

ISMCC: Enhancing Goodput in WANs through In-Network Caching and Multipath Coordination

Yibing Zhao* **, Haojiang Deng* **

*National Network New Media Engineering Research Center, Institute of Acoustics,
Chinese Academy of Sciences, Beijing, China

**School of Electronic, Electrical and Communication Engineering,
University of Chinese Academy of Sciences, Beijing, China

zhaoyb@dsp.ac.cn, denghj@dsp.ac.cn

Abstract—In modern wide-area networks (WANs), the surge of AI-driven traffic, the intensified traffic burstiness, and the difficulty of long-distance loss recovery collectively impose significant stress on data transmission. Although in-network storage and per-packet multipath transmission have become feasible solutions, existing approaches primarily optimize network-level metrics while insufficiently considering end-to-end performance. When packets of the same flow traverse multiple paths or caching nodes, they easily arrive out-of-order at the receiver and severely degrade end-to-end goodput. To address this challenge, we propose ISMCC, a network-layer per-packet congestion control method that jointly optimizes in-network caching and multipath transmission. ISMCC employs a per-interface cache system to absorb excessive traffic and handle varying levels of out-of-order packets. It effectively allocates available bandwidth among multiple next hops based on traffic states. Backpressure signal is utilized to coordinate in-network cache systems and the transport-layer hosts. Evaluation results show that ISMCC significantly reduces out-of-order packets and improves end-to-end goodput over existing network-layer multipath schemes, enabling finer-grained of network resource allocation.

Keywords— In-network caching, multipath, congestion control, out-of-order, WANs.

I. INTRODUCTION

With the development of AI and large-scale data processing services, interconnections among service providers and data centers have grown dramatically. Consequently, wide-area networks (WANs) are facing a surge in traffic demand, which is expected to reach 3280 EB per month by 2033 [1]. Large-scale data transfers in WANs face two major challenges. On the one hand, increasing link capacity shortens flow completion times, thereby amplifying traffic burstiness. On the other hand, long-distance transmission increases the cost of loss recovery. These challenges lead to network congestion and degradation of end-to-end performance. Traditional approaches adjust transmission rates or reroute traffic at end hosts, which does not effectively utilize network resources.

The advancement of middleboxes equips network nodes with computing [3] and storage [4] capabilities, which can be utilized to make forwarding decisions and alleviate congestion.

Prior works on in-network storage include temporary caching of congested traffic [5] and replica placement [6]. The storage size of the caching system can be up to hundreds of GB [4]. In terms of rerouting, existing work such as backpressure routing [7] [8] and LISP routing [9] makes per-hop routing decisions to maximize throughput or meet traffic demands. This type of in-network control provides two main advantages: (1) Compared with source-based sensing and control, network nodes can monitor accurate, real-time network conditions and execute transmission control accordingly; (2) Compared with competition among flows at transport layer, network nodes can allocate network resources more efficiently based on traffic demands and network-layer state.

Existing in-network control schemes primarily optimize network-level metrics such as packet loss and throughput, while insufficiently considering end-to-end performance [10]. When different portions of a single flow traverse heterogeneous states—such as being cached at different locations or routed via alternative paths—packets often arrive out of order at the receiver. In TCP-like scenarios, out-of-order arrivals can trigger rigorous retransmissions and significantly degrade performance. This issue becomes detrimental in a WAN environment, where path diversity is high and delay heterogeneity may reach millisecond-scale. Current mitigation approaches, including sender-side out-of-order transmission [11] and receiver-side reordering [12], offer limited improvement in this environment, and introduce considerable buffering and computational overhead at end hosts.

To address this issue, this paper proposes an In-network Cache & Multipath Congestion Control method (ISMCC). This network-layer method jointly orchestrates in-network caching and multipath transmission to optimize network resources and improve end-to-end goodput (i.e., throughput excluding losses and out-of-order data). In order to maximize link utilization and reduce flow-record overhead, this method makes decisions at the per-packet level and aggregates state information by destinations and delay buckets. When congestion occurs, ISMCC utilizes multipath and cache to enhance the network capacity. Cache systems are deployed at interfaces of network nodes to deal with excess and out-of-

order traffic. ISMCC models network states together with the occupancy of cache system, path delay, transmission rates and traffic states to make fine-grained coordination of cache and multipath transmission. By caching bursty traffic, enabling multipath transmission and constraining disorder at the receiver, ISMCC effectively enhances end-to-end goodput. The main contributions are as follows:

1. This paper proposes a network-layer congestion control method ISMCC. It develops a state machine based on network condition that dynamically orchestrates in-network caching and multipath transmission. Experiment results demonstrate that it significantly enhances the network capacity and end-to-end goodput.

2. ISMCC develops a per-interface cache system that elastically absorbs traffic bursts and constrains out-of-order arrival at receiver. Multipath transmissions are jointly scheduled between cache system and the detour next hop, while backpressure signals coordinate cache systems at different nodes and the sender. Experiment results show that it enables per-packet multipath transmission while substantially reducing out-of-order arrivals at receiver.

II. ISMCC

This section introduces In-network Cache & Multipath Congestion Control (ISMCC), a network-layer congestion control method. We first outline the overall framework of ISMCC and then delve into its key components, which include the cache system, multipath transmission, backpressure signal, and coordination with edge nodes.

A. Framework

The objective of this study is to mitigate localized congestion and improve end-to-end goodput by enhancing the decision-making capability of network nodes. The key functions of ISMCC are summarized as follows:

- **Cache**

Temporarily stores excess packets to smooth traffic bursts, and transmit cached data to fully utilize available bandwidth while mitigating reordering at the receiver.

- **Multipath**

Schedules packets on a per-packet basis between the primary and the detour next hop to mitigate congestion and exploit path diversity.

- **Backpressure**

Propagates congestion signals to upstream nodes, enabling cache system and end hosts to adjust sending behavior and prevent overflow.

These functions are jointly orchestrated at network nodes according to real-time network conditions. When incoming traffic exceeds the link capacity, rather than relying on conventional transport-layer window adjustments, ISMCC caches the excess traffic directly at the interface of the congested router. The cache system is considered as an elastic bloated buffer whose queuing delay increases with the accumulated data. Once the delay offsets the delay difference between the primary and the detour next hop, traffic can be routed on a per-packet basis through the alternative next-hop

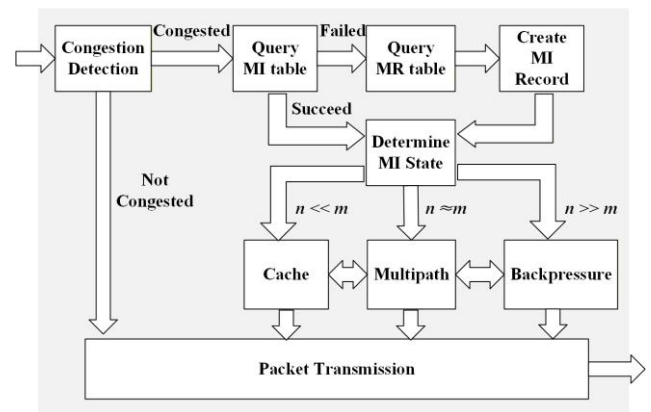


Figure 1. The framework of ISMCC

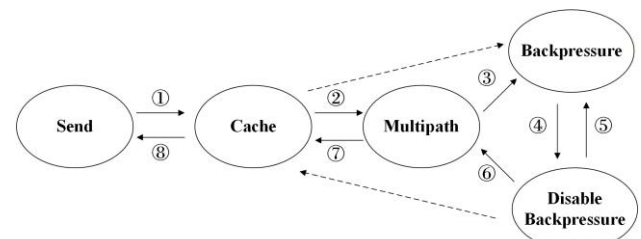


Figure 2. The transition map of MI states. ① The available bandwidth is exhausted. ② n is close to m . ③ n is greater than $m+\lambda$ and the available bandwidth of detour next-hop is exhausted. ④ n drops below m . ⑤ During t , n arises above m . ⑥ During t , n stays below m . ⑦ n drops below $m-\lambda$. ⑧ $n = 0$.

without introducing out-of-order arrival. Through this mechanism, the system alleviates localized congestion while preserving high end-to-end performance.

The next hops for different destinations are pre-computed by existing multipath routing algorithms [13] and stored in the Multipath Routing Table (MR). The alternative next-hop is selected based on latency and available bandwidth. At each network node, the status of traffic is aggregated by destination as Multipath Information (MI) and recorded in MI Table. When the throughput of a MI record approaches the limit of ISMCC at a single network node, backpressure signals are propagated upstream through ACKs. Upon receiving backpressure signals, the corresponding MI record is marked. Once the signal reaches the sender, the transmission rate is throttled accordingly.

Fig. 1 illustrates the collaboration of these components at each interface. Assume that the delay disparity between current path and the detour path is δ , and the sending rate is r . Then the amount of potentially out-of-order data at the receiver is $m = \delta r$. When congestion occurs, the network node evaluates m and the sub-cache occupancy n . When $n \ll m$, ISMCC prioritizes caching to mitigate congestion. When n approaches m , ISMCC initiates per-packet multipath transmission. When $n \gg m$, ISMCC reaches its limit at the current node, and backpressure is triggered. If the MR table indicates no available alternative next hop for a MI record, only cache and backpressure mechanisms are applied.

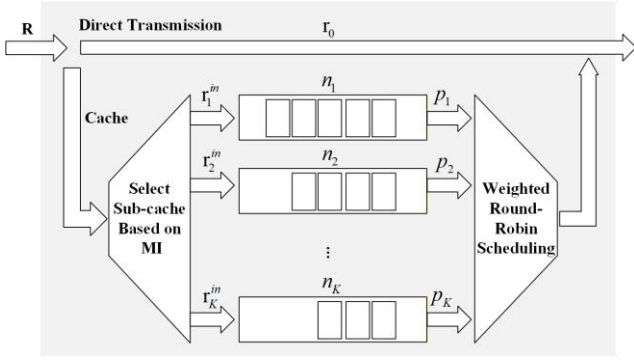


Figure 3. The workflow of cache system

During this process, each MI record maintains five distinct states, each corresponding to a different behavior of ISMCC. The MI state machine is summarized in Fig. 2, and the transition process of the state machine is detailed in the following sections.

B. In-network cache system

ISMCC develops per-interface cache system at each network node. The hardware design of the cache system follows that of [9]. To cope with different levels of out-of-order arrivals, the cache system is divided into K sub-caches, each corresponding to a delay-difference level. The delay difference levels are denoted as

$$\Delta = [\delta_1, \delta_2, \dots, \delta_K]$$

Assuming the sending rate of sub-cache k is r_k^{out} , the sufficient buffer size required for multipath transmission is $m_k = r_k^{out} \delta_k$. Each MI record is assigned to the sub-cache whose delay level closely matches the delay difference of the corresponding detour next hop.

The MR table records the next-hop candidates and their costs for each destination. In this study, the cost of a next hop is proportional to the shortest propagation delay from that next hop to the destination. This definition ensures relatively stable costs while preserving their correlation with path delays. To minimize delay overhead introduced by detouring, the detour next hop with the lowest cost and sufficient residual bandwidth is preferred. If a suitable next hop is identified, delay difference between the chosen next hop and the current next-hop is evaluated and matched with a corresponding sub-cache. If no alternative next hop is available, the packets associated with this MI record do not participate in multipath transmission. Instead, a sub-cache with sufficient residual bandwidth is randomly selected to store the excess traffic. Finally, an MI record including destination, sub-cache index, number of contained flows and MI state is logged in the MI table.

The cache system continuously monitors the occupancy and incoming rate of each sub-cache. The traffic rates are estimated using the Exponentially Weighted Moving Average (EWMA) method [14]. The total input rate of the interface is denoted by R . The rate of traffic that flows directly into the sending buffer is r_0 . The input rates for each sub-caches are

$r_1^{in}, r_2^{in}, \dots, r_K^{in}$, and the corresponding occupancies are $n_1, n_2, \dots, n_K$. The capacity of interface is denoted by C .

Each sub-cache operates in one of three distinct states: *Increase*, *Multipath*, and *Flush*. The *Increase* state is triggered when MI encounters congestion and the occupancy of corresponding sub-cache n is less than m . In this state, congested traffic is rapidly directed into the corresponding sub-cache. The *Multipath* state is activated when congestion persists and n reaches m . In this state, traffic is scheduled on a per-packet basis between the detour next-hop and the current next-hop. The *Flush* state occurs when congestion subsides, during which the sub-cache drains the stored packets. Notice that these three operational states of sub-caches cannot occur simultaneously within the same cache system. When $R < C$, all sub-caches remain in the *Flush* state. When $R > C$, the sub-caches can transition to the *Increase* or *Multipath* states.

The output of sub-cache employs a Weighted Round Robin (WRR) mechanism. The available bandwidth for the cache system is $C - r_0$. Each sub-cache k is assigned a weight p_k . When $R > C$, the sending rate of each sub-cache is $r_k^{out} = p_k(C - r_0)$. Sub-caches operate in two sending modes. In *Best-effort* mode, the sub-caches attempt to utilize all the available capacity. In *Limit* mode, the maximum sending rate of a sub-cache is restricted. The *Limit* mode controls the amount of traffic forwarded to a specific next hop after receiving backpressure signals.

The weights of WRR for the sub-caches are dynamically adjusted at intervals of i . When $\frac{Ri + \sum_{l \in K} n_l}{i} < C - r_0$,

indicating sufficient available bandwidth, the cache system operates in the *Flush* state, and weights are assigned to drain all remaining cached packets. Conversely, when

$\frac{Ri + \sum_{l \in K} n_l}{i} > C - r_0$, weights allocation is guided by a

fairness principle. Suppose sub-cache k contains f_k data

flows. The ideal weight for sub-cache k is $\frac{f_k}{\sum_{l \in K} f_l}$. However,

the current input rate and the occupancy of sub-caches may not able to instantly reach this ideal weight. For the sub-caches with greater input rate and occupancy than the equivalent of ideal weight, the excess traffic is either routed via a detour next-hop in *Multipath* state or handled in *Backpressure* state. For sub-caches operating in the *Increase* state, the sending rate is upper bound by $r^L = r^{in}$ to avoid draining the stored packets or inhibiting congestion window growth. For sub-caches in the *Limit* send states, the weight is upper-bounded by the limited rate r^L . In practice, weights are first assigned to sub-caches that are below their ideal weight allocation, and then the remaining available bandwidth is distributed equally among the other sub-caches. This complete procedure is detailed in Algorithm 1.

Algorithm 1: Weight Update Algorithm

Input: Link capacity C , direct transmission rate r_0 . The input rate r_k , flow number f_k , occupancy n_k , limit rate v_k^L and states of each sub-cache. The subset of each sub-cache state S_{Limit} , $S_{Increase}$, $S_{Multipath}$, S_{Flush} .

Output: The weight p_k for each sub-cache $k \in K$.

```

1: If  $\frac{Ri + \sum_{l \in K} n_l}{i} < C - r_0$  then
2:   For  $k \in S_{Limit}$  do
3:      $p_k = \frac{r_k^L}{C - r_0}$ 
4:   end For
5:   For  $k \in K \setminus S_{Limit}$  do
6:      $p_k = \frac{r_k i + n_k}{Ri + \sum_{l \in k} n_l}$ 
7:   end For
8: Else
9:    $X \leftarrow \phi$ 
10:  For  $k \in S_{Limit}$  and  $k \in S_{Increase}$  do
11:    If  $r_k^L < \frac{f_k}{\sum_{l \in K} f_l} (C - r_0)$  then
12:       $p_k = \frac{r_k^L}{C - r_0}$ 
13:       $X \leftarrow X \cup k$ 
14:    end If
15:  end For
16:  For  $k \in K \setminus X$  do
17:     $p_k = \frac{(C - r_0 - \sum_{l \in X} r_l) f_k}{(C - r_0) \sum_{l \in K \setminus X} f_l}$ 
18:  end For
19: end If
20: return  $\{p_k, k \in K\}$ 

```

During weight adjustment, network nodes should coordinate with transport-layer congestion control mechanism to regulate the congestion window of the sender. Experimental results indicate that when weights of sub-caches are allocated fairly, the endpoints transmission rates converge to their fair shares. However, for sub-caches in the *Increase* mode, the transport-layer additive increase mechanism causes both n and

m to grow simultaneously, preventing the system from entering the *Multipath* state, where n equals m . To address this issue, this paper introduces a Multipath Slow Start mechanism, which is similar to regular slow start mechanism. A multipath slow start threshold is set at the sender, set to the current window size plus the configurable threshold. When the MI state transits into *Multipath*, a 2-bit signal is piggybacked on ACKs. Upon receiving this signal, the sender enters the multipath slow start phase. In this phase, the congestion window grows exponentially until n increases sufficiently to match m , and then triggers the transitions of sub-cache to *Multipath*.

By caching bursty traffic at network nodes, ISMCC reduces packet loss and out-of-order delivery. Compared with transport-layer approaches that react at endpoints, ISMCC shortens the control cycle and enhances the efficiency of network resource allocation.

C. Multipath

ISMCC activates multipath transmission when the capacity of the primary next hop becomes saturated. The selection of detour next hop is described in the previous section. When the occupancy of a sub-cache approaches m , the delay difference between the original next-hop and the detouring next-hop becomes negligible. Under this condition, per-packet multipath scheduling introduces minimal packet reordering. Consequently, the operating state of the sub-cache transitions from *Increase* to *Multipath*, and the MI state switches from Cache to Multipath.

The adjustment of sending rate for each sub-cache causes the fluctuation of m , leading to the oscillations of MI state. To prevent this, ISMCC defines a multipath interval $[m - \lambda, m + \lambda]$ to control the behavior of incoming packets. When the occupancy of the corresponding sub-cache exceeds $m + \lambda$, packets are forwarded to the detour next hop. When it falls below $m - \lambda$, packets are transmitted to the sub-cache. During the multipath transmission, if the input rate decreases or the load of other sub-caches increase, the value of m will decrease, temporarily driving the buffer occupancy above $m + \lambda$. In such cases, packets continue to transfer via the detour next hop until the occupancy falls below $m - \lambda$. The available bandwidth of the detour next hop is constantly monitored. If the buffer occupancy exceeds $m + \lambda$ and the detour available bandwidth is exhausted, the MI state transits to Backpressure.

This multipath method avoids large-scale packet reordering at the receiver, which requires a reordering buffer of size m and introduces application-layer head-of-line blocking. In ISMCC, the required reordering buffer is limited to approximately 2λ , as the sub-cache occupancy bounded within the multipath interval. By ensuring near in-order delivery, ISMCC enables finer-grained bandwidth allocation and improves end-to-end performance.

D. Backpressure

Backpressure is triggered when ISMCC reaches its operational limit on a single network node. If multipath transmission is active, the MI state switches to *Backpressure* when the available bandwidth of the detour next hop is

exhausted and the sub-cache exceeds $m + \lambda$. If multipath is unavailable, the MI state enters to Backpressure once the sub-cache occupancy reaches the same threshold.

When backpressure is triggered, a 2-bit backpressure signal is piggybacked on ACKs and propagated to the upstream node. Upon receiving this signal, the upstream node re-evaluates the MI state and starts to limit the traffic injected to the congested next hop. If the cache operate state of the MI record is in Multipath state, it will switch to Cache state. If it is already in Cache state, the send state of corresponding sub-cache will be switched to *Limit*. The backpressure limit rate r^L is configured to $p_k(C - r_0) - \gamma$, which is current sending rate minus a constant γ . In the beginning, ISMCC utilizes cache system to deal with backpressure signal. If the occupancy of the corresponding sub-cache exceeds the upper bound, MI state switches from Cache to Backpressure, and the backpressure signal are propagated further upstream.

As upstream traffic decreases, congestion at the backpressure node will eventually dismiss. Recovery is detected once the sub-cache occupancy drops below m . At this point, MI state transitions to Disable Backpressure and a recovery timer is started to monitor this process. If no further congestion occurs before the timer expires, Backpressure is considered to dismiss and a 2-bit backpressure-disable signal is piggybacked to the upstream. When receiving this message, the upstream MI state is re-evaluated and switched to Send, Cache or Multipath.

The sender also adapts its congestion control behavior upon receiving the backpressure signal. Under the conventional additive-increase multiplicative-decrease (AIMD) scheme, the backpressure signal is interpreted as a congestion signal, triggering multiplicative decrease to reduce the sending rate. Conversely, a backpressure disable signal indicates congestion alleviation, prompting additive increase to probe available bandwidth.

III. EXPERIMENT

In this section, ISMCC is evaluated through NS3-3.23 simulator. TCP is employed as the transport-layer protocol. The proposed features are integrated into the transport and network layer protocol stack. Simulations are conducted on a PC equipped with an Intel Core i5-12500 processor (6 cores, 2.60 GHz), 16 GB of RAM, and a 512 GB hard drive, running Ubuntu 22.04 operating system.

A. Experiment Settings

The experiments employ simple topologies to facilitate clear observation of system behavior and performance. The topology depicted in Fig. 4(a) is used to evaluate the performance of multiple sub-caches. TCP traffics flow from Node A to Node G and Node A to Node H. The delay for each link across the topology is set to 2ms. The capacity is set to 100 Mbps for links BC, CF, BF, BD, BE, EF, and 300 Mbps for links AB, FG, FH. The topology comprises three paths for these transmissions, with a fixed delay difference of 2ms. Two sub-caches are configured on the interface of link BF at Node

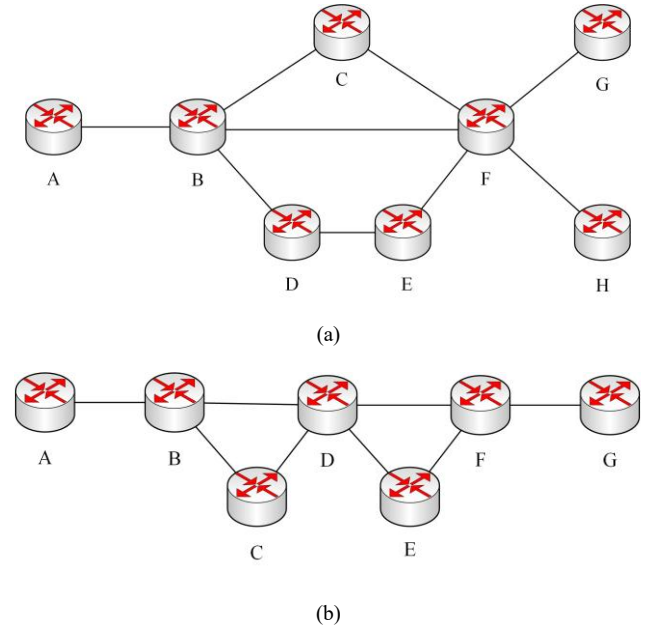


Figure 4. The experiment topology

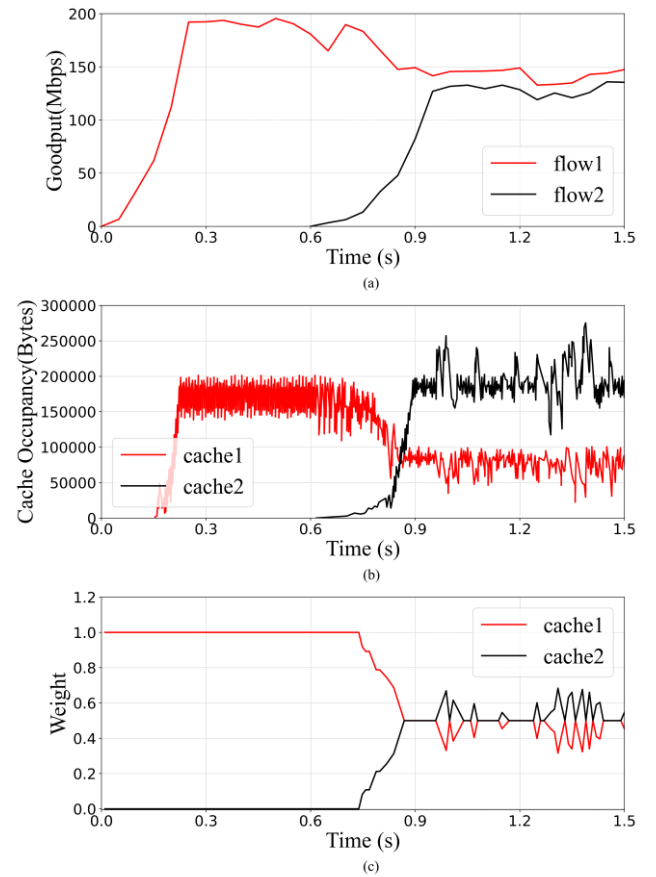


Figure 5. Behavior of ISMCC at a Single Node. (a) Throughput. (b) Occupancy of sub-caches. (c) Weight of sub-caches

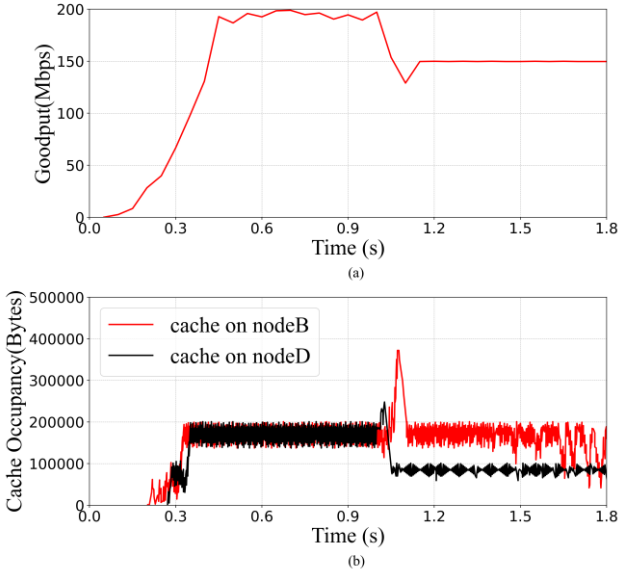


Figure 6. Behavior of ISMCC at two Nodes with Backpressure. (a) Throughput. (b) Occupancy of sub-caches.

B, with $\Delta = [2\text{ms}, 4\text{ms}]$. The parameter λ is set to 10 packets. The topology in Fig.4(b) is utilized to assess the performance of backpressure. TCP traffic originates from Node A and destined to Node G. The link capacity and delay are similar as in Fig. 4(a). There are two different next-hop at Node B and Node D. Therefore, a single sub-cache is configured on the interface of link BD at Node B and link DF at Node D individually, each with a $\Delta = [2\text{ms}]$.

B. Experiment Results

The first experiment evaluates the function of the sub-caches in topology Fig. 4(a). In this setup, two TCP flows are initiated from Node A, destined for Node G and Node H, with the start time set at 0s and 0.6s, respectively. The evaluation metrics include flow goodput, sub-cache sending weight, and sub-cache occupancy. The results are shown in Fig. 5. Fig. 5(a) displays the end-to-end performance. Both Flow 1 and Flow 2 begin in the TCP slow-start phase, exhibiting exponential growth. When Flow1 reaches the capacity of a single path, it transitions to multipath slow-start mode, occupying the space in Sub-cache1. Subsequently, it enters multipath transmission mode to utilize the capacity of two paths. When Flow 2 initiates, the goodput of Flow 1 declines as both flows share the same shortest path. Since Flow 1 had already occupied the lower-delay sub-cache1 ($\delta = 2\text{ms}$), Flow 2 was assigned to the higher-delay sub-cache2 ($\delta = 4\text{ms}$). As the rate of Flow 2 increases, the rate of Flow1 decreased correspondingly. This reduction led to a gradual decline in m value and occupancy of Sub-cache1. Fig. 5(b) and Fig. 5(c) depict the performance of sub-caches. In Fig. 5(b), Flow 1 initially occupies the Sub-cache1. During the multipath transmission, the occupancy is fluctuated within the multipath interval $[m - \lambda, m + \lambda]$. After Flow 2 starts, the m value of Flow 1 gradually decreases while that of Flow2 increases, leading to changes in the occupancy of their respective sub-

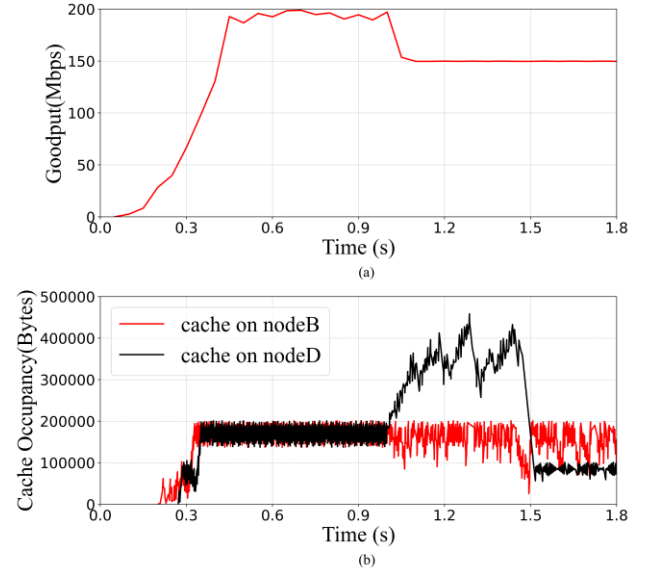


Figure 7. Behavior of ISMCC at two Nodes with Caches. (a) Throughput. (b) Occupancy of sub-caches.

cache. In Fig. 5(c), the sending weights of both sub-caches converge to 0.5, leading to a fair sharing of the available bandwidth on the shortest path, as shown in Fig. 5(a). It can be observed that the weights of sub-caches exhibit slight fluctuations after convergence, which can be attributed to inaccuracies in rate measurement.

The second experiment evaluates the performance of ISMCC between different network nodes in topology Fig. 4(b). A single TCP flow starts from Node A and destined at Node G. For both Node B and Node D, traffic can switch to an alternative next-hop. The TCP traffic initiates at 0 s, and the available bandwidth of link DF is reduced to 50 Mbps at 1.0 s. In the first configuration, Node B and Node D utilize *Backpressure* signal, and the results are shown in Fig. 6. It can be observed that the cache occupancy at both Node B and Node D exhibits a brief spike before stabilizing at their respective m values. Correspondingly, the goodput experiences a minor drop before recovering to utilize the maximum available bandwidth. The *Backpressure* signal is triggered simultaneously with the cache spike, propagating from Node D to Node B. Once the signal reaches Node A, transmission rate is adjusted by transport-layer congestion control. In the second setup (Fig. 7), where backpressure is disabled, the cache of Node D is primarily responsible for absorbing the excessive traffic. Under the control of the cache system, the sending rate is limited within the capacity. However, relying solely on cache system adjustments responses slower than backpressure, while also consuming more cache resources.

The third experiment compares the goodput performance of ISMCC with conventional per-packet random ECMP routing. This evaluation was conducted with a single TCP flow from Node A to Node D in topology Fig. 4(a). Only two paths are utilized. As demonstrated in Fig. 8, ISMCC achieves substantially higher goodput than random ECMP, with this

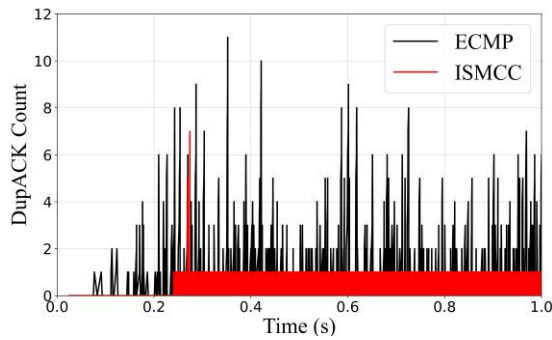


Figure 8. Goodput vs. Time

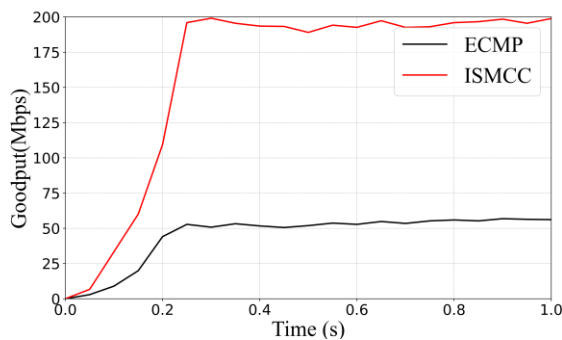


Figure 9. Dupack count vs. Time

advantage emerging from the initial slow-start phase and being maintained thereafter. This improvement stems primarily from ISMCC's ability to significantly reduce out-of-order packet delivery. The reduction of out-of-order packets triggered less unnecessary re-transmission and facilitates constant window growth at transport layer. As shown in Fig. 9, the out-of-order packet is generally within 1 packet for ISMCC, which is significantly lower than that of ECMP.

IV. CONCLUSIONS

In this paper, we address the challenge of end-to-end goodput degradation in WANs, which arises from packet reordering at the receiver when in-network caching and multipath forwarding are deployed independently. Existing in-network schemes often optimize network-level metrics such as throughput or packet loss, while overlooking the detrimental impact of out-of-order delivery on transport-layer performance, especially in TCP-like flows.

To bridge this gap, we proposed ISMCC, a congestion control mechanism that operates at the network layer and jointly orchestrates in-network caching and multipath transmission. By explicitly modeling the relationship between cache occupancy, path delay, and sending rate, ISMCC effectively constrains the amount of out-of-order data arriving at the receiver, thereby significantly improving application-level goodput. The key advantages of this approach include finer-grained resource allocation, reduced control cycle, and compatibility with existing end-host transport protocols.

Experimental results demonstrate that ISMCC achieves up to a threefold improvement in TCP goodput compared to conventional ECMP-based routing, highlighting its ability to make better use of network resources while maintaining fair and stable flow performance.

In future work, we will focus on enhancing the adaptability of ISMCC to complex WAN conditions, such as multi-hop situation.

ACKNOWLEDGMENT

This work was supported by the Basic and Frontier Exploration Project independently deployed by the Institute of Acoustics, Chinese Academy of Sciences: Research on Wide-Area RDMA Transmission Acceleration Technology Based on SEANet (Project No. JCQY202407).

REFERENCES

- [1] Nokia, "Annual traffic report: Global network traffic trends 2022–2025," Nokia, 2022. [Online]. Available: <https://www.nokia.com/asset/213660/>
- [2] R. Bhatia, F. Hao, M. Kodialam, and T. V. Lakshman, "Optimized network traffic engineering using segment routing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Hong Kong, China, Apr. 2015, pp. 657–665.
- [3] B. Zhou, C. Wu, Q. Yang, Z. Zhu, Y. Nie, and W. Yu, "Traffic-aware fast reroute mechanism exploiting disjoint subpaths for named data networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Rome, Italy, May 2023, pp. 4013–4019.
- [4] D. Lin, M. Hamdi, and J. Muppala, "Distributed packet buffers for high-bandwidth switches and routers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1178–1192, Jul. 2012.
- [5] N. Laoutaris, G. Smaragdakis, R. Stanojevic, P. Rodriguez, and R. Sundaram, "Delay-tolerant bulk data transfers on the Internet," *IEEE/ACM Trans. Netw.*, vol. 21, no. 6, pp. 1852–1865, Dec. 2013.
- [6] M. Badov, A. Seetharam, J. Kurose, V. Firoiu, and S. Nanda, "Congestion-aware caching and search in information-centric networks," in **Proc. 1st ACM Conf. Inf.-Centric Netw. (ICN)**, Paris, France, Sep. 2014, pp. 37–46.
- [7] B. Liu, Q. Liang, and E. Modiano, "Tracking MaxWeight: Optimal control for partially observable and controllable networks," *IEEE/ACM Trans. Netw.*, vol. 31, no. 4, pp. 1809–1821, Aug. 2023.
- [8] N. M. Jones, G. S. Paschos, B. Shrader, and E. Modiano, "An overlay architecture for throughput optimal multipath routing," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 2615–2628, Oct. 2017.
- [9] C.-D. Phung, M. Coudron, and S. Secci, "Internet acceleration with LISP traffic engineering and multipath TCP," in *Proc. Conf. Innov. Clouds, Internet Netw. (ICIN)*, Paris, France, Feb. 2018, pp. 1–8.
- [10] S. Rene, O. Ascigil, I. Psaras, and G. Pavlou, "A congestion control framework based on in-network resource pooling," *IEEE/ACM Trans. Netw.*, vol. 30, no. 2, pp. 683–697, Apr. 2022.
- [11] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, O. Mehani, and R. Boreli, "DAPS: Intelligent delay-aware packet scheduling for multipath transport," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Sydney, NSW, Australia, Jun. 2014, pp. 1222–1227.
- [12] P. Huang, X. Zhang, Z. Chen, C. Liu, and G. Chen, "LEFT: Lightweight and fast packet reordering for RDMA," in **Proc. 8th Asia-Pacific Workshop Netw. (APNet)**, Hong Kong, China, Jun. 2024, pp. 67–73.
- [13] K. Schneider, B. Zhang, and L. Benmohamed, "Hop-by-hop multipath routing: Choosing the right nexthop set," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Toronto, ON, Canada, Jul. 2020, pp. 2273–2282.
- [14] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, and G. Varghese, "CONGA: Distributed congestion-aware load balancing for datacenters," in *Proc. ACM SIGCOMM Conf.*, Chicago, IL, USA, Aug. 2014, pp. 503–514.