig. 5(a), given $s = 1.5$ with a Very Slow estimator, most sent interests were satisfied in approximately 100 ms, so the average RTT was almost equal to 100 ms. As seen in Fig. 5(b), the estimator calculated an RTO close to, but always higher than, 100 ms for most instances. As a result, bad timeouts rarely happened, and interests with good timeouts had small $W$ values.

Generally, a Very Slow estimator produced RTO value close to the average RTT, indicating that the former value is suitable for high popularity distribution gains due to convergent RTTs. While the RTO value did not show good performance with low popularity distribution gains, it is possible to protect the estimator from spurious timeouts, using a proper setting for the multiplier of *RTTVAR*. We will assess this possibility in Subsection 4.3. Prior to that, the interaction between $RTO_{min}$ and EWMA parameters is analyzed.

In Subsection 4.1, we saw that setting the $RTO_{min}$ near to average RTT gives the highest performance. For this reason, we set the $RTO_{min}$ to 100 ms and repeated the analysis on EWMA parameters.

The results are reported in Fig. 6, which shows that increasing the $RTO_{min}$ did not cause a notable difference for Very Slow estimator performance. However, other estimators exhibited an obvious accuracy improvement for large popularity distribution gains. This figure also shows that Very Slow estimator maintained the highest performance for $s \leq 1.25$. However, for $s \geq 1.25$, there was no obvious optimal estimator, and faster estimators only produced greater aggression.

Table 4 indicates the growth rate of consumers download rates with Very Slow, Slow, Fast and Very Fast compared to Steady estimator. Table 4 reflects all above observations, and shows that rapid adaption to changes in high popularity distribution gains achieved a slightly higher




(a) (b)

**Fig. 4.** Effect of varying EWMA parameters ($RTO_{min} = 0$ ms, $k = 4$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

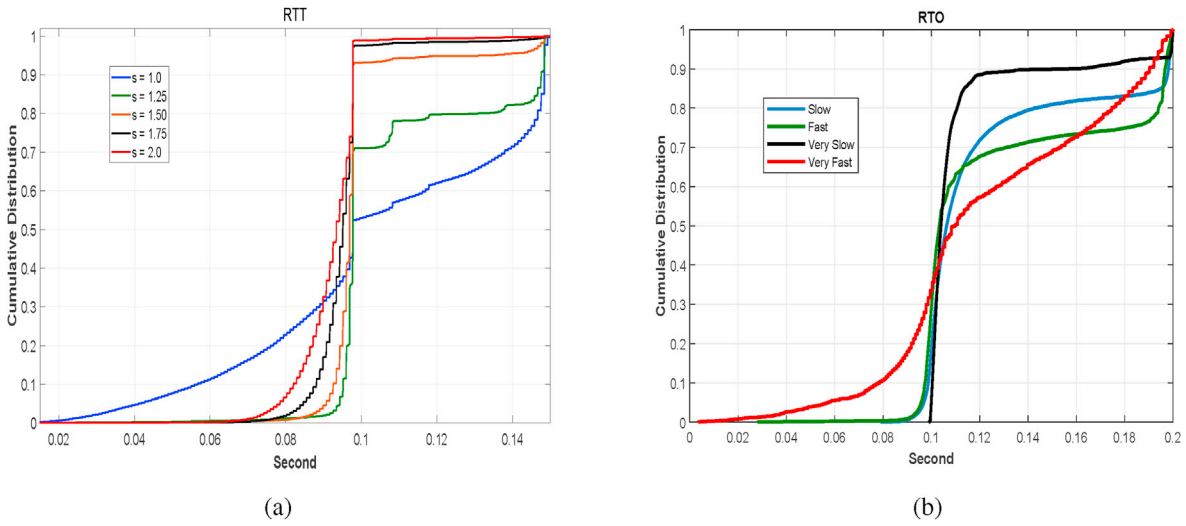



**Fig. 5.** Cumulative distribution of *RTT*s and *RTO*s for "Very Slow" estimator: (a) Cumulative distribution of *RTT*s, (b) Cumulative distribution of *RTO*s in $s = 1.5$.

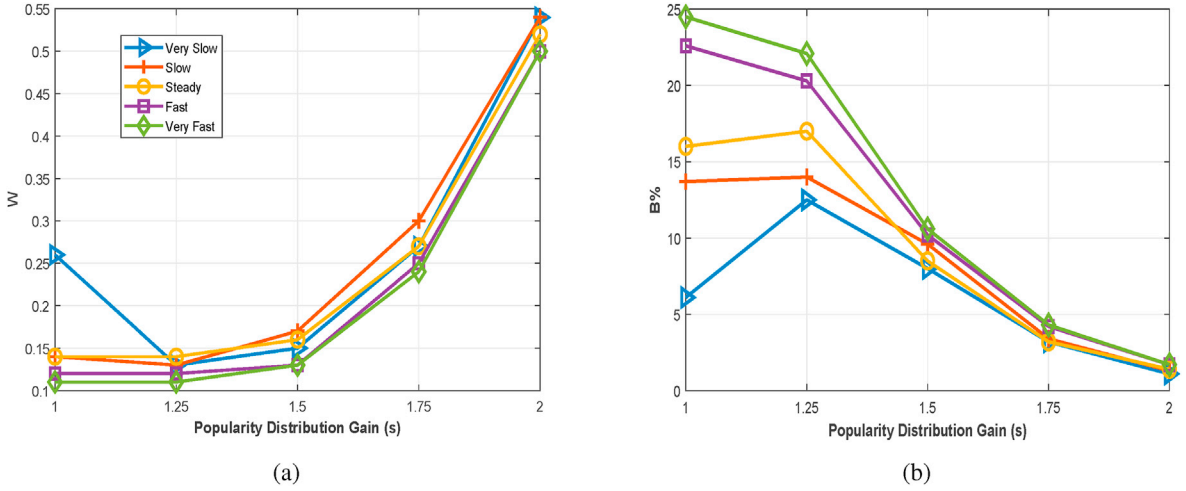



**Fig. 6.** Effect of varying EWMA parameters ($RTO_{min} = 100$ ms, $k = 4$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

download rate because of more convergent content requests for high $s$.

As a general conclusion, we can say that the basic setting of the Jacobson algorithm is too aggressive to use in CCN, and its performance will be improved by slowly adapting to changes and setting the $RTO_{min}$ near the average RTT.

### 4.3. Varying the RTTVAR factor (k)

The final parameter that we consider in RTO estimation is $K$ an *RTTVAR* multiplier that provides a means of trading off waiting time for unnecessary timeouts. Fig. 7 shows the performance of estimators with distinct values for $k$ from 2 to 12. These estimators used EWMA parameters based on standard Jacobson algorithm and $RTO_{min} = 0$ ms.

**Table 4**
Growth rate of consumer download rate compared to steady ($RTO_{min} = 100$ ms).

| Popularity Distribution Gain(s) | Very Slow | Slow | Fast | Very Fast |
|---|---|---|---|---|
| | Growth Rate (%) | Growth Rate (%) | Growth Rate (%) | Growth Rate (%) |
| 1.0 | +8.38 | +2.54 | −6.68 | −12.52 |
| 1.25 | +1.90 | +0.13 | −8.21 | −3.62 |
| 1.5 | +0.80 | +0.07 | −1.66 | −2.74 |
| 1.75 | +0.21 | +0.19 | +0.25 | +0.13 |
| 2.0 | −2.14 | −1.53 | −0.90 | −0.41 |

Fig. 7 indicates that as $k$ increases, bad timeouts decrease but $W$ values also increase to produce no obvious superior estimator. Nevertheless, it is advisable that $k$ be of an optimal value to prevent multiple mistaken timeouts and sluggish transmissions.

Our previous analysis has specified that setting the $RTO_{min}$ near to average RTT, and slowly adapting to changes, improves the estimator performance. Figs. 8 and 9 illustrate the results of varying $k$ in estimators with $RTO_{min} = 100$ ms and Steady change adaption, and with $RTO_{min} = 0$ ms and Very Slow adaption to changes, respectively.

Fig. 8 shows that inappropriate setting of EWMA parameters cannot be compensated by varying $k$, so that in high popularity distribution gains, larger $k$ values increase both $W$ and $B$. However, Fig. 9 indicates that Very Slow change adaption and a correct k setting allows an estimator to attain high performance in all popularity distribution gains. In this scenario, increasing $k$ balanced $B$ and $W$ for low popularity distribution gains while not causing a large $W$ value for high popularity distribution gains. This is because low popularity distribution possesses a large *SRTTVAR*, whereas the high popularity distribution possesses a small *SRTTVAR*.

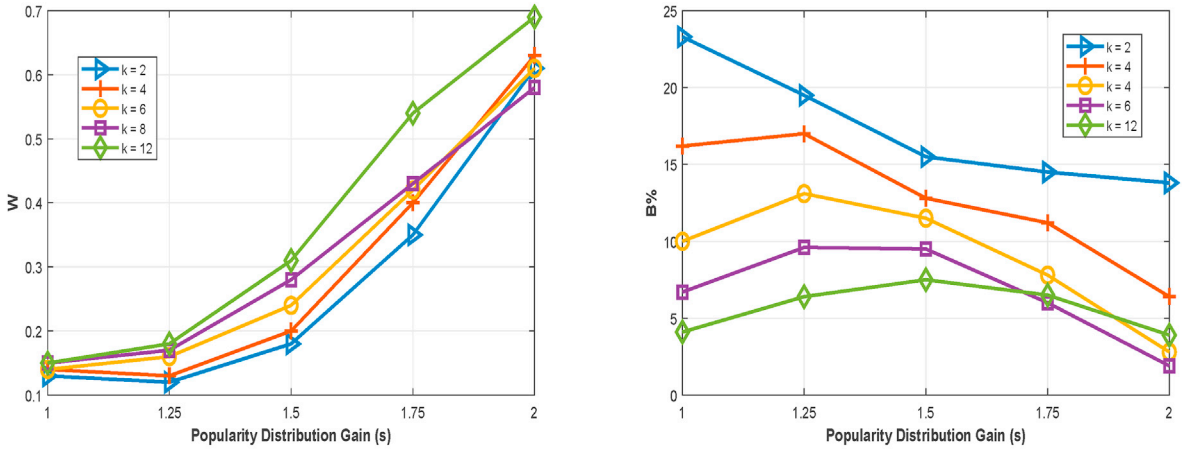



**Fig. 7.** Effect of varying *RTTVAR* factor($k$) ($RTO_{min} = 0$ ms, $\alpha = \frac{1}{8}$, $\beta = \frac{1}{4}$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

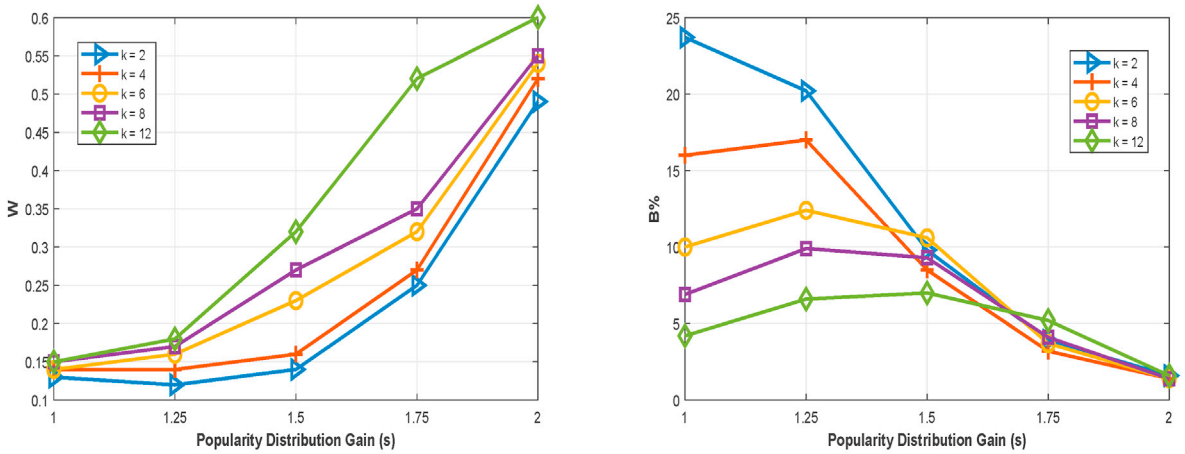



**Fig. 8.** Effect of varying *RTTVAR* factor($k$) ($RTO_{min} = 100$ ms, $\alpha = \frac{1}{8}$, $\beta = \frac{1}{4}$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

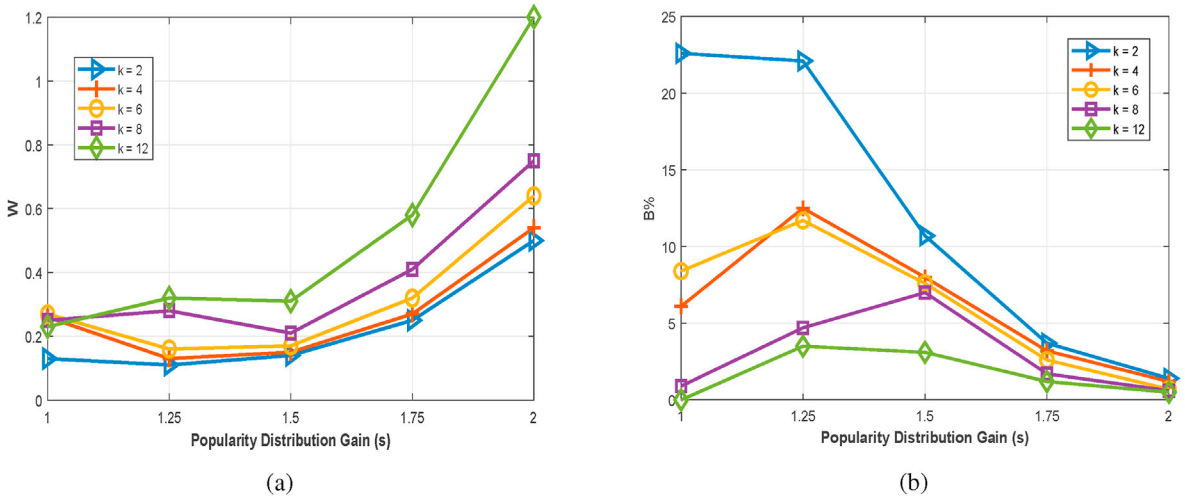



**Fig. 9.** Effect of varying *RTTVAR* factor($k$) ($RTO_{min} = 0$ ms, $\alpha = \frac{1}{80}$, $\beta = \frac{1}{40}$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

## 5. Reconsidering Jacobson algorithm for CCN

Our evaluations and analyses thus far have shown that:

- The accuracy of the Jacobson algorithm in CCN is highly dominated by EWMA parameters, which should be set to small values for long period averaging; this is because expected RTT for the next interest in CCN depends on content popularity as opposed to recently met RTTs.
- The *RTTVAR* factor can efficiently control waiting times to prevent unnecessary timeouts. Estimators that adapt to changes very slowly and have $k$ set to values between 4 and 8, will balance $B$ and $W$ values for low popularity distribution gains and not increase $W$ values for high gains. The *RTTVAR* values fluctuate inversely with the popularity distribution gain.
- Setting the $RTO_{min}$ near average RTT can correct the inappropriate EWMA parameters setting for high popularity distribution gains but cannot improve estimator performance for low popularity distribution gains. Therefore, an advisable strategy is to set small EWMA parameters values and estimate an average RTT for $RTO_{min}$.

Fig. 10 reports the performance of a reconsidered estimator based on these points for different $k$. This figure shows a slightly higher performance compared to Fig. 8, after setting $RTO_{min} = 100$ ms. Based on this figure, a generally efficient setting for the Jacobson algorithm in all popularity distribution gains is possible as long as very slow change adaptation is present, the $RTO_{min}$ is set near to average RTT, and $k = 6$.

Fig. 11 shows the performance of the reconsidered Jacobson estimator compared to the "BasicJacobson" estimator [30] and two other RTO estimators described in Section 2. We named these estimators "PerProvider" [23] and "Predictive" [24].

According to Fig. 11, predictive estimator achieved the highest performance due of its ability to predict the provider of an interest and then reliably estimate RTO using the specific provider's estimator. However, as described in Section 2, this scheme places a long anticipated chunks header in each interest and data packet for processing by each intermediate router. Therefore, the header creates high information and computation overhead. Moreover, based on this scheme's implementation, we considered only an original server for the requested chunks with one route to them, that did not permit requests for aggregation in intermediate routers. Such assumptions are not the usual way CCN works. As a result, the Predictive estimator is not a practical estimator. Therefore, we have implemented and analyzed this scheme for comparison purposes.

It is apparent from Fig. 11 that the reconsidered Jacobson estimator has better performance than the BasicJacobson and PerProvider estimators, in all content popularity distribution gains.

Our results show that using the Jacobson algorithm with basic configuration in CCN causes many spurious timeouts while failing to produce good performance in the case of good timeouts, which consequently decreases network throughput. Maintaining an estimator for each provider cannot resolve this problem due to request aggregation and dynamic forwarding in intermediate routers. Such a method only produces slightly better performance than the basic Jacobson algorithm in large popularity distribution gains. However, the reconsidered Jacobson estimator successfully balanced late and premature retransmissions in all content popularity distribution gains without imposing more information or computation overheads on the nodes or any false assumptions about CCN.

## 6. Conclusions and future works

In this paper, we assessed the performance of the Jacobson algorithm in CCN to propose an accurate RTO estimator for this architecture. For this purpose, we extended a large-scale, inclusive simulation analysis and evaluated the Jacobson estimator in CCN by varying $RTO_{min}$, EWMA parameters, and an RTT variance multiplier against different content popularity distribution gains.

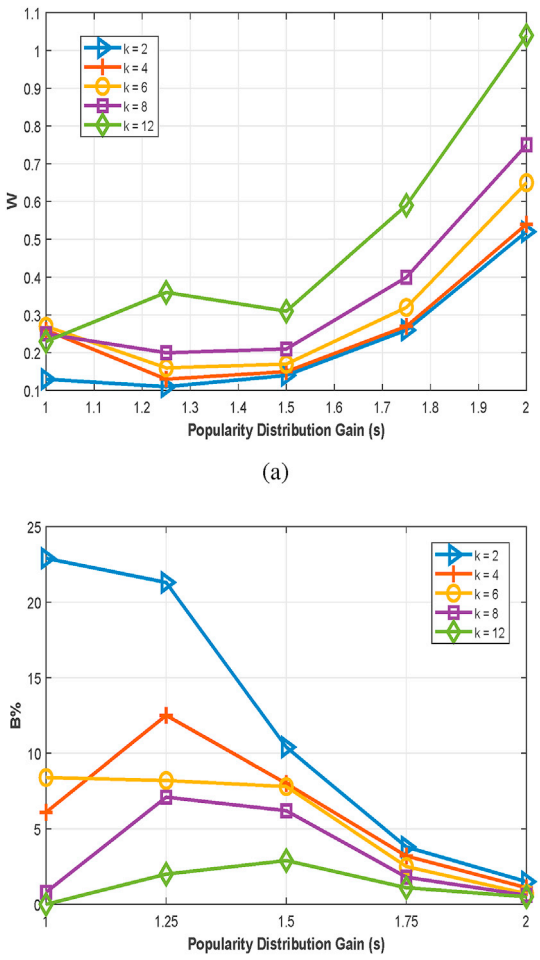


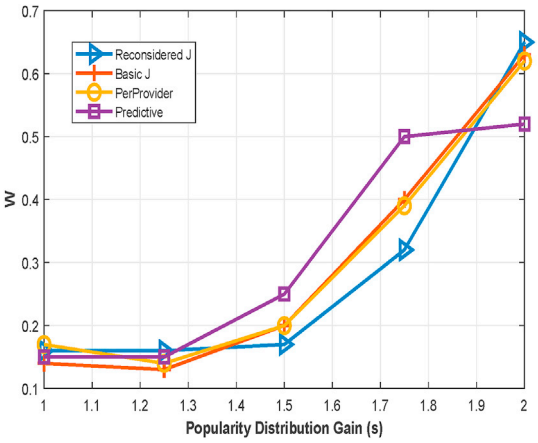

**Fig. 10.** Effect of varying *RTTVAR* factor($k$) in the reconsidered Jacobson estimator ($RTO_{min} = 100$ ms, $\alpha = \frac{1}{80}$, $\beta = \frac{1}{40}$): (a) mean normalized timeout cost ($W$), (b) mean proportion of bad timeouts ($B$).

Our evaluation demonstrated that EWMA parameters have the highest impact on estimator's accuracy and should be set to small values for a long averaging period. Moreover, setting the $RTO_{min}$ near to the average RTT significantly improved estimator performance for high popularity distribution gains. We also found that the RTT variance factor directly affected RTO and should be set according to EWMA parameters to prevent late or premature retransmissions.

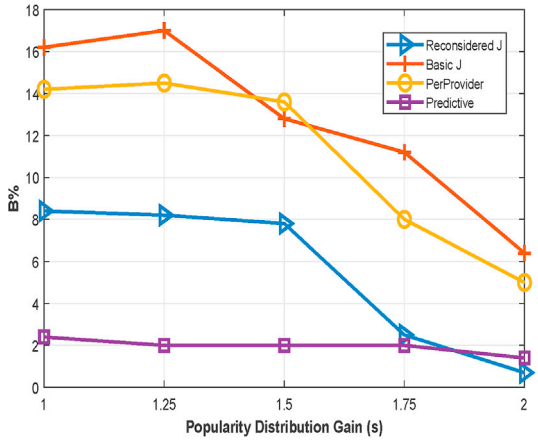
Based on these findings, our reconsidered estimator halved the number of bad timeouts while doubling estimator performance compared to the basic configuration of the Jacobson estimator and other proposed estimators for CCN. It also did not increase information or computation overhead.

The Jacobson algorithm and all other proposed schemes for estimating RTO in CCN are based on recursive linear filtering, making them suitable for estimation in Gaussian signal environments. However, the RTT statistics are often impulsive and cannot be accurately modeled as a Gaussian process; thus, leading to poor RTO performance [17].

Therefore, some novel approaches for RTO estimation in TCP are based on weighted median filters [33] or machine-learning techniques [17], and [34], which are also promising methods for proposig more accurate RTO estimators in CCN. Using these methods to propose more accurate RTO estimators in CCN is our avenue for future works.




(a)




(b)

**Fig. 11.** Comparing the accuracy of the reconsidered Jacobson estimator with "BasicJacobson", "PerProvider" and "Predictive" estimators: (a) mean normalized timeout cost (*W*), (b) mean proportion of bad timeouts (*B*).

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] J. Kurose, Information-centric networking: the evolution from circuits to packets to content, Comput. Network. 66 (2014) 112–120.

[2] S.N.S. Hashemi, A. Bohlooli, Analytical modeling of multi-source content delivery in information-centric networks, Comput. Network. 140 (2018) 152–162.

[3] R. Hou, L. Zhang, T. Wu, T. Mao, J. Luo, Bloom-filter-based request node collaboration caching for named data networking, Cluster Comput. 22 (3) (2019) 6681–6692.

[4] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, J. Cao, Named data networking: a survey, Comput. Sci. Rev. 19 (2016) 15–55.

[5] Q. Chen, R. Xie, F.R. Yu, J. Liu, T. Huang, Y. Liu, Transport control strategies in named data networking: a survey, IEEE Commun. Surv. Tutorials 18 (3) (2016) 2052–2083.

[6] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, B. Zhang, Congestion control in named data networking - a survey, Comput. Commun. 86 (2016) 1–11.

[7] M. Nikzad, K. Jamshidi, A. Bohlooli, A responsibility-based transport control for named data networking, Future Generat. Comput. Syst. 106 (2020) 518–533.

[8] T. Bonald, L. Mekinda, L. Muscariello, Fair throughput allocation in information-centric networks, Comput. Network. 125 (2017) 122–131.

[9] Z. Wang, H. Luo, H. Zhou, J. Li, R2t: a rapid and reliable hop-by-hop transport mechanism for information-centric networking, IEEE Access 6 (2018) 15311–15325.

[10] A. Ndikumana, S. Ullah, K. Thar, N.H. Tran, B.J. Park, C.S. Hong, Novel cooperative and fully-distributed congestion control mechanism for content centric networking, IEEE Access 5 (2017) 27691–27706.

[11] T. Kato, M. Bandai, A hop-by-hop window-based congestion control method for named data networking, in: 2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC), 2018, pp. 1–7.

[12] S.N.S. Hashemi, A. Bohlooli, 3cp: coordinated congestion control protocol for named-data networking, IEEE Trans. Netw. Serv. Manag. 18 (3) (2021) 3918–3932.

[13] V. Jacobson, Congestion avoidance and control, SIGCOMM Comput. Commun. Rev. 18 (4) (1988) 314–329.

[14] M. Allman, V. Paxson, On estimating end-to-end network path properties, in: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '99, ACM, New York, NY, USA, 1999, pp. 263–274.

[15] S. Braun, M. Monti, M. Sifalakis, C. Tschudin, An empirical study of receiver-based aimd flow-control strategies for ccn, in: 2013 22nd International Conference on Computer Communication and Networks (ICCCN), 2013, pp. 1–8.

[16] H. Ben Abraham, P. Crowley, Controlling strategy retransmissions in named data networking, in: Proceedings of the Symposium on Architectures for Networking and Communications Systems, ANCS '17, IEEE Press, Piscataway, NJ, USA, 2017, pp. 70–81.

[17] L. Ma, G.R. Arce, K.E. Barner, Tcp retransmission timeout algorithm using weighted medians, IEEE Signal Process. Lett. 11 (6) (2004) 569–572.

[18] G. Xylomenos, C.N. Ververidis, V.A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K.V. Katsaros, G.C. Polyzos, A survey of information-centric networking research, IEEE Commun. Surv. Tutorials 16 (2) (2014) 1024–1049.

[19] L.A. Adamic, B.A. Huberman, Zipf's law and the internet, Glottometrics 3 (1) (2002) 143–150.

[20] L. Breslau, P. Cao, L. Fan, G. Phillips, S. Shenker, Web caching and zipf-like distributions: evidence and implications, in: INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings, vol. 1, IEEE, 1999, pp. 126–134, vol. 1.

[21] Y. Kim, I. Yeom, Performance analysis of in-network caching for content-centric networking, Comput. Network. 57 (13) (2013) 2465–2482.

[22] S.N. Seyyed Hashemi, A. Bohlooli, Analytical characterization of cache replacement policy impact on content delivery time in information-centric networks, Int. J. Commun. Syst. 32 (18) (2019) e4154.

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

[24] L. Saino, C. Cocora, G. Pavlou, Cctcp: a scalable receiver-driven congestion control protocol for content centric networking, in: 2013 IEEE International Conference on Communications (ICC), 2013, pp. 3775–3780.

[25] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, Named data networking, SIGCOMM Comput. Commun. Rev. 44 (3) (2014) 66–73.

[26] Nsf named data networking project, available [Online] http://www.named-data.net/.

[27] I. Psaras, V. Tsaoussidis, Why tcp timers (still) don't work well, Comput. Network. 51 (8) (2007) 2033–2048.

[28] A. Kesselman, Y. Mansour, Optimizing tcp retransmission timeout, in: P. Lorenz, P. Dini (Eds.), Networking - ICN 2005, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 133–140.

[29] V. Jacobson, Congestion avoidance and control, SIGCOMM Comput. Commun. Rev. 25 (1) (1995) 157–187.

[30] V. Paxson, M. Allman, C. T. R. Timer, Rfc 6298, Computing TCP's Retransmission Timer, 2000.

[31] N. Spring, R. Mahajan, D. Wetherall, Measuring isp topologies with rocketfuel, Comput. Commun. Rev. 32 (4) (2002) 133–145.

[32] S. Mastorakis, A. Afanasyev, L. Zhang, On the evolution of ndnsim: an open-source simulator for ndn experimentation, Comput. Commun. Rev. 47 (3) (2017) 19–33.

[33] L. Yin, R. Yang, M. Gabbouj, Y. Neuvo, Weighted median filters: a tutorial, IEEE Trans. Circuits Syst. II Analog Digital Signal Process. 43 (3) (1996) 157–192.

[34] B.A. Arouche Nunes, K. Veenstra, W. Ballenthin, S. Lukin, K. Obraczka, A machine learning framework for tcp round-trip time estimation, EURASIP J. Wirel. Commun. Netw. 2014 (1) (2014) 47.